COLING 2014

The 25th International Conference on Computational Linguistics

Proceedings of COLING 2014: Technical Papers

August 23-29, 2014
Dublin, Ireland

ISBN 978-1-941643-26-6

Preface

This volume contains papers from the 25th International Conference on Computational Linguistics (Coling 2014) held in Dublin, Ireland. The conference is organized by the Centre for Global Intelligent Content (CNGL) and held at the Helix Conference Centre at Dublin City University (DCU) from 25–29 August 2014, under the auspices of the International Committee on Computational Linguistics (ICCL).

COLING is almost 50 years old, its first gathering having taken place in New York in 1965. It has been organized once every two years, initially in odd years and then in even years, after COLING 1976 in Ottawa. Throughout its long history, COLING's aspiration to provide an amicable forum for participants with broad backgrounds to present and share their ideas remains the same. We believe that the inherent complexity of language is worthy of study from diverse perspectives and that COLING provides a venue for fruitful interdisciplinary interaction.

We accepted 217 papers (138 oral presentations and 79 poster presentations) from 685 effective submissions, having received 705 submissions in total. Regardless of the format of presentation, all of the accepted papers were allocated 12 pages in the proceedings.

The review process of a large conference such as COLING is always complex and occasionally encounters difficulties. The program committee has to cope with the challenges of selecting which papers to accept among a large quantity of high quality submissions. The task of choosing 217 papers from 685 strong submissions covering the ever broadening fields of computational linguistics was not an easy one.

To cope with the anticipated difficulties, we asked six senior colleagues to join the Scientific Advisory Board (SAB) and help us through all stages of reviewing papers. They are: Ralph Grishman (New York University, USA), Yuji Matsumoto (NAIST, Japan), Joakim Nivre (Uppsala Univ., Sweden), Michael Picheny (IBM TJ Watson Research Center, USA), Donia Scott (Univ. of Sussex, UK), and Chengqing Zong (CAS, China).

We had 20 thematic areas and each area was chaired by two or more area chairs. Thanks to over 800 responsive reviewers, the review process proceeded in a very smooth manner, and each paper was read at least by three reviewers. In some cases, papers and their reviews were carefully assessed by Area Co-Chairs, one of the SAB members and by us, in our roles as Program Committee Co-Chairs. We are extremely happy with the very strong set of papers that has been accepted for presentation at the conference. It is, however, with regret that we had no choice but to reject a large number of high quality papers, due to the sheer volume of submissions received.

We would like to thank the SAB members and the Program Committee Area Chairs for their dedicated and efficient review work, and our reviewers for their professionalism in delivering high quality reviews. We also thank the authors of all the papers for submitting their fruits of labour to COLING. Although we were only able to accept a small subset of the submitted papers, we do hope that all authors and reviewers have benefited from this process of indirect dialogue.

Last but not least, we would like to thank the people who made COLING 2014 and this volume possible. We thank General Chairs, Josef van Genabith (Universität des Saarlandes/DFKI) and Andy Way (CNGL, DCU), and the chairs of the Local Organizing Committee, Cara Green (CNGL, DCU) and John Judge (CNGL/NCLT, DCU), for their tireless work. We are especially grateful to the Publications Chairs, Joachim Wagner (CNGL, DCU), Liadh Kelly (CNGL, DCU) and Lorraine Goeuriot (CNGL, DCU), for their hard work in preparing the proceedings.

Prof. Jan Hajic (Charles University, Czech Republic)
Prof. Junichi Tsujii (Microsoft Research, China)
COLING 2014 Program Committee Co-Chairs
July 8, 2014

Welcome from the General Chairs

We are very pleased indeed to welcome you all to COLING 2014, the 25th International Conference on Computational Linguistics. We are particularly proud that the ICCL selected Dublin City University (DCU) as the location of COLING 2014.

DCU and its National Centre for Language Technology (NCLT) have a long track record in NLP. Unlike India, the previous COLING host country, Ireland is a very small country. A unique feature of the Irish University landscape is that universities team up with industry partners and each other to pool expertise to form large research centres. DCU is a founding member of CNGL, the Centre for Gobal Intelligent Content. COLING 2014 is organised by DCU in partnership with the CNGL, and as General Chairs we are proud to represent both DCU and CNGL.

The conference is taking place at the Helix Conference Centre, a stunning building added to the DCU campus in 2002. DCU is a young, dynamic and ambitious university; since admitting its first students in 1980, DCU has grown in both student numbers and size and now occupies a 72-acre site in Glasnevin, just to the north of Dublin city centre. To date almost 50,000 students have graduated from DCU and are now playing significant roles in enterprise and business globally. Today in 2014, DCU delivers more than 200 programmes to over 12,000 students across its four faculties — Humanities and Social Sciences, Science and Health, Engineering and Computing and DCU Business School. DCU's excellence is recognised internationally and it is ranked among the top-50 young Universities worldwide (QS 'Top 50 under 50' 2013). In the last eight years, DCU has twice been named Sunday Times 'University of the Year'.

At the time of writing, the total number of people registered to attend COLING has exceeded 675. With delegates from 58 countries, COLING 2014 will witness a colourful diversity of language and culture, which is appropriate given that Dublin is known as the localisation capital of the world. Some evidence for this comes from our sponsors, to whom we are extremely grateful: Baidu, eBay, Microsoft, Symantec and Google.

We are very pleased with the programme that has been assembled for you, comprising of four days for the main conference with a total of 138 oral presentations, 79 posters and a special track with 28 demo presentations, two days of workshops and tutorials before the main conference, and other satellite workshops immediately after. 18 topical workshops with a sharp focus on issues of key interest today will be attended by about 191 delegates, and the 6 high-quality tutorials are sure to attract large crowds. Social events include a welcome reception on the evening of 24th August, the conference banquet in the Guinness Storehouse on 26th, and excursions to some beautiful places of interest on 27th.

When DCU was awarded COLING two years ago, our own personal situations were quite different. One of us was away working in the translation industry in the UK, while the other was leading the Science Foundation Ireland and Industry-funded CNGL research center. Over the past few months, we have changed countries, and jobs: Andy is back as Deputy Director of the CNGL's Centre for Intelligent Content, while Josef has moved to Saarbrücken to take up a Chair and a Scientific Directorship at DFKI.

While these changes were taking place, we both had the backing of a remarkable team. The organization of an event on the scale of COLING takes enormous energy, planning and commitment from a large number of individuals. We have assembled a large, competent team of volunteers who are available to assist you while you are here in Dublin. We are sure that all of you participating at COLING — at tutorials, workshops, or the main conference — will enjoy the time you spend here in Ireland, and will look back on the event as one of the most memorable that you attend. Finally, thanks to all of you for coming. We hope you all enjoy the conference, that you benefit from the excellent programme that has been assembled, and that you go away from here having made new friends.

Prof. Josef Van Genabith (Universität des Saarlandes/DFKI, Germany)
Prof. Andy Way (CNGL, DCU, Ireland)

Organisers

General Chairs

Prof. Andy Way, CNGL, Dublin City University, Ireland

Prof. Josef van Genabith, Universität des Saarlandes/DFKI, Germany

Programme Chairs

Prof. Junichi Tsujii, Microsoft Research, China

Prof. Jan Hajic, Charles University, Czech Republic

Workshops Chairs

Dr. Jennifer Foster, CNGL, Dublin City University, Ireland

Prof. Dan Gildea, University of Rochester, USA

Prof. Tim Baldwin, University of Melbourne, Australia

Publication Chairs

Dr. Joachim Wagner, CNGL, Dublin City University, Ireland

Dr. Liadh Kelly, CNGL, Dublin City University, Ireland

Dr. Lorraine Goeuriot, CNGL, Dublin City University, Ireland

Publicity Chairs

Dr. Dorothy Kenny, SALIS, Dublin City University, Ireland

Prof. Seong-Bae Park, Kyungpook National University, Korea

Local Chairs

Dr. Cara Greene, CNGL, Dublin City University, Ireland

Dr. John Judge, CNGL / NCLT, Dublin City University, Ireland

Scientific Advisory Board

Prof. Joakim Nivre, Uppsala Univ., Sweden

Prof. Yuji Matsumoto, NAIST, Japan

Dr. Michael Picheny, IBM TJ Watson Research Center, USA

Prof. Donia Scott, Univ. of Sussex, UK

Prof. Chengqing Zong, CAS, China

Prof. Ralph Grishman, New York University, USA

Tutorial Chairs

Prof. Qun Liu, CNGL, Dublin City University, Ireland

Prof. Fei Xia, University of Washington, USA

Demo Chairs

Dr. Lamia Tounsi, CNGL, Dublin City University, Ireland

Dr. Rafal Rak, NaCTeM, University of Manchester, UK

Sponsorship Chairs

Dr. Páraic Sheridan, CNGL, Dublin City University, Ireland

Prof. Hans Uszkoreit, DFKI, Germany

Dr. Huaping Zhang, Beijing Institute of Technology, China

Dr. Hiromi Nakaiwa, NTT, Japan

Programme Committee

Programme Chairs

Junichi Tsujii, Microsoft Research, China
Jan Hajic, Charles University, Czech Republic

Linguistic Issues in CL and NLP

Emily M. Bender, University of Washington, USA
Eva Hajicova, Charles University, Czech Republic
Igor Boguslavsky, Universidad Politecnica de Madrid, Spain

Machine Learning for CL and NLP

Jason Eisner, Johns Hopkins University, USA
Yoshimasa Tsuruoka, University of Tokyo, Japan

Cognitive Issues in CL and NLP

Philippe Blache, CNRS & Université d'Aix-Marseille, France
Ted Gibson, MIT, USA

Morphology, Word Segmentation, Tagging and Chunking

Reut Tsarfaty, Weizmann Institute of Science, Israel
Yue Zhang, Singapore University of Technology and Design, Singapore

Syntax, Grammar Induction, Syntactic and Semantic Parsing

Laura Kallmeyer, Heinrich-Heine-Universität, Germany
Ryan McDonald, Google, USA

Lexical Semantics and Ontologies

Chu-Ren Huang, Hong Kong Polytechnic University, Hong Kong
Alessandro Oltramari, Carnegie Mellon University, USA

Semantic Processing, Distributional Semantics and Compositional Semantics

Stephen Clark, University of Cambridge, UK
Alessandro Lenci, University of Pisa, Italy

Modeling of Discourse and Dialogue

Nicolas Asher, CNRS & Université Paul Sabatier, France
Marilyn Walker, University of California Santa Cruz, USA

Natural Language Generation and Summarisation

Albert Gatt, University of Malta, Malta
Advaith Siddharthan, University of Aberdeen, UK

Paraphrasing and Textual Entailment

Ido Dagan, Bar Ilan University, Israel
Kentaro Inui, Tohoku University, Japan

Sentiment Analysis, Opinion Mining and Social Media

Rada Mihalcea, University of Michigan, USA
Bing Liu, University of Illinois at Chicago, USA

Information Retrieval and Question Answering

Gareth Jones, CNGL, Dublin City University, Ireland
Siddharth Patwardhan, IBM Research, USA

Information Extraction and Database Linking

Danushka Bollegala, University of Liverpool, UK

Seung-won Hwang, Postec, Korea

Applications

Srinivas Bangalore, AT&T Labs-Research, USA

Heyan Huang, Beijing Institute of Technology, China

Guillaume Jacquet, Joint Research Centre, Italy

Multimodal and Natural Language Interfaces and Dialog Systems

Kristiina Jokinen, University of Helsinki, Finland

David Traum, University of Southern California, USA

Speech Recognition, Text-To-Speech, Spoken Language Understanding

Nick Campbell, Trinity College Dublin, Ireland

Alex Potamianos, National Technical University Crete, Greece

Machine Translation

Phillip Koehn, University of Edinburgh, UK / Johns Hopkins University, USA

Chris Quirk, Microsoft Research, USA

Tiejun Zhao, Harbin Institute of Technology, China

Resources

Pushpak Bhattacharyya, IIT Bombay, India

Nicoletta Calzolari, ILC-CNR, Italy

Martha Palmer, University of Colorado, USA

Languages with less resources

Steven Bird, University of Melbourne, Australia

Mark Liberman, University of Pennsylvania, USA

Rajeev Sangal, IIT Banaras Hindu University, India

Koenraad De Smedt, University of Bergen, Norway

Software and Tools

Jesús Cardeñosa, Universidad Politecnica de Madrid, Spain

Jing-Shin Chang, National Chi Nan University, Taiwan

Reviewers

Adam Meyers, Adam Pease, Adam Przepiórkowski, Adeline Nazarenko, Adrian Iftene, Afra Alishahi, Afsaneh Fazly, AiTi Aw, Alan Akbik, Alan Ritter, Aleksandrs Berdicevskis, Alessandra Zarcone, Alessandro Moschitti, Alessandro Oltramari, Alessandro Valitutti, Alexander Allauzen, Alexander Clark, Alexander M. Rush, Alexandra Balahur, Alexandra Birch, Alexandre Bouchard, Alexandre Rademaker, Alexandros Potamianos, Alexey Bogdanov, Alexis Nasr, Alexis Palmer, Alice Oh, Aline Villavicencio, Aljoscha Burchardt, Alvaro Rodrigo, Amanda Stent, Amba Kulkarni, Amin Mantrach, Amir Razavi, Amitava Das, Amit Goyal, Ananthakrishnan Ramanathan, Anastasia Krithara, Anders Søgaard, Andrea Gesmundo, Andrea Schalley, Andrew Kehler, Ang Sun, Ani Nenkova, Anirban Dasgupta, Anna Korhonen, Annalina Caputo, Anna Rumshisky, Anna Stavrianou, Ann Bies, Annemarie Friedrich, Annie Zaenen, Anoop Sarkar, Anselmo Peñas, Antal van den Bosch, António Branco, Antonio Pareja-Lora, Antonio Toral, Arantza Diaz de Ilarraza, Aravind Joshi, Ariadna Quattoni, Arne Mauser, Aron Culotta, Arul Menezes, Arulmozi Selvaraj, Arvind Agarwal, Arzucan Özgür, Asli Celikyilmaz, Athanasios Katsamanis, Atsushi Fujii, Atsushi Fujita, Aurélie Herbelot, Avirup Sil, Barbara Di Eugenio, Barbara Hemforth, Barbara Plank, Barry Haddow, Beatrice Alex, Beatrice Daille, Benjamin Roth, Ben King, Benoit Crabbé,

Bernd Bohnet, Bill Byrne, Bill Keller, Bishan Yang, Bonan Min, Bonaventura Coppola, Borja Navarro, Bo Tan, Boyan Onyshkevych, Branimir Boguraev, Brian Roark, Brigitte Grau, Canasai Kruengkrai, Caner Derici, Carlos Busso, Carlos Gómez-Rodríguez, Carmen Banea, Carolina Gallardo, Caroline Brun, Caroline Hagege, Cecile Paris, Cem Akkaya, Cettolo Mauro, Changxing Wu, Chen Chen, Chengqing Zong, Chi-Ho Li, Chris Brew, Chris Callison-Burch, Chris Fox, Christian Buck, Christiane Fellbaum, Christian Scheible, Christina Lioma, Christina Sauper, Christof Monz, Christopher Mitchell, Christopher Potts, Christoph Lofi, Christoph Tillmann, Christos Christodoulopoulos, Chu-Ren Huang, Ciprian Chelba, Claire Bonial, Claire Gardent, Claudia Leacock, Colin Cherry, Conghui Zhu, Constantin Orasan, Corina Forascu, Cornelia Caragea, Costanza Navarretta, Cristina Bosco, Cyril Goutte, Daisuke Kawahara, Damianos Karakos, Dan Bikel, Dan Flickinger, Dan Garrette, Dan Goldwasser, Daniel Gildea, Daniel Gillick, Danilo Croce, Dan Lassiter, Dan Tufiş, Danushka Bollegala, Daoud Clarke, David Burkett, David Chiang, David Graff, David Hall, David Kauchak, David McClosky, David Reitter, David Schlangen, David Talbot, David Vilar, David Weir, Deepak Ramachandran, Degen Huang, Dekai Wu, Delip Rao, Derek F. Wong, Deyi Xiong, Diana Inkpen, Diana McCarthy, Diarmuid Ó Séaghdha, Diego De Cao, Diego Molla, Dietmar Zaefferer, Dipanjan Das, Dipti Sharma, Dirk Hovy, Djamé Seddah, Dominique Laurent, Dongdong Zhang, Donghong JI, Donia Scott, Donna Byron, Dorothee Beermann, Doug Downey, Dwaipayan Roy, Edmundo Pavel Soriano Morales, Eduard Dragut, Edward Grefenstette, Egoitz Laparra, Ehud Reiter, Eiji Aramaki, Ekaterina Ovchinnikova, Ekaterina Shutova, Eleftherios Avramidis, Elena Cabrio, Elena Kozerenko, Elena Lloret, Elif Aktolga, Ellen Riloff, Ellen Voorhees, Emiel Krahmer, Emily Mower Provost, Emily Pitler, Endong Xun, Eric de la Clergerie, Erik Cambria, Eugenio Martinez-Camara, Eunyoung Ha, Eva Hajicova, Eva Maria Vecchi, Evelina Fedorenko, Fabio Celli, Fabio Massimo Zanzotto, Fabre Cécile, Fangtao Li, Farah Benamara, Federico Alberto Pozzi, Feifei Zhai, Fei Huang, Felix Hieber, Fernando Martínez-Santiago, Francis Bond, Francisco Casacuberta, Francisco Guzman, Francis Tyers, François Portet, François Yvon, Franco M. Luque, Frank Keller, Franz Och, Furu Wei, Gabriella Lapesa, Ganesh Ramakrishnan, Gary Geunbae Lee, Geli Fei, Gemma Boleda, George Foster, Georgiana Dinu, Gerald Penn, Gerard de Melo, Gerard Kempen, German Rigau, Gholamreza Haffari, Gianluca Lebani, Gina-Anne Levow, Giovanni Semeraro, Girish Jha, Giuseppe Attardi, Giuseppe Carenini, Giuseppe Di Fabbrizio, Giuseppe Riccardi, Graeme Hirst, Graham Neubig, Grazyna Demenko, Gregor Leusch, Grzegorz Chrupała, Guangyou Zhou, Guenter Neumann, Guergana Savova, Guillermo Garrido, Guodong Zhou, Guo-Wei Bian, Guy Lapalme, Haifeng Wang, Hailong Cao, Haitao Mi, Haixun Wang, Hai Zhao, Hany Hassan, Hao Zhang, Harksoo Kim, Harold Somers, Helmut Horacek, Hema Raghavan, Heng Ji, Hidekazu Oiwa, Hideto Kazawa, Hikaru Yokono, Himanshu Sharma, Hinrich Schuetze, Hiroshi Kanayama, Hiroya Takamura, Hiroyuki Shindo, Hitoshi Nishikawa, Hoa Trang Dang, Holger Schwenk, Hongwei Ding, Houfeng Wang, Hristo Tanev, Hsin-Hsi Chen, Hua Wu, Hui Fang, Hwee Tou Ng, Hyuckchul Jung, Idan Szpektor, Ielka van der Sluis, Ilyas Cicekli, Ines Rehbein, Ioannis Korkontzelos, Irina Kobozeva, Iryna Gurevych, Ivan Habernal, Jaakko Vaeyrynen, Jagadeesh Jagarlamudi, James Fan, James Mayfield, Jan Niehues, Jan Šnajder, Jan van Santen, Janyce Wiebe, Jarmila Panevova, Jason D Williams, Jason Naradowsky, Jean-Gabriel Ganascia, Jean Senellart, Jeff Mitchell, Jennifer Chu-Carroll, Jennifer Culbertson, Jennifer Gillenwater, Jennifer Williams, Jens Edlund, Jeremy Reffin, jerry feldman, Jiajun Chen, Jiajun Zhang, Jianfeng Gao, Jian Su, Jianxing Yu, Jian Xu, Jill Burstein, Jim Blevins, Jingbo Zhu, Jing Jiang, Jingsong Su, Jinho D. Choi, Ji-Rong Wen, Joel Nothman, Joern Wuebker, Johan Bos, Johannes Hoffart, John Carroll, John Chen, John Hale, John Prager, Jonas Kuhn, Jonathan Ginzburg, Jongwuk Lee, Jordi Atserias Batalla, Jörg Tiedemann, Jose Angel Olivas, José B. Mariño, Josef Ruppenhofer, Josef van Genabith, Joseph Le Roux, Jose Pinto, Joyce Chai, Juergen Trouvain, Julien Ah-Pine, Julie Weeds, Jungi Kim, Jungyeul Park, Jun-Ping Ng, Jun Suzuki, Junyi Jessy Li,

Jun Zhao, Juri Ganitkevitch, Jyoti Pawar, Kaiqi Zhao, Kai-Wei Chang, Kai Zhao, Kalina Bontcheva, Kam-Fai Wong, Kang Liu, Kangqi Luo, Kareem Darwish, Kashif Shah, Kathy McKeown, Katja Markert, Katrin Erk, Kees van Deemter, Keh-Yih Su, Keikichi Hirose, Keisuke Sakaguchi, Keith Hall, Keith Vander Linden, Kemal Oflazer, Ken Church, Kenji Sagae, Kenny Zhu, Kentaro Torisawa, Kenton Murray, Kevin Cohen, Kevin Duh, Kevin Gimpel, Kevin Knight, Kevin Small, Key-Sun Choi, Khalil Sima'an, Khiet Truong, kim gerdes, Kiril Simov, Klaus Macherey, Koenraad De Smedt, Konstantin Zuev, Krasimir Angelov, Kristian Woodsend, Kristina Striegnitz, Kristy Boyer, Kumiko Tanaka-Ishii, Kuzman Ganchev, L. Alfonso Urena Lopez, Lane Schwartz, Laurent Besacier, Laurent Prévot, Laure Vieu, Lei Cui, Leila Kosseim, Lei Zhang, Lemao Liu, Leonid Iomdin, Leo Wanner, Le Zhao, Liang-Chih Yu, Lidia Pivovarova, Lidia S. Chao, Li Dong, Liheng Xu, Likun Qiu, Limin Yao, Liu Zhanyi, Livia Polanyi, Lorenzo Ferrone, Louise McNally, Lucia Specia, Lucy Vanderwende, Luis Espinosa, Luke Zettlemoyer, Lun-Wei Ku, Lu Qin, Maayan Zhitomirsky-Geffet, Magnus Sahlgren, Maja Popović, Makoto Miwa, Malvina Nissim, Mamoru Komachi, Mandar Mitra, Manfred Pinkal, Manfred Stede, Marcello Federico, Marco Baroni, Marco Turchi, Margaret Mitchell, Margot Mieskes, Mária Gósy, Maria Liakata, Maria Pershina, Marie Candito, Marie-Catherine de Marneffe, Mariët Theune, Marine Carpuat, Mariona Taulé, Marjorie McShane, Marketa Lopatkova, Mark Fishel, Mark Granroth-Wilding, Mark-Jan Nederhof, Mark Liberman, Mark Sammons, Mark Steedman, Markus Dreyer, Marta R. Costa-jussà, Mary Ellen Foster, Masato Hagiwara, Massimo Poesio, Matthew Purver, Matthew Stone, Matthias Gallé, Matthias Huck, Maud Ehrmann, Md. Faisal Mahbub Chowdhury, Mengqiu Wang, Meng Zhang, Menno van Zaanen, Messina Enza, Michael Carl, Michael Elhadad, Michael Gamon, Michael Johnston, Michael Paul, Michael Picheny, Micha Elsner, Michael Strube, Michael White, Michael Wiegand, Michel Galley, Miguel Ballesteros, Mihael Arcan, Mikael Kågebäck, Mike Dillinger, Mikel Forcada, Mikel Iruskieta, Mike Thelwall, Miles Osborne, Milica Gasic, Ming Zhou, Minlie Huang, Min Zhang, Miriam Butt, Mohamed Maamouri, Mohamed Yahya, Mohit Bansal, Mona Diab, Monojit Choudhury, Mu Li, Muntsa Padró, Muyun Yang, Myungha Jang, Nadir Durrani, Nancy Ide, Naoaki Okazaki, Nathaniel Smith, Nathan Schneider, Ndapandula Nakashole, Nianwen Xue, Nikolaos Lagos, Niladri Chatterjee, Nitish Aggarwal, Nobuhiro Kaji, Noémie Elhadad, Nuria Gala, Oliver Ferschke, Oliver Niebuhr, Olivier Pietquin, Omri Abend, Ondrej Bojar, Oscar Täckström, Owen Rambow, Özlem Çetinoğlu, Paavo Alku, Pablo Duboue, Pablo Gervás, Pallika Kanani, Paola Merlo, Pascal Denis, Patrick Ehlen, Patrick Nguyen, Patrick Paroubek, Patrick Ruch, Patrik Lambert, Paul Buitelaar, Paul McNamee, Paul Meurer, Paul Piwek, Paul Soma, Paul Thompson, Pavel Braslavski, Peng Xu, Peter Ljunglöf, Peter Turney, Petya Osenova, Phil Blunsom, Philipp Cimiano, Philippe Muller, Philipp Koehn, Philip Wille, Phillippe Langlais, Piek Vossen, Pierpaolo Basile, Pierre Zweigenbaum, Prasanth Kolachina, Preslav Nakov, Pushpak Bhattacharyya, Qiang Zhou, Qin Gao, Qi Zhang, Qun Liu, Radu Florian, Raffaella Bernardi, Raghavendra Udupa, Rahul Jha, Ralph Grishman, Rami Al-Rfou, Raquel Fernandez, Raquel Hervas, Raquel Justo, Rashmi Prasad, Ravi Sinha, Reed Coke, Reid Swanson, Richard Johansson, Richard Socher, Richard Sutcliffe, Richard Tzong-Han Tsai, Richard Zens, Riyaz Bhatt, Roberto Basili, Roberto Zamparelli, Rodney Nielsen, Roi Reichart, Roland Kuhn, Roman Yangarber, Ronaldo Martins, Ron Artstein, Ronen Feldman, Roser Saurí, Roy Bar-Haim, Rui Xia, Ruli Manurung, Rune Sætre, Rutu Mulkar, Ryuichiro Higashinaka, Sabine Bergler, Sabine Schulte im Walde, Sadao Kurohashi, Saif Mohammad, Salah Ait-Mokhtar, Sameer Pradhan, Sameer Singh, Sameh Alansary, Sanda Harabagiu, Sandra Kübler, Sandra Williams, Sara Stymne, Saša Hasan, Sasha Blair-Goldensohn, Satoshi Sekine, Saurabh Kataria, Scott Martin, Sebastian Padó, Secondary Reviewer, Seong-Bae Park, Sergei Nirenburg, Seth Kulick, Seung-won Hwang, Shachar Mirkin, Shafiq Joty, Shalom Lappin, Shankar Kumar, Shan Wang, Shih-Hung Wu, Shikhar Kumar Sarma, Shou-de Lin, Shoushan Li, Shujie Liu, Shu-Kai Hsieh, Shuly Wintner, Siddharth Agrawal, Simone Teufel, Sinno J. Pan,

Slav Petrov, Smaranda Muresan, Sobha Lalitha Devi, Song Feng, Sophia Ananiadou, Srinivas Bangalore, Sriram Venkatapathy, Stefan Bott, Stefan Dumitrescu, Stefan Evert, Stefan L. Frank, Stefano Borgo, Stefano Faralli, Stefan Riezler, Stefan Scherer, Stefan Thater, Stephan Busemann, Stephanie Strassel, Stephan Oepen, Stephen Clark, Stergos Afantenos, Steve DeNeefe, Steven Bird, Steven Piantadosi, Sudeep Gandhe, Sudeshna Sarkar, Sung-Hyon Myaeng, Suresh Manandhar, Susanne Burger, Suzanne Stevenson, Sven Hartrumpf, Svetla Koeva, Svetlana Timoshenko, Swapna Somasundaran, Sylvain Pogodalla, Taesun Moon, Takenobu Tokunaga, Takuya Matsuzaki, Tamara Polajnar, Tara McIntosh, Taro Watanabe, Tatsuya Kawahara, Taylor Berg-Kirkpatrick, Tejaswini Deoskar, Teresa Herrmann, Terry Koo, Teruhisa Misu, Thierry Poibeau, Thomas Mueller, Tieyun Qian, Timothy Baldwin, Tim Van de Cruys, Ting Liu, Tom Kwiatkowski, Tommaso Caselli, Tomohide Shibata, Tomoyuki Kajiwara, Tong Xiao, Toni Marti, Tony Veale, Toshiaki Nakazawa, Toshiyuki Sadanobu, Tracy Holloway King, U Kang, Ulrich Germann, Ulrike Pado, Vahed Qazvinian, Valentin Spitkovsky, Valeria de Paiva, Valia Kordoni, Vanni Zavarella, Vasudeva Varma, Verena Rieser, Verginica Mititelu, Verónica Pérez-Rosas, Veronique Hoste, Veselin Stoyanov, Victor Raskin, Vincent Ng, Virach Sornlertlamvanich, Vivek Kumar Rangarajan Sridhar, Viviana Mascardi, Vladimir Selegey, Wai Lam, Waleed Ammar, Wanxiang Che, Wayne Xin Zhao, Weiwei Sun, Wei Xu, Wei Zhang, Wenbin Jiang, Wen-Chih Peng, Wenjie Li, Wenliang Chen, Wen-Lian Hsu, Wenxuan Gao, William Jarrold, William Schuler, Wim Peters, Wlodek Zadrozny, Wolfgang Macherey, Wolfgang Maier, Wolf-Tilo Balke, Wray Buntine, Xavier Blanco Escoda, Xavier Carreras, Xavier Tannier, Xiaodan Zhu, Xiaodong He, Xiaodong Shi, Xiaoguang Hu, Xiaojun Wan, Xiaoqiang Luo, Xiaoyan Zhu, Xing Wang, Xin Wang, Xinyan Xiao, Xipeng Qiu, Xuanjing Huang, Xu Jian, Yael Netzer, Yajuan Lv, Yang Ding, Yanghua Xiao, Yang Liu, Yangqiu Song, Yanjun Ma, Yannick Versley, Yashar Mehdad, Yejin Choi, Yisong Yue, Yoav Artzi, Yoav Goldberg, Yoshinori Sagisaka, Young-In Song, Yue Lu, Yuhong Guo, Yujie Zhang, Yuji Matsumoto, Yukiko Nakano, Yulan He, Yunqing Xia, Yunyao Li, Yuval Marton, Yu Zhou, Zdenka Uresova, Željko Agić, Zhenghua Li, Zhiyuan Cai, Zhiyuan Chen, Zhongjun He, Zhongqiang Huang, Ziheng Lin, and Zornitsa Kozareva

Invited Speakers

Mary Harper — *Learning from 26 languages: Program Management and Science in the Babel Program*

Mary will give her invited talk on Monday August 25th.

Ted Gibson — *Language for Communication: Language as Rational Inference*

Ted will give his invited talk on Tuesday August 26th.

Qun Liu — *Annotation Adaptation and Language Adaptation in NLP*

Qun will give his invited talk on Thursday August 28th.

Martin Kay — *Does a Computational Linguist have to be a Linguist?*

Martin will give his invited talk on Friday August 29th.

After Dinner Speaker

Tony Veale — *Creative Twitterbots: Putting Words (and Wit) Into the Mouths of Bots*

Tony will give his talk after the conference dinner on Wednesday August 27th.

Table of Contents

Conference Program

Monday, August 25, 2014

08:45-09:00 Welcome and Opening by General, Progam and Local Chairs

Session Mo11: (09:00-10:15) Invited Talk 1

09:00–10:15 *Learning from 26 Languages: Program Management and Science in the Babel Program*
Mary Harper

10:15-10:45 Coffee Break

Session Mo21: (10:45-12:25) Modeling of Discourse and Dialogue I

10:45–11:10 *Unsupervised learning of rhetorical structure with un-topic models*
Diarmuid Ó Séaghdha and Simone Teufel

11:10–11:35 *Cross-lingual Coreference Resolution of Pronouns*
Michal Novak and Zdenek Zabokrtsky

11:35–12:00 *A Context-Aware NLP Approach For Noteworthiness Detection in Cellphone Conversations*
Francesca Bonin, Jose San Pedro and Nuria Oliver

12:00–12:25 *Hierarchical Topical Segmentation with Affinity Propagation*
Anna Kazantseva and Stan Szpakowicz

Session Mo22: (10:45-12:25) Sentiment Analysis, Opinion Mining and Social Media I

10:45–11:10 *Capturing Cultural Differences in Expressions of Intentions*
Marc Tomlinson, David Bracewell and Wayne Krug

11:10–11:35 *Identification of Implicit Topics in Twitter Data Not Containing Explicit Search Queries*
Suzi Park and Hyopil Shin

11:35–12:00 *Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts*
Cicero dos Santos and Maira Gatti

Monday, August 25, 2014 (continued)

12:00–12:25 *Joint Inference and Disambiguation of Implicit Sentiments via Implicature Constraints*
Lingjia Deng, Janyce Wiebe and Yoonjung Choi

Session Mo23: (10:45-12:25) Information Retrieval and Question Answering

10:45–11:10 *Group Non-negative Matrix Factorization with Natural Categories for Question Retrieval in Community Question Answer Archives*
Guangyou Zhou, Yubo Chen, Daojian Zeng and Jun Zhao

11:10–11:35 *Multi-Objective Search Results Clustering*
Sudipta Acharya, Sriparna Saha, Jose G. Moreno and Gaël Dias

11:35–12:00 *Query-by-Example Image Retrieval using Visual Dependency Representations*
Desmond Elliott, Victor Lavrenko and Frank Keller

12:00–12:25 *Augmenting Business Entities with Salient Terms from Twitter*
Riham Mansour, Nesma Refaei and Vanessa Murdock

Session Mo24: (10:45-12:25) Machine Learning for CL and NLP

10:45–11:10 *A PAC-Bayesian Approach to Minimum Perplexity Language Modeling*
Sujeeth Bharadwaj and Mark Hasegawa-Johnson

11:10–11:35 *Co-learning of Word Representations and Morpheme Representations*
Siyu Qiu, Qing Cui, Jiang Bian, Bin Gao and Tie-Yan Liu

11:35–12:00 *A Probabilistic Model for Learning Multi-Prototype Word Embeddings*
Fei Tian, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen and Tie-Yan Liu

12:00–12:25 *Learning Task-specific Bilexical Embeddings*
Pranava Swaroop Madhyastha, Xavier Carreras and Ariadna Quattoni

12:25-14:00 Lunch Break

Monday, August 25, 2014 (continued)

Session Mo3P: (14:00-15:15) Posters I

Building Large-Scale Twitter-Specific Sentiment Lexicon : A Representation Learning Approach
Duyu Tang, Furu Wei, Bing Qin, Ming Zhou and Ting Liu

Political Tendency Identification in Twitter using Sentiment Analysis Techniques
Ferran Pla and Lluís-F. Hurtado

A Study of using Syntactic and Semantic Structures for Concept Segmentation and Labeling
Iman Saleh, Scott Cyphers, Jim Glass, Shafiq Joty, Lluís Màrquez, Alessandro Moschitti and Preslav Nakov

Time-aware Personalized Hashtag Recommendation on Social Media
Qi Zhang, Yeyun Gong, Xuyang Sun and Xuanjing Huang

Sarcasm Detection on Czech and English Twitter
Tomáš Ptáček, Ivan Habernal and Jun Hong

A Three-Step Transition-Based System for Non-Projective Dependency Parsing
Ophélie Lacroix and Denis Béchet

Collaborative Topic Regression with Multiple Graphs Factorization for Recommendation in Social Media
Qing Zhang and Houfeng Wang

High Performance Word Sense Alignment by Joint Modeling of Sense Distance and Gloss Similarity
Michael Matuschek and Iryna Gurevych

Multi-view Chinese Treebanking
Likun Qiu, Yue Zhang, Peng Jin and Houfeng Wang

Rapid Development of a Corpus with Discourse Annotations using Two-stage Crowdsourcing
Daisuke Kawahara, Yuichiro Machida, Tomohide Shibata, Sadao Kurohashi, Hayato Kobayashi and Manabu Sassano

Chinese Word Ordering Errors Detection and Correction for Non-Native Chinese Language Learners
Shuk-Man Cheng, Chi-Hsin Yu and Hsin-Hsi Chen

Unsupervised Training Set Generation for Automatic Acquisition of Technical Terminology in Patents
Alex Judea, Hinrich Schütze and Soeren Bruegmann

A Data Driven Approach for Person Name Disambiguation in Web Search Results
Agustín D. Delgado, Raquel Martínez, Víctor Fresno and Soto Montalvo

Picking the Amateur's Mind - Predicting Chess Player Strength from Game Annotations
Christian Scheible and Hinrich Schütze

Zipf's Law and Statistical Data on Modern Tibetan
Huidan Liu, Minghua Nuo and Jian Wu

Simple or Complex? Assessing the readability of Basque Texts
Itziar Gonzalez-Dios, María Jesús Aranzabe, Arantza Díaz de Ilarraza and Haritz Salaberri

Influence of Target Reader Background and Text Features on Text Readability in Bangla: A Computational Approach
Manjira Sinha, Tirthankar Dasgupta and Anupam Basu

Inducing Word Sense with Automatically Learned Hidden Concepts
Baobao Chang, Wenzhe Pei and Miaohong Chen

Inferring Knowledge with Word Refinements in a Crowdsourced Lexical-Semantic Network
Manel Zarrouk and Mathieu Lafourcade

A Supervised Learning Approach Towards Profiling the Preservation of Authorial Style in Literary Translations
Gerard Lynch

Author Verification Using Common N-Gram Profiles of Text Documents
Magdalena Jankowska, Evangelos Milios and Vlado Keselj

Dynamically Integrating Cross-Domain Translation Memory into Phrase-Based Machine Translation during Decoding
Kun Wang, Chengqing Zong and Keh-Yih Su

Machine Translation Quality Estimation Across Domains
José G. C. de Souza, Marco Turchi and Matteo Negri

Investigating the Usefulness of Generalized Word Representations in SMT
Nadir Durrani, Philipp Koehn, Helmut Schmid and Alexander Fraser

Confusion Network for Arabic Name Disambiguation and Transliteration in Statistical Machine Translation
Young-Suk Lee

Fourteen Light Tasks for comparing Analogical and Phrase-based Machine Translation
Rafik Rhouma and Phillippe Langlais

Finding Zelig in Text: A Measure for Normalising Linguistic Accommodation
Simon Jones, Rachel Cotterill, Nigel Dewdney, Kate Muir and Adam Joinson

The annotation of the Central Unit in Rhetorical Structure Trees: A Key Step in Annotating Rhetorical Relations
Mikel Iruskieta, Arantza Díaz de Ilarraza and Mikel Lersundi

Measuring Lexical Cohesion: Beyond Word Repetition
Anna Kazantseva and Stan Szpakowicz

Fast Tweet Retrieval with Compact Binary Codes
Weiwei Guo, Wei Liu and Mona Diab

Learning Sense-specific Word Embeddings By Exploiting Bilingual Resources
Jiang Guo, Wanxiang Che, Haifeng Wang and Ting Liu

Using unmarked contexts in nominal lexical semantic classification
Lauren Romeo, Sara Mendes and Núria Bel

Skill Inference with Personal and Skill Connections
Zhongqing Wang, Shoushan Li, Hanxiao Shi and Guodong Zhou

Jointly or Separately: Which is Better for Parsing Heterogeneous Dependencies?
Meishan Zhang, Wanxiang Che, Yanqiu Shao and Ting Liu

An LR-inspired generalized lexicalized phrase structure parser
Benoit Crabbé

Modeling Review Argumentation for Robust Sentiment Analysis
Henning Wachsmuth, Martin Trenkmann, Benno Stein and Gregor Engels

Biber Redux: Reconsidering Dimensions of Variation in American English
Rebecca J. Passonneau, Nancy Ide, Songqiao Su and Jesse Stuart

Cross-lingual Discourse Relation Analysis: A corpus study and a semi-supervised classification system
Junyi Jessy Li, Marine Carpuat and Ani Nenkova

Enforcing Topic Diversity in a Document Recommender for Conversations
Maryam Habibi and Andrei Popescu-Belis

Identifying Important Features for Graph Retrieval
Zhuo Li, Sandra Carberry, Hui Fang and Kathleen McCoy

15:15-15:45 Coffee Break

Session Mo41: (15:45-17:25) Modeling of Discourse and Dialogue II

15:45–16:10 *Inducing Discourse Connectives from Parallel Texts*
Majid Laali and Leila Kosseim

16:10–16:35 *Lyrics-based Analysis and Classification of Music*
Michael Fell and Caroline Sporleder

16:35–17:00 *Interpretation of Chinese Discourse Connectives for Explicit Discourse Relation Recognition*
Hen-Hsen Huang, Tai-Wei Chang, Huan-Yuan Chen and Hsin-Hsi Chen

17:00–17:25 *Unsupervised Coreference Resolution by Utilizing the Most Informative Relations*
Nafise Sadat Moosavi and Michael Strube

Session Mo42: (15:45-17:25) Sentiment Analysis, Opinion Mining and Social Media II

15:45–16:10 *Knowledge Sharing via Social Login: Exploiting Microblogging Service for Warming up Social Question Answering Websites*
Yang Xiao, Wayne Xin Zhao, Kun Wang and Zhen Xiao

16:10–16:35 *Review Topic Discovery with Phrases using the Pólya Urn Model*
Geli Fei, Zhiyuan Chen and Bing Liu

16:35–17:00 *Joint Opinion Relation Detection Using One-Class Deep Neural Network*
Liheng Xu, Kang Liu and Jun Zhao

17:00–17:25 *A Generative Model for Identifying Target Companies of Microblogs*
Yeyun Gong, Yaqian Zhou, Ya Guo, Qi Zhang and Xuanjing Huang

Session Mo43: (15:45-17:25) Semantic Processing, Distributional Semantics and Compositional Semantics I

15:45–16:10 *Inducing Latent Semantic Relations for Structured Distributional Semantics*
Sujay Kumar Jauhar and Eduard Hovy

16:10–16:35 *Improving distributional thesauri by exploring the graph of neighbors*
Vincent Claveau, Ewa Kijak and Olivier Ferret

16:35–17:00 *Towards Syntax-aware Compositional Distributional Semantic Models*
Lorenzo Ferrone and Fabio Massimo Zanzotto

17:00–17:25 *Low-Dimensional Manifold Distributional Semantic Models*
Georgia Athanasopoulou, Elias Iosif and Alexandros Potamianos

Monday, August 25, 2014 (continued)

Session Mo44: (15:45-17:25) Software, Tools

15:45–16:10 *An Entity-Centric Coreference Resolution System for Person Entities with Rich Linguistic Information*
Marcos Garcia and Pablo Gamallo

16:10–16:35 *Unsupervised Multiword Segmentation of Large Corpora using Prediction-Driven Decomposition of n-grams*
Julian Brooke, Vivian Tsang, Graeme Hirst and Fraser Shein

16:35–17:00 *docrep: A lightweight and efficient document representation framework*
Tim Dawborn and James R. Curran

17:00–17:25 *Why Implementation Matters: Evaluation of an Open-source Constraint Grammar Parser*
Dávid Márk Nemeskey, Francis Tyers and Mans Hulden

Tuesday, August 26, 2014

Session Tu11: (09:00-10:15) Invited Talk 2

09:00–10:15 *Language for Communication: Language as Rational Inference*
Edward Gibson

10:15-10:45 Coffee Break

Session Tu21: (10:45-12:25) Syntax, Grammar Induction, Syntactic and Semantic Parsing I

10:45–11:10 *Soft Cross-lingual Syntax Projection for Dependency Parsing*
Zhenghua Li, Min Zhang and Wenliang Chen

11:10–11:35 *Automatic Feature Selection for Agenda-Based Dependency Parsing*
Miguel Ballesteros and Bernd Bohnet

11:35–12:00 *Predicate-Argument Structure Analysis with Zero-Anaphora Resolution for Dialogue Systems*
Kenji Imamura, Ryuichiro Higashinaka and Tomoko Izumi

12:00–12:25 *Feature Embedding for Dependency Parsing*
Wenliang Chen, Yue Zhang and Min Zhang

Tuesday, August 26, 2014 (continued)

Session Tu22: (10:45-12:25) Sentiment Analysis, Opinion Mining and Social Media III

10:45–11:10 *Identifying Emotional and Informational Support in Online Health Communities*
Prakhar Biyani, Cornelia Caragea, Prasenjit Mitra and John Yen

11:10–11:35 *Identifying Emotion Labels from Psychiatric Social Texts Using Independent Component Analysis*
Liang-Chih Yu and Chun-Yuan Ho

11:35–12:00 *Modeling Mutual Influence Between Social Actions and Social Ties*
Xiaofeng Yu and Junqing Xie

12:00–12:25 *Discovering Topical Aspects in Microblogs*
Abhimanyu Das and Anitha Kannan

Session Tu23: (10:45-12:25) Applications I

10:45–11:10 *Utilizing Microblogs for Automatic News Highlights Extraction*
Zhongyu Wei and Wei Gao

11:10–11:35 *A Step Towards Usable Privacy Policy: Automatic Alignment of Privacy Statements*
Fei Liu, Rohan Ramanath, Norman Sadeh and Noah A. Smith

11:35–12:00 *An Off-the-shelf Approach to Authorship Attribution*
Jamal A. Nasir, Nico Görnitz and Ulf Brefeld

12:00–12:25 *Automatic Prediction of Aesthetics and Interestingness of Text Passages*
Debasis Ganguly, Johannes Leveling and Gareth Jones

Tuesday, August 26, 2014 (continued)

Session Tu24: (10:45-12:25) Modeling of Discourse and Dialogue III

10:45–11:10 *Triple based Background Knowledge Ranking for Document Enrichment*
Muyu Zhang, Bing Qin, Ting Liu and Mao Zheng

11:10–11:35 *Towards an open-domain conversational system fully based on natural language processing*
Ryuichiro Higashinaka, Kenji Imamura, Toyomi Meguro, Chiaki Miyazaki, Nozomi Kobayashi, Hiroaki Sugiyama, Toru Hirano, Toshiro Makino and Yoshihiro Matsuo

11:35–12:00 *The Impact of Deep Hierarchical Discourse Structures in the Evaluation of Text Coherence*
Vanessa Wei Feng, Ziheng Lin and Graeme Hirst

12:00–12:25 *Lexical Chaining for Measuring Discourse Coherence Quality in Test-taker Essays*
Swapna Somasundaran, Jill Burstein and Martin Chodorow

12:25-14:00 Lunch Break

Session Tu3P: (14:00-15:15) Posters II

Improving Cloze Test Performance of Language Learners Using Web N-Grams
Martin Potthast, Matthias Hagen, Anna Beyer and Benno Stein

A Framework for Translating SMS Messages
Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore and Ron Shacham

A Hybrid Approach to Features Representation for Fine-grained Arabic Named Entity Recognition
Fahd Alotaibi and Mark Lee

Prior-informed Distant Supervision for Temporal Evidence Classification
Ridho Reinanda and Maarten de Rijke

Identification of Basic Phrases for Kazakh Language using Maximum Entropy Model
Gulila Altenbek, Xiaolong Wang and Gulizhada Haisha

Collecting Bilingual Audio in Remote Indigenous Communities
Steven Bird, Lauren Gawne, Katie Gelbart and Isaac McAlister

Inclusive yet Selective: Supervised Distributional Hypernymy Detection
Stephen Roller, Katrin Erk and Gemma Boleda

Automatic Discovery of Adposition Typology
Rishiraj Saha Roy, Rahul Katare, Niloy Ganguly and Monojit Choudhury

What good are 'Nominalkomposita' for 'noun compounds': Multilingual Extraction and Structure Analysis of Nominal Compositions using Linguistic Restrictors
Patrick Ziering and Lonneke van der Plas

Automatic Classification of Communicative Functions of Definiteness
Archna Bhatia, Chu-Cheng Lin, Nathan Schneider, Yulia Tsvetkov, Fatima Talib Al-Raisi, Laleh Roostapour, Jordan Bender, Abhimanu Kumar, Lori Levin, Mandy Simons and Chris Dyer

Argument structure of adverbial derivatives in Russian
Igor Boguslavsky

Active Learning in Noisy Conditions for Spoken Language Understanding
Hossein Hadian and Hossein Sameti

A Self-adaptive Classifier for Efficient Text-stream Processing
Naoki Yoshinaga and Masaru Kitsuregawa

A Dependency Edge-based Transfer Model for Statistical Machine Translation
Hongshen Chen, Jun Xie, Fandong Meng, Wenbin Jiang and Qun Liu

Fast Domain Adaptation of SMT models without in-Domain Parallel Data
Prashant Mathur, Sriram Venkatapathy and Nicola Cancedda

Discriminative Language Models as a Tool for Machine Translation Error Analysis
Koichi Akabe, Graham Neubig, Sakriani Sakti, Tomoki Toda and Satoshi Nakamura

A Structured Language Model for Incremental Tree-to-String Translation
Heng Yu, Haitao Mi, Liang Huang and Qun Liu

A Lexicalized Reordering Model for Hierarchical Phrase-based Translation
Hailong Cao, Dongdong Zhang, Mu Li, Ming Zhou and Tiejun Zhao

Automatic Corpus Expansion for Chinese Word Segmentation by Exploiting the Redundancy of Web Information
Xipeng Qiu, ChaoChao Huang and Xuanjing Huang

Fast High-Accuracy Part-of-Speech Tagging by Independent Classifiers
Robert Moore

Morfessor FlatCat: An HMM-Based Method for Unsupervised and Semi-Supervised Learning of Morphology
Stig-Arne Grönroos, Sami Virpioja, Peter Smit and Mikko Kurimo

Japanese Word Reordering Integrated with Dependency Parsing
Kazushi Yoshida, Tomohiro Ohno, Yoshihide Kato and Shigeki Matsubara

Query-focused Multi-Document Summarization: Combining a Topic Model with Graph-based Semi-supervised Learning
Yanran Li and Sujian Li

Ranking Multidocument Event Descriptions for Building Thematic Timelines
Kiem-Hieu Nguyen, Xavier Tannier and Véronique Moriceau

Integrating Language and Vision to Generate Natural Language Descriptions of Videos in the Wild
Jesse Thomason, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko and Raymond Mooney

Cross-Topic Authorship Attribution: Will Out-Of-Topic Data Help?
Upendra Sapkota, Thamar Solorio, Manuel Montes, Steven Bethard and Paolo Rosso

Online Gaming for Crowd-sourcing Phrase-equivalents
A Kumaran, Melissa Densmore and Shaishav Kumar

Unsupervised Verb Inference from Nouns Crossing Root Boundary
Soon Gill Hong, Sin-hee Cho and Mun Yong Yi

Enriching Wikipedia's Intra-language Links by their Cross-language Transfer
Takashi Tsunakawa, Makoto Araya and Hiroyuki Kaji

Chinese Irony Corpus Construction and Ironic Structure Analysis
Yi-jie Tang and Hsin-Hsi Chen

Global Methods for Cross-lingual Semantic Role and Predicate Labelling
Lonneke van der Plas, Marianna Apidianaki and Chenhua Chen

Multilingual Semantic Parsing : Parsing Multiple Languages into Semantic Representations
Zhanming Jie and Wei Lu

Unsupervised Word Sense Induction using Distributional Statistics
Kartik Goyal and Eduard Hovy

Group based Self Training for E-Commerce Product Record Linkage
Xin Zhao, Yuexin Wu, Hongfei Yan and Xiaoming Li

Reducing Over-Weighting in Supervised Term Weighting for Sentiment Analysis
Haibing Wu and Xiaodong Gu

Sentiment Classification with Graph Co-Regularization
Guangyou Zhou, Jun Zhao and Daojian Zeng

Hybrid Deep Belief Networks for Semi-supervised Sentiment Classification
Shusen Zhou, Qingcai Chen, Xiaolong Wang and Xiaoling Li

Latent Dynamic Model with Category Transition Constraint for Opinion Classification
Takeshi Kobayakawa

Sentence Compression for Target-Polarity Word Collocation Extraction
Yanyan Zhao, Wanxiang Che, Honglei Guo, Bing Qin, Zhong Su and Ting Liu

15:15-15:45 Coffee Break

Tuesday, August 26, 2014 (continued)

Session Tu41: (15:45-17:25) Syntax, Grammar Induction, Syntactic and Semantic Parsing II

15:45–16:10 *Hybrid Grammars for Discontinuous Parsing*
Mark-Jan Nederhof and Heiko Vogler

16:10–16:35 *From neighborhood to parenthood: the advantages of dependency representation over bigrams in Brown clustering*
Simon Suster and Gertjan van Noord

16:35–17:00 *An Empirical Evaluation of Automatic Conversion from Constituency to Dependency in Hungarian*
Katalin Ilona Simkó, Veronika Vincze, Zsolt Szántó and Richárd Farkas

17:00–17:25 *Deep-Syntactic Parsing*
Miguel Ballesteros, Bernd Bohnet, Simon Mille and Leo Wanner

Session Tu42: (15:45-17:25) Semantic Processing, Distributional Semantics and Compositional Semantics II

15:45–16:10 *Modeling Newswire Events using Neural Networks for Anomaly Detection*
Pradeep Dasigi and Eduard Hovy

16:10–16:35 *Million-scale Derivation of Semantic Relations from a Manually Constructed Predicate Taxonomy*
Motoki Sano, Kentaro Torisawa, Julien Kloetzer, Chikara Hashimoto, István Varga and Jong-Hoon Oh

16:35–17:00 *Combining Supervised and Unsupervised Parsing for Distributional Similarity*
Martin Riedl, Irina Alles and Chris Biemann

17:00–17:25 *A Markovian approach to distributional semantics with application to semantic compositionality*
Edouard Grave, Guillaume Obozinski and Francis Bach

Tuesday, August 26, 2014 (continued)

Session Tu43: (15:45-17:25) Applications II

15:45–16:10 *A Beam-Search Decoder for Disfluency Detection*
Xuancong Wang, Hwee Tou Ng and Khe Chai Sim

16:10–16:35 *Single Document Keyphrase Extraction Using Label Information*
Sumit Negi

16:35–17:00 *Predicting Interesting Things in Text*
Michael Gamon, Arjun Mukherjee and Patrick Pantel

17:00–17:25 *Context Dependent Claim Detection*
Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni and Noam Slonim

Session Tu44: (15:45-17:25) Language Resources

15:45–16:10 *Annotating Argument Components and Relations in Persuasive Essays*
Christian Stab and Iryna Gurevych

16:10–16:35 *Building a Hierarchically Aligned Chinese-English Parallel Treebank*
Dun Deng and Nianwen Xue

16:35–17:00 *3arif: A Corpus of Modern Standard and Egyptian Arabic Tweets Annotated for Epistemic Modality Using Interactive Crowdsourcing*
Rania Al-Sabbagh, Roxana Girju and Jana Diesner

17:00–17:25 *Empirical Analysis of Aggregation Methods for Collective Annotation*
Ciyang Qing, Ulle Endriss, Raquel Fernandez and Justin Kruger

Wednesday, August 27, 2014

Full Day Excursions

Thursday, August 28, 2014

Session Th11: (09:00-10:15) Invited Talk 3

09:00–10:15 *Annotation Adaptation and Language Adaptation in NLP*
Qun Liu

10:15-10:45 Coffee Break

Session Th21: (10:45-12:25) IE/Database Linking I

10:45–11:10 *Collective Named Entity Disambiguation using Graph Ranking and Clique Partitioning Approaches*
Ayman Alhelbawy and Robert Gaizauskas

11:10–11:35 *Analysis and Refinement of Temporal Relation Aggregation*
Taylor Cassidy and Heng Ji

11:35–12:00 *The Wisdom of Minority: Unsupervised Slot Filling Validation based on Multi-dimensional Truth-Finding*
Dian Yu, Hongzhao Huang, Taylor Cassidy, Heng Ji, Chi Wang, Shi Zhi, Jiawei Han, Clare Voss and Malik Magdon-Ismail

12:00–12:25 *Common Space Embedding of Primal-Dual Relation Semantic Spaces*
Hidekazu Oiwa and Jun'ichi Tsujii

Thursday, August 28, 2014 (continued)

Session Th22: (10:45-12:25) Lexical Semantics and Ontologies I

10:45–11:10 *An Enhanced Lesk Word Sense Disambiguation Algorithm through a Distributional Semantic Model*
Pierpaolo Basile, Annalina Caputo and Giovanni Semeraro

11:10–11:35 *Word Sense Induction Using Lexical Chain based Hypergraph Model*
Tao Qian, Donghong JI, Mingyao Zhang, Chong Teng and Congling Xia

11:35–12:00 *Minimally Supervised Classification to Semantic Categories using Automatically Acquired Symmetric Patterns*
Roy Schwartz, Roi Reichart and Ari Rappoport

12:00–12:25 *Novel Word-sense Identification*
Paul Cook, Jey Han Lau, Diana McCarthy and Timothy Baldwin

Session Th23: (10:45-12:25) Natural Language Generation and Summarization I

10:45–11:10 *Learning to Summarise Related Sentences*
Emmanouil Tzouridis, Jamal Nasir and Ulf Brefeld

11:10–11:35 *Learning to Generate Coherent Summary with Discriminative Hidden Semi-Markov Model*
Hitoshi Nishikawa, Kazuho Arita, Katsumi Tanaka, Tsutomu Hirao, Toshiro Makino and Yoshihiro Matsuo

11:35–12:00 *Query-Focused Opinion Summarization for User-Generated Content*
Lu Wang, Hema Raghavan, Claire Cardie and Vittorio Castelli

12:00–12:25 *Generating Supplementary Travel Guides from Social Media*
Liu Yang, Jing Jiang, Lifu Huang, Minghui Qiu and Lizi Liao

Thursday, August 28, 2014 (continued)

Session Th24: (10:45-12:25) Modeling of Discourse and Dialogue IV and Multimodal Processing

10:45–11:10 *Ensemble-Based Medical Relation Classification*
Jennifer D'Souza and Vincent Ng

11:10–11:35 *Combining Natural and Artificial Examples to Improve Implicit Discourse Relation Identification*
Chloé Braud and Pascal Denis

11:35–12:00 *Reinforcement Learning of Cooperative Persuasive Dialogue Policies using Framing*
Takuya Hiraoka, Graham Neubig, Sakriani Sakti, Tomoki Toda and Satoshi Nakamura

12:00–12:25 *Towards multimodal modeling of physicians' diagnostic confidence and self-awareness using medical narratives*
Joseph Bullard, Cecilia Ovesdotter Alm, Qi Yu, Pengcheng Shi and Anne Haake

12:25-14:00 Lunch Break

Session Th31: (14:00-15:15) Semantic Processing, Distributional Semantics and Compositional Semantics III

14:00–14:25 *Towards Semantic Validation of a Derivational Lexicon*
Britta Zeller, Sebastian Padó and Jan Šnajder

14:25–14:50 *Detecting Learner Errors in the Choice of Content Words Using Compositional Distributional Semantics*
Ekaterina Kochmar and Ted Briscoe

14:50–15:15 *A Novel Distributional Approach to Multilingual Conceptual Metaphor Recognition*
Michael Mohler, Bryan Rink, David Bracewell and Marc Tomlinson

Thursday, August 28, 2014 (continued)

Session Th32: (14:00-15:15) Morphology, Word Segmentation, Tagging and Chunking I

14:00–14:25 *Part of Speech Tagging for French Social Media Data*
Farhad Nooralahzadeh, Caroline Brun and Claude Roux

14:25–14:50 *Morphological Analysis for Japanese Noisy Text based on Character-level and Word-level Normalization*
Itsumi Saito, Kugatsu Sadamitsu, Hisako Asano and Yoshihiro Matsuo

14:50–15:15 *Adapting taggers to Twitter with not-so-distant supervision*
Barbara Plank, Dirk Hovy, Ryan McDonald and Anders Søgaard

Session Th33: (14:00-15:15) Speech Recognition, Text-To-Speech, Spoken Language Understanding

14:00–14:25 *Interpolated Dirichlet Class Language Model for Speech Recognition Incorporating Long-distance N-grams*
Md. Akmal Haidar and Douglas O'Shaughnessy

14:25–14:50 *Situated Incremental Natural Language Understanding using a Multimodal, Linguistically-driven Update Model*
Casey Kennington, Spyros Kousidis and David Schlangen

14:50–15:15 *Quality Estimation for Automatic Speech Recognition*
Matteo Negri, Marco Turchi, José G. C. de Souza and Falavigna Daniele

Session Th34: (14:00-15:15) Lesser Resourced Languages

14:00–14:25 *A Generic Anaphora Resolution Engine for Indian Languages*
Sobha Lalitha Devi, Vijay Sundar Ram and Pattabhi RK Rao

14:25–14:50 *Converting Phrase Structures to Dependency Structures in Sanskrit*
Pawan Goyal and Amba Kulkarni

14:50–15:15 *Uncertainty Detection in Hungarian Texts*
Veronika Vincze

15:15-15:45 Coffee Break

Thursday, August 28, 2014 (continued)

Session Th41: (15:45-17:25) Syntax, Grammar Induction, Syntactic and Semantic Parsing III

15:45–16:10 *Rediscovering Annotation Projection for Cross-Lingual Parser Induction*
Jörg Tiedemann

16:10–16:35 *Synchronous Constituent Context Model for Inducing Bilingual Synchronous Structures*
Xiangyu Duan, Min Zhang and Qiaoming Zhu

16:35–17:00 *Syntactic Parsing and Compound Recognition via Dual Decomposition: Application to French*
Joseph Le Roux, Antoine Rozenknop and Matthieu Constant

17:00–17:25 *Learning the Taxonomy of Function Words for Parsing*
Dongchen Li, Xiantao Zhang, Dingsheng Luo and Xihong Wu

Session Th42: (15:45-17:25) Machine Translation I

15:45–16:10 *A Neural Reordering Model for Phrase-based Translation*
Peng Li, Yang Liu, Maosong Sun, Tatsuya Izuha and Dakun Zhang

16:10–16:35 *Recurrent Neural Network-based Tuple Sequence Model for Machine Translation*
Youzheng Wu, Taro Watanabe and Chiori Hori

16:35–17:00 *Class-Based Language Modeling for Translating into Morphologically Rich Languages*
Arianna Bisazza and Christof Monz

17:00–17:25 *Latent Domain Translation Models in Mix-of-Domains Haystack*
Cuong Hoang and Khalil Sima'an

Thursday, August 28, 2014 (continued)

Session Th43: (15:45-17:25) Linguistic and Cognitive Issues in CL and NLP I

15:45–16:10 *Language Family Relationship Preserved in Non-native English*
Ryo Nagata

16:10–16:35 *Why Gender and Age Prediction from Tweets is Hard: Lessons from a Crowdsourcing Experiment*
Dong Nguyen, Dolf Trieschnigg, A. Seza Doğruöz, Rilana Gravel, Mariet Theune, Theo Meder and Franciska De Jong

16:35–17:00 *Exploring Syntactic Features for Native Language Identification: A Variationist Perspective on Feature Encoding and Ensemble Optimization*
Serhiy Bykh and Detmar Meurers

17:00–17:25 *Applying automatically parsed corpora to the study of language variation*
Jelke Bloem, Arjen Versloot and Fred Weerman

Session Th44: (15:45-17:25) Natural Language Generation and Summarization II and Paraphrasing

15:45–16:10 *Empirical analysis of exploiting review helpfulness for extractive summarization of online reviews*
Wenting Xiong and Diane Litman

16:10–16:35 *Lexico-syntactic text simplification and compression with typed dependencies*
Mandya Angrosh, Tadashi Nomoto and Advaith Siddharthan

16:35–17:00 *Learning when to point: A data-driven approach*
Albert Gatt and Patrizia Paggio

17:00–17:25 *Generating Acrostics via Paraphrasing and Heuristic Search*
Benno Stein, Matthias Hagen and Christof Bräutigam

Friday, August 29, 2014

Session Fr11: (09:00-10:15) Invited Talk 4

09:00–10:15 *Does a Computational Linguist have to be a Linguist?*
Martin Kay

10:15-10:45 Coffee Break

Session Fr21: (10:45-12:25) Machine Translation II

10:45–11:10 *Query Lattice for Translation Retrieval*
Meiping Dong, Yong Cheng, Yang Liu, Jia Xu, Maosong Sun, Tatsuya Izuha and Jie Hao

11:10–11:35 *RED: A Reference Dependency Based MT Evaluation Metric*
Hui Yu, Xiaofeng Wu, Jun Xie, Wenbin Jiang, Qun Liu and Shouxun Lin

11:35–12:00 *Quality Estimation of English-French Machine Translation: A Detailed Study of the Role of Syntax*
Rasoul Kaljahi, Jennifer Foster, Johann Roturier and Raphael Rubino

12:00–12:25 *Effective Incorporation of Source Syntax into Hierarchical Phrase-based Translation*
Tong Xiao, Adrià de Gispert, Jingbo Zhu and Bill Byrne

Session Fr22: (10:45-12:25) IE/Database Linking II

10:45–11:10 *BEL: Bagging for Entity Linking*
Zhe Zuo, Gjergji Kasneci, Toni Gruetze and Felix Naumann

11:10–11:35 *Exploratory Relation Extraction in Large Text Corpora*
Alan Akbik, Thilo Michael and Christoph Boden

11:35–12:00 *An Analysis of Causality between Events and its Relation to Temporal Information*
Paramita Mirza and Sara Tonelli

12:00–12:25 *Exploring Fine-grained Entity Type Constraints for Distantly Supervised Relation Extraction*
Yang Liu, Kang Liu, Liheng Xu and Jun Zhao

Friday, August 29, 2014 (continued)

Session Fr23: (10:45-12:25) Linguistic and Cognitive Issues in CL and NLP II

10:45–11:10 *Using Collections of Human Language Intuitions to Measure Corpus Representativeness*
Reinhard Rapp

11:10–11:35 *Limited memory incremental coreference resolution*
Kellie Webster and James R. Curran

11:35–12:00 *Left-corner Transitions on Dependency Parsing*
Hiroshi Noji and Yusuke Miyao

12:00–12:25 *Data-driven Measurement of Child Language Development with Simple Syntactic Templates*
Shannon Lubetich and Kenji Sagae

Session Fr24: (10:45-12:25) Lexical Semantics and Ontologies II

10:45–11:10 *Employing Event Inference to Improve Semi-Supervised Chinese Event Extraction*
Peifeng Li, Qiaoming Zhu and Guodong Zhou

11:10–11:35 *Supervised Ranking of Co-occurrence Profiles for Acquisition of Continuous Lexical Attributes*
Julian Brooke and Graeme Hirst

11:35–12:00 *Unsupervised extraction of semantic relations using discourse cues*
Juliette Conrath, Stergos Afantenos, Nicholas Asher and Philippe Muller

12:00–12:25 *HARPY: Hypernyms and Alignment of Relational Paraphrases*
Adam Grycner and Gerhard Weikum

12:25-14:00 Lunch Break

Friday, August 29, 2014 (continued)

Session Fr31: (14:00-15:15) Machine Translation III

14:00–14:25 *Limitations of MT Quality Estimation Supervised Systems: The Tails Prediction Problem*
Erwan Moreau and Carl Vogel

14:25–14:50 *Augment Dependency-to-String Translation with Fixed and Floating Structures*
Jun Xie, Jinan Xu and Qun Liu

14:50–15:15 *Soft Dependency Matching for Hierarchical Phrase-based Machine Translation*
Hailong Cao, Dongdong Zhang, Ming Zhou and Tiejun Zhao

Session Fr32: (14:00-15:15) Lexical Semantics and Ontologies III

14:00–14:25 *Using Spreading Activation to Evaluate and Improve Ontologies*
Ronan Mac an tSaoir

14:25–14:50 *Learning to Distinguish Hypernyms and Co-Hyponyms*
Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir and Bill Keller

14:50–15:15 *"One Entity per Discourse" and "One Entity per Collocation" Improve Named-Entity Disambiguation*
Ander Barrena, Eneko Agirre, Bernardo Cabaleiro, Anselmo Peñas and Aitor Soroa

Session Fr33: (14:00-15:15) IE/Database Linking III

14:00–14:25 *Comparable Study of Event Extraction in Newswire and Biomedical Domains*
Makoto Miwa, Paul Thompson, Ioannis Korkontzelos and Sophia Ananiadou

14:25–14:50 *A Probabilistic Co-Bootstrapping Method for Entity Set Expansion*
Bei Shi, Zhenzhong Zhang, Le Sun and Xianpei Han

14:50–15:15 *Separating Brands from Types: an Investigation of Different Features for the Food Domain*
Michael Wiegand and Dietrich Klakow

Friday, August 29, 2014 (continued)

Session Fr34: (14:00-15:15) Morphology, Word Segmentation, Tagging and Chunking II

14:00–14:25 *Unsupervised Instance-Based Part of Speech Induction Using Probable Substitutes*
Deniz Yuret, Mehmet Ali Yatbaz and Enis Sert

14:25–14:50 *Solving Substitution Ciphers with Combined Language Models*
Bradley Hauer, Ryan Hayward and Grzegorz Kondrak

14:50–15:15 *Unsupervised Word Segmentation in Context*
Gabriel Synnaeve, Isabelle Dautriche, Benjamin Börschinger, Mark Johnson and Emmanuel Dupoux

15:15-15:45 Coffee Break

Session Fr41: (15:45-17:25) Best Paper Talks and Closing

15:45–16:15 *Relation Classification via Convolutional Deep Neural Network*
Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou and Jun Zhao

16:15–16:45 *A context-based model for Sentiment Analysis in Twitter*
Andrea Vanzo, Danilo Croce and Roberto Basili

16:45-17:25 Closing Ceremony

Learning from 26 Languages: Program Management and Science in the Babel Program

Mary Harper
Incisive Analysis Office
Intelligence Advanced Research Projects Activity
Office of the Director of National Intelligence
USA
mary.harper@iarpa.gov

Invited Speaker Abstract

This presentation will illustrate how program management and science can cooperate to increase our understanding of human languages and algorithms for processing them. In this presentation, I will use the IARPA Babel program as an example. The goal of the Babel Program is to rapidly develop speech recognition capability for keyword search in new languages, working with speech recorded in a variety of conditions and with limited amounts of transcription. The speech data is recorded in native countries and contains variability in speaker demographics and recording conditions. The Program will ultimately address a broad set of languages with a variety of phonotactic, phonological, tonal, morphological, and syntactic characteristics. I will discuss the data resources collected to support the research, the challenges that performers have faced when working with a variety of languages collected in realistic environments, the lessons learned, and future directions.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, page 1, Dublin, Ireland, August 23-29 2014.

Unsupervised learning of rhetorical structure with un-topic models

Diarmuid Ó Séaghdha
Computer Laboratory
University of Cambridge
Cambridge, UK
do242@cam.ac.uk

Simone Teufel
Computer Laboratory
University of Cambridge
Cambridge, UK
sht25@cam.ac.uk

Abstract

In this paper we investigate whether unsupervised models can be used to induce conventional aspects of rhetorical language in scientific writing. We rely on the intuition that the rhetorical language used in a document is general in nature and independent of the document's topic. We describe a Bayesian latent-variable model that implements this intuition. In two empirical evaluations based on the task of argumentative zoning (AZ), we demonstrate that our generality hypothesis is crucial for distinguishing between rhetorical and topical language and that features provided by our unsupervised model trained on a large corpus can improve the performance of a supervised AZ classifier.

1 Introduction

Scientific writing has many conventions. Some exist at the level of sentence construction, such as a preference for the passive voice or for deverbal nominalisations. Others relate to the high-level organisation of a paper: a typical paper at an NLP conference may be divided into sections covering the introduction, related work, methods, experimental results and conclusion. There are also intermediate levels of convention that use lexical and phrasal items to signal the role played by each part of the text in the argument the authors wish to construct. The theory of *argumentative zoning* (AZ) describes how a scientific article can be analysed in terms of text blocks (or *zones*) that share a rhetorical function (Teufel, 2010). For example: part of the article may consist of background information, another part may describe the aim of the research, other parts may report the authors' own work or compare that work to alternative approaches in the literature. Supervised computational systems can be trained to mark up the AZ structure of a text automatically (see Section 2); the output of such systems has been shown to aid summarisation and human browsing of the scientific literature (Teufel and Moens, 2002; Guo et al., 2011a; Contractor et al., 2012). However, supervised systems require manually annotated training data that must be created anew for each discipline (and language) before they can be deployed, while large quantities of unannotated text are often available. For this reason, there is considerable value in developing unsupervised systems that induce aspects of rhetorical structure from unannotated text.

In this paper we advance a hypothesis about the *generality* of rhetorical language. We propose that the words and linguistic constructs used to express rhetorical function in a scientific paper are independent of the paper's topic. Naturally there will be some variation across research areas and there may be large differences across disciplines, but within a discipline we do not expect that the specific subject of a paper plays a significant role in how the authors construct their argument. For example, the following template could be used to generate an abstract for very many papers in NLP and other fields:

> The problem of _____ has received a lot of attention because of its relevance to _____. CITATION proposed an approach based on the method of _____. In this paper we present a method for _____ that has the following advantages over prior work: _____. We demonstrate the empirical effectiveness of our method by reporting experiments on _____ data, where it outperforms the approach of CITATION by __%.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2–13, Dublin, Ireland, August 23-29 2014.

This leads us to the idea of two-stage "recipes" for scientific papers, whereby the authors start with a framework of boilerplate text that matches the rhetorical argument they wish to make. The authors can then fill in the gaps with the substance of their research contribution.

The two-stage model is of course an idealisation of how scientists construct their papers, but it is useful as an inspiration for a computational model that implements the generality hypothesis. We propose BOILERPLATE-LDA, a generative model that assigns responsibility for generating each word in an abstract to a document-specific topic model or to a rhetorical language model that is not specific to the document. Essentially, we induce argumentative structure from the parts of the text that are not well-explained by the topic model. Hence we describe BOILERPLATE-LDA as an "un-topic model". We evaluate our model in two settings: a clustering evaluation that treats BOILERPLATE-LDA as performing unsupervised argumentative zoning, and a downstream evaluation where the induced structure is not taken as explicitly modelling argumentative zones but is used to provide informative features for a supervised AZ classifier. In both cases, we show that BOILERPLATE-LDA performs well on a very challenging task.

2 Related work

There has been great interest in unsupervised learning among NLP researchers due to the availability of large amounts of unprocessed text through the Web, newswire providers, scientific repositories and other sources in contrast to the onerous requirements of creating task-specific manually annotated data for training supervised analysers. Particularly relevant to our work is the field of topic modelling, where Bayesian latent-variable models are used to induce meaningful generalisations from observations of co-occurrences. Blei et al. (2003) introduced Latent Dirichlet Allocation (LDA) as a model of thematic structure in documents, but subsequent work has adapted the general framework to many different purposes in modelling text as well as other kinds of data. This includes research on modelling aspects of document structure such as topic segmentation, implementing the intuitions that neighbouring blocks of text are coherent in the sense of lexical similarity (Purver et al., 2006; Gruber et al., 2007; Eisenstein and Barzilay, 2008; Du et al., 2013). The model most similar to ours (that we are aware of) is the model of Ritter et al. (2010), which captures dialogue acts and transitions between them in Twitter conversations.

Despite the general popularity of unsupervised approaches, rhetorical analysis has generally been treated as a problem for supervised machine learning. Classification-based approaches to argumentative zoning typically use a sequence classifier such as a maximum-entropy Markov model or conditional random field (Teufel and Moens, 2002; Siddharthan and Teufel, 2007; Hirohata et al., 2008; Guo et al., 2010). Guo et al. (2011b) take a semi-supervised approach based on active learning and self-training.

Two unsupervised approaches in the literature are Varga et al. (2012) and Reichart and Korhonen (2012). Varga et al. use a topic model variant called ZONE-LDA that assigns each sentence a latent variable index or "topic" and assumes that the words in the sentence are generated from a distribution particular to the topic; in this situation each topic is assumed to correspond to a distinct argumentative zone. Such a model will have the effect of clustering sentences that share lexical items. Varga et al. also propose a model they call ZONE-LDA-B, in which some common words are assigned to a "background" distribution that is independent of the sentence category; this model performs worse than ZONE-LDA in their evaluation. Reichart and Korhonen take an approach based on Markov random fields. They construct a graphical model in which sentence vertices are connected by potentials weighted according to adjacency and sentence similarity, as well as hand-defined rules about passivisation and sentence location.

The papers cited in the two preceding paragraphs have focused on rhetorical analysis in scientific writing, yet there are many other textual genres where argumentation is conventionalised. For example, Burstein et al. (2003) identify building blocks analogous to AZ zones in the writing of English language learners and demonstrate that a supervised classification approach can be used to mark up their essays. Also in the educational domain, Madnani et al. (2012) train a supervised classifier to detect the "shell" language that learners use to organise the high-level structure of their compositions; this is quite close to our idea of "templates" or "recipes" for scientific papers. Sauper and Barzilay (2009) and Chen et al. (2009) both present models that learn structural conventions in Wikipedia articles without relying on human annotation. Sauper and Barzilay's model induces the typical section structure of Wikipedia articles

about a specific entity type (e.g., *Actors* or *Diseases*) and retrieves web snippets relevant to each section for a target entity, before performing multidocument summarisation to produce a new entry for posting to Wikipedia. Chen et al. take a Bayesian segmentation approach to implicitly learn the topical section structure of articles and use a generalised Mallows model, a distribution over permutations, to identify a canonical ordering for sections.[1] Other forms of general rhetorical analysis include Rhetorical Structure Theory (Mann and Thompson, 1988; Marcu, 2000), which captures local discourse relations between segments of text; RST provides a layer of analysis that is separate and complementary to more global schemes such as argumentative zoning.

3 Intuitions

The performance of unsupervised learning depends on how intuitions about the task are incorporated in the statistical model. Our approach relies on three main intuitions:

Sentence similarity: All else being equal, we expect that lexically similar sentences will have similar purposes. At the same time, lexical similarity alone is not sufficient to capture shared argumentative function: all sentences in a paper about parsing will be similar to each other, while the introductory sentences of a parsing paper and a machine translation paper may share few similar lexical items.

Adjacency: The theory of argumentative zones suggests that sentences with the same rhetorical function will often be grouped together into blocks. Additionally, we expect that authors will follow general conventions about the order of zones, e.g., starting with background and goal statements and progressing to results and conclusions.

Generality: We expect that the language used to convey rhetorical function is independent of the topical content of the paper.

Sentence similarity can be captured using standard lexical similarity measures or through the clustering effects of a topic model. The adjacency assumption can be implemented using a linear-chain sequence model such as a Hidden Markov Model. The ZONE-LDA approach of Varga et al. (2012) relies on sentence similarity alone. Reichart and Korhonen's (2012) model combines sentence similarity and adjacency. To the best of our knowledge, the generality hypothesis has not previously been investigated. The model we describe in Section 4 incorporates all three intuitions in its structure.

4 Models

The model we propose assumes that each word in a sentence is generated either from an LDA-style topic model or from a distribution associated with the rhetorical category assigned to the sentence. The former captures the subject matter of the document; the latter captures conventional language that is independent of the document's subject matter. The sentence categories are generated from a first-order Markov model. The assignment of responsibility for a word is implemented through a so-called "switching variable", a binary-valued latent variable. This is a commonly used mechanism for interpolating language models (Griffiths et al., 2004; Reisinger and Mooney, 2010; Ahmed and Xing, 2010); in many cases, the goal is to assign common words to a "background" distribution that is not considered an object of interest from a topic modelling perspective. In our case it is this non-topical part of the text that is the object of interest.

The dependencies between variables in our full BOILERPLATE-LDA model are shown by the plate diagram in Figure 1. The corresponding "generative story" is as follows:

[1] It would be interesting to swap in Chen et al.'s generalised Mallows model for the HMM-style ordering model in BOILERPLATE-LDA. The former has the advantage of capturing non-local ordering effects, while the latter has the advantage of not assuming a single canonical ordering.

for topic $t \in \{1 \dots |T|\}$ **do**
  (Draw a distribution over words)
  $\boldsymbol{\Phi}_t \sim Dirichlet(\boldsymbol{\beta})$
end for
for zone $z \in \{1 \dots |Z|\}$ **do**
  (Draw a distribution over words)
  $\boldsymbol{\Psi}_z \sim Dirichlet(\boldsymbol{\gamma})$
  (Draw a transition distribution)
  $\boldsymbol{\Lambda}_z \sim Dirichlet(\boldsymbol{\lambda})$
end for
(Draw the switch distribution)
$\Sigma \sim Beta(\sigma_0, \sigma_1)$
for doc $d \in \{1 \dots |D|\}$ **do**
  (Draw a distribution over topics)
  $\boldsymbol{\theta}_d \sim Dirichlet(\boldsymbol{\alpha})$
  for sentence $s \in Sentences(d)$ **do**
    $z_s \sim Multinomial(\boldsymbol{\Lambda}_{z_{s-1}})$
    for word $i \in Words(s)$ **do**
      (Draw a switch indicator)
      $b_i = Beta(\Sigma)$
      if $b_i = 0$ **then**
        (Draw a word from the zone-word distribution)
        $w_i \sim Multinomial(\boldsymbol{\Psi}_{z_s})$
      else
        (Draw a topic)
        $t_i \sim Multinomial(\boldsymbol{\theta}_d)$
        (Draw a word from the topic-word distribution)
        $w_i \sim Multinomial(\boldsymbol{\Phi}_{t_i})$
      end if
    end for
  end for
end for

We train the model using Gibbs sampling. Due to Dirichlet-multinomial and beta-Bernoulli conjugacy it is relatively straightforward to integrate out the multinomial and Bernoulli distribution parameters $\boldsymbol{\theta}$, $\boldsymbol{\Phi}$, $\boldsymbol{\Psi}$ and Σ and derive update rules for a collapsed Gibbs sampler. Each iteration of the sampler visits each sentence in the corpus in turn, first sampling the sentence label assignment z_s and then sampling for each word in the sentence the switch indicator b_i and (if $b_i = 1$) the topic assignment t_i. The sentence label update is performed using what Gao and Johnson (2008) call a pointwise collapsed Gibbs sampler. Omitting hyperparameters for clarity, the sampling probabilities can be written as

$$P(z_i = z|\boldsymbol{z}^{-i}, \boldsymbol{w}, \boldsymbol{b}) \propto \frac{f_{z_{i-1}\to z} + \kappa_z}{f_{z_{i-1}} + \sum_{z'} \kappa_{z'}} \frac{f_{z\to z_{i+1}} + I(z = z_{i+1}) + \kappa_{z_{i+1}}}{f_z^{-i} + I(z = z_{i+1}) + \sum_{z'} \kappa_{z'}} \prod_{v \in V} \frac{\Gamma(f^{-i}_{zv,b=0} + f_{s_i v,b=0} + \gamma)}{\Gamma(f_z^{-i} + f_{s_i} + \gamma|V|)} \tag{1}$$

where $f_{z->z'}$ is the transition frequency from zone z to zone z', f_z is the number of sentences assigned zone z; $I(z = z_{i+1})$ has value 1 if the two zone assignments are equal and 0 otherwise; V is the vocabulary of word types; $f_{zv,b=0}$ is the number of words of type z that appear in sentences assigned zone z and whose corresponding switch variable has value 0; $f_{s_i v,b=0}$ is the number of words of type v that appear in sentence s_i and whose corresponding switch variable has value 0; the superscript $^{-i}$ indicates that the frequency is calculated over all sentences except s_i. We introduce observed start and end state variables z_s and z_e to handle the boundaries at the beginning and end of each document.

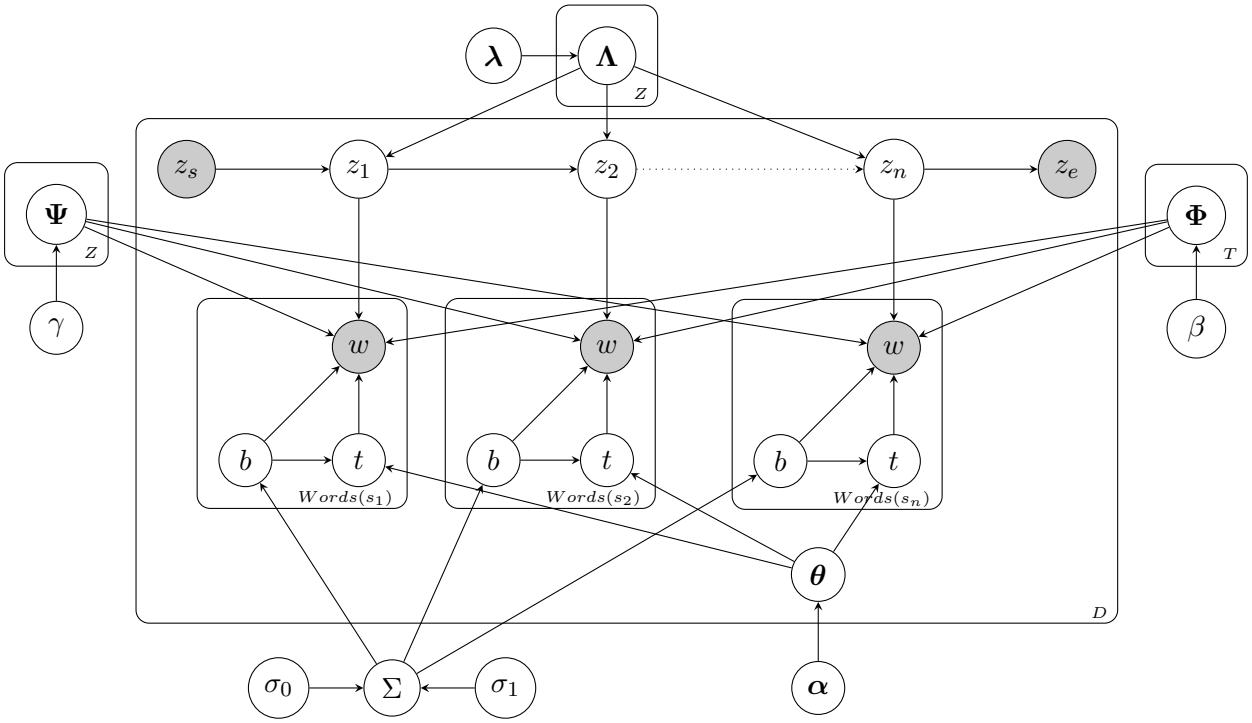


Figure 1: Plate diagram for BOILERPLATE-LDA

The topic and switch variables for each word are sampled in a blocked fashion; the sampling probabilities are similar to the standard LDA updates:

$$
\begin{aligned}
P(b_j = 0, t_j = \emptyset | \mathbf{z}^{-j}, \mathbf{b}^{-j}, \mathbf{t}, \mathbf{w}) &\propto (f^{-j}_{b=0} + \Sigma_0) \frac{f^{-j}_{z_i w_j, b=0} + \gamma}{f_{z_i, b=0} + |V|\gamma} \\
P(b_j = 1, t_j = t | \mathbf{z}^{-j}, \mathbf{b}^{-j}, \mathbf{t}, \mathbf{w}) &\propto (f^{-j}_{b=1} + \Sigma_1) \frac{f^{-j}_{tw_j} + \alpha_z}{f^{-j}_{w_j, b=1} + \sum_{z'} \alpha_{z'}} \frac{f^{-j}_{zw_j} + \beta}{f^{-j}_{z} + |V|\beta} \\
P(b_i = 0, t_i \neq \emptyset | \mathbf{z}^{-j}, \mathbf{b}^{-j}, \mathbf{t}, \mathbf{w}) &= 0 \\
P(b_i = 1, t_i = \emptyset | \mathbf{z}^{-j}, \mathbf{b}^{-j}, \mathbf{t}, \mathbf{w}) &= 0
\end{aligned} \tag{2}
$$

where we use j to index words and i to index sentences; f_{tw_j} is the number of words of type w_j that are assigned topic t; the superscript $^{-j}$ indicates that the frequency is calculated over all words except j.

5 Experiments

5.1 Data

For evaluation, we use a collection of abstracts compiled by Guo et al. (2010). These abstracts had originally been collected in the context of semi-automated cancer risk assessment by searching PubMed for abstracts mentioning one or more of a list of chemicals known to have carcinogenic properties (Korhonen et al., 2009). Guo et al. annotated abstracts for five of these chemicals using an AZ scheme with seven categories: *Background*, *Objective*, *Method*, *Result*, *Conclusion*, *Related work* and *Future work*.[2] In order to test whether our models can also perform over a large, heterogeneous dataset, we also used a collection of 129,595 abstracts taken from a collection of open-access journal articles. Preprocessing involved sentence splitting, tokenisation and part-of-speech tagging using the Stanford CoreNLP toolkit[3] and the removal of all tokens containing non-alphanumeric characters, all tokens of character length one

[2]The annotated dataset has been made available at `http://www.cl.cam.ac.uk/~yg244/abstract_az.html`.
[3]`http://nlp.stanford.edu/software/corenlp.shtml`

and a small set of stop words.[4] This left a training corpus of 16,841,280 tokens.

5.2 Clustering Evaluation

5.2.1 Evaluation

Our first quantitative evaluation investigates whether the zones induced by BOILERPLATE-LDA correspond to the argumentative zones identified by human theorists. We treat this as a clustering task with the gold standard provided by Guo et al.'s (2010) dataset. The clustering evaluation measures we use are the Adjusted Rand Index (Hubert and Arabie, 1985) and Adjusted Mutual Information (Vinh et al., 2010); both measures are normalised to have a maximum value of 1 and are adjusted for chance so that the expected score given to a random clustering is 0. This second property makes them conservative in comparison to other evaluation measures. We report results with the number of zones $|Z| \in \{10, 20, 50\}$ and number of topics $|T| \in \{10, 20, 50, 100\}$; for each combination of settings we report the average evaluation score attained by three independent runs of the learning algorithm.

5.2.2 Models

For our evaluation, we test the following models:

BOILERPLATE-LDA: Our full model, as described in Section 4.

BOILERPLATE-LDA-MULT: A simplified model where the Markov dependencies between zone assignments are replaced by a flat multinomial; the probability of a zone is independent of the adjacent sentences.

BOILERPLATE-LDA-NOTOPICS: A simplified model where all words in a sentence are generated from the zone distribution $\boldsymbol{\Psi}_{z_s}$; this is almost identical to Varga et al.'s (2012) ZONE-LDA model.

K-MEANS: A standard k-means clustering model run until convergence. The features for each sentence consist of tf-idf-transformed lexical frequencies, part-of-speech tags and a location feature computed by dividing the abstract into 5 bins.

The BOILERPLATE-LDA models are all trained for 1000 iterations of Gibbs sampling. The Dirichlet hyperparameters are re-estimated every 10 iterations; the topic hyperparameters $\boldsymbol{\alpha}$ are optimised using a fixed-point iteration to maximise the log-evidence (Minka, 2003; Wallach, 2008), while the other hyperparameters are sampled using Hamiltonian Monte Carlo (Neal, 2010). K-MEANS was run until convergence.

5.2.3 Results

Figure 2 gives an illustration of the zone representation induced at the end of one run of BOILERPLATE-LDA with the settings $|Z| = 10$, $|T| = 100$. Firstly, we list the most probable words for each zone (2a). While the model may not find a perfect match for the gold-standard inventory of argumentative zones, we can see that some induced zones describe standard methodology (8,9), others describe results and implications (1,3,7) and others describe motivations (2,5,6). Inspection of the transition matrix (2b) confirms our expectation that self-transitions have the highest probability; we also observed that the zones most frequently transitioned to from the start state are the motivational zones and the zones most frequently transitioned from to the end state are the results/implications zones. The example abstracts in Figure 3 illustrate how BOILERPLATE-LDA can be used to mark up the text of an abstract as "boilerplate" or "non-boilerplate" based on the values of the switch variables b_i.

[4] The part-of-speech tags are not used by BOILERPLATE-LDA but they are used as features for other models.

1	results, suggest, our, data, study, role, findings, we, between, indicate, important, studies
2	study, we, using, used, investigated, determine, present, between, investigate, analysis, aim
3	increased, significantly, levels, showed, found, observed, significant, after, compared, higher
4	two, sequence, we, found, region, sequences, we, three, identified, between, different, analysis
5	use, more, studies, study, used, however, important, health, most, treatment, clinical, potential
6	role, important, known, studies, however, shown, including, involved, mechanisms, cell
7	case, we, patient, report, rare, most, common, reported, presented, disease, associated, cause
8	CI, significantly, respectively, significant, between, group, mean, higher, compared, more, found
9	study, years, using, two, patients, included, total, group, three, data, after, used, collected, age
10	we, data, analysis, used, using, new, approach, based, method, information, developed, more

(a) Most probable words for each zone

From \ To	Start	1	2	3	4	5	6	7	8	9	10	End
Start	0.00	0.00	0.10	0.01	0.08	0.24	0.36	0.10	0.00	0.03	0.08	0.00
1	0.00	0.37	0.01	0.04	0.01	0.03	0.01	0.00	0.00	0.00	0.01	0.50
2	0.00	0.02	0.26	0.25	0.07	0.01	0.02	0.00	0.05	0.29	0.01	0.00
3	0.00	0.27	0.02	0.59	0.02	0.02	0.02	0.00	0.02	0.00	0.01	0.04
4	0.00	0.12	0.02	0.09	0.62	0.00	0.03	0.00	0.01	0.00	0.05	0.05
5	0.00	0.01	0.10	0.00	0.00	0.55	0.03	0.02	0.00	0.06	0.04	0.18
6	0.00	0.06	0.21	0.05	0.05	0.05	0.50	0.01	0.00	0.01	0.05	0.02
7	0.00	0.02	0.02	0.01	0.01	0.15	0.04	0.63	0.01	0.02	0.00	0.08
8	0.00	0.09	0.01	0.14	0.01	0.07	0.00	0.02	0.61	0.04	0.01	0.01
9	0.00	0.01	0.11	0.05	0.01	0.05	0.00	0.02	0.20	0.54	0.01	0.00
10	0.00	0.05	0.02	0.01	0.07	0.02	0.01	0.00	0.01	0.01	0.68	0.12
End	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

(b) Zone transition probabilities between adjacent sentences

Figure 2: Zones induced by one run of BOILERPLATE-LDA ($|Z| = 10$, $|T| = 100$)

VASP: A Volumetric Analysis of Surface Properties Yields Insights into Protein-Ligand Binding Specificity

Many algorithms that compare protein structures can reveal similarities that suggest related biological functions, even at great evolutionary distances. Proteins with related function often exhibit differences in binding specificity, but few algorithms identify structural variations that effect specificity. To address this problem, we describe the Volumetric Analysis of Surface Properties (VASP), a novel volumetric analysis tool for the comparison of binding sites in aligned protein structures. VASP uses solid volumes to represent protein shape and the shape of surface cavities, clefts and tunnels that are defined with other methods. Our approach, inspired by techniques from constructive solid geometry, enables the isolation of volumetrically conserved and variable regions within three dimensionally superposed volumes. We applied VASP to compute a comparative volumetric analysis of the ligand binding sites formed by members of the steroidogenic acute regulatory protein (StAR)-related lipid transfer (START) domains and the serine proteases. Within both families, VASP isolated individual amino acids that create structural differences between ligand binding cavities that are known to influence differences in binding specificity. Also, VASP isolated cavity subregions that differ between ligand binding cavities which are essential for differences in binding specificity. As such, VASP should prove a valuable tool in the studyof protein-ligand binding specificity.

A new usage of functionalized oligodeoxynucleotide probe for site-specific modification of a guanine base within RNA

Site-specific modification of RNA is of great significance to investigate RNA structure, function and dynamics. Recently, we reported a new method for sequence- and cytosine-selective chemical modification of RNA based on the functional group transfer reaction of the 1-phenyl-2-methylydene-1,3-diketone unit of the 6-thioguanosine base incorporated in the oligodeoxynucleotide probe. In this study, we describe that the functionality transfer rate is greatly enhanced and the selectivity is shifted to the guanine base when the reaction is performed under alkaline conditions. Detailed investigation indicated that the 2-amino group of the enolate form of rG is the reactant of the functionality transfer reaction. As a potential application of this efficient functionality transfer reaction, a pyrene group as a relatively large fluorescent group was successfully transferred to the target guanine base of RNA with a high guanine and site selectivity. This functionality transfer reaction with high efficiency and high site-selectivity would provide a new opportunity as a unique tool for the study of RNA.

Figure 3: Examples of abstracts marked up for boilerplate (underlined) and non-boilerplate (faded text) by BOILERPLATE-LDA

Model	$\|T\|$	$\|Z\| = 10$		$\|Z\| = 20$		$\|Z\| = 50$	
		ARI	NMI	ARI	NMI	ARI	NMI
BOILERPLATE-LDA	10	0.19	0.15	0.09	0.09	0.04	0.07
	20	0.20	0.16	0.03	0.10	0.03	0.08
	50	0.26	0.21	0.18	0.16	0.05	0.10
	100	**0.32**	**0.28**	0.20	0.20	0.07	0.14
BOILERPLATE-LDA-MULT	10	0.13	0.11	0.08	0.08	0.04	0.06
	20	0.10	0.13	0.04	0.09	0.03	0.07
	50	0.21	0.16	0.13	0.14	0.06	0.10
	100	0.18	0.16	0.14	0.14	0.07	0.11
BOILERPLATE-LDA-NOTOPICS	0	0.00	0.02	0.04	0.05	0.06	0.05
K-MEANS	0	0.05	0.05	0.03	0.06	0.03	0.04

Table 1: Results of the clustering evaluation. $|Z|$ is the number of zones; $|T|$ is the number of topics.

The results of the clustering evaluation are presented in Table 1. Clearly, this is a challenging task; the BOILERPLATE-LDA-NOTOPICS and K-MEANS models, which do not filter out topic-specific vocabulary, perform little better than chance in terms of identifying argumentative zones (recall that for the ARI and AMI measures, zero means "not greater than expected by chance" rather than "no correlation at all"). BOILERPLATE-LDA-MULT performs better than those models though not as well as the full BOILERPLATE-LDA model, indicating that sequential structure is important for inducing rhetorical regularities. In general, the best results are attained with low settings of $|Z|$ and high settings of $|T|$; this seems to create the "bottleneck" effect needed to focus the model on purely rhetorical information. The highest scores (ARI = 0.32, AMI = 0.28) are attained by BOILERPLATE-LDA with the settings $|Z| = 10$, $|T| = 100$.

5.3 Supervised Evaluation

5.3.1 Evaluation

A second evaluation of BOILERPLATE-LDA's usefulness is to test whether it can yield features that improve the performance of a supervised argumentative zoning system. It is possible for an unsupervised model to induce structure that does not map exactly onto a pre-existing set of labels but still captures valuable information about the underlying phenomenon that can be of use to a supervised classifier when combined with other information sources. To this end, we train and evaluate supervised models on the same dataset of Guo et al. (2010) that we used for the clustering evaluation. We perform 10-fold cross-validation and report Accuracy (proportion of sentences labelled correctly) as well as macro-averaged Precision, Recall and F-Score. To measure statistical significance we use two-tailed paired t-tests, following Dietterich (1998).[5]

5.3.2 Models

We use two supervised sequence classification algorithms for training models:

LR: A logistic regression classifier with a "history" feature encoding the previous sentence's label, trained with L_1 regularisation, using the implementation in LibLinear.[6]

CRF: A first-order conditional random field classifier, trained with L_1 regularisation, using the implementation in Mallet.[7]

In both cases, the predicted labelling for a test document is given by the most probable (Viterbi) sequence according to the trained model. We use the following feature sets:

[5] In order to address concerns about the suitability of the t-tests under non-normality, we replicated the tests using Wilcoxon's signed-ranks test as recommended by Demšar (2006); the results were identical.

[6] http://www.csie.ntu.edu.tw/~cjlin/liblinear/

[7] http://mallet.cs.umass.edu/

	LR				CRF			
Model	Acc	P	R	F	Acc	P	R	F
BASELINE	0.83	0.71	0.70	0.70	0.85	**0.75**	0.64	0.67
+BOILERPLATE-LDA	**0.84**	**0.72**	**0.71**	**0.71**	**0.86**	0.74	**0.65**	**0.68**
+LDA-BAG (50)	0.83	0.69	0.68	0.68	0.84	0.73	0.62	0.64
+LDA-BAG (100)	0.83	0.69	0.69	0.69	0.84	0.72	0.64	0.66
+LDA-MAX (50)	0.83	0.71	0.69	0.69	0.85	0.72	0.64	0.66
+LDA-MAX (100)	**0.84**	0.71	0.69	0.70	0.85	0.74	0.63	0.66

Table 2: Results of the supervised evaluation

BASELINE: Our baseline set of features is a standard set for supervised argumentative zoning: all unigrams and bigrams in the sentence, all part-of-speech tags in the sentence and a location feature computed by dividing the abstract into 5 bins.

+BOILERPLATE-LDA: The baseline model with additional features corresponding to the zone index assigned by BOILERPLATE-LDA to the sentence. We set $|Z| = 10$, $|T| = 100$ since that setting performed best in the clustering evaluation. As before, we use the output of three independently learned sampling chains, giving each sentence three zone features; the classifier should learn which chains are better than others during training.

+LDA-BAG: The baseline model with additional features derived from standard Latent Dirichlet Allocation models trained on the same corpus as BOILERPLATE-LDA. As LDA assigns a topic to each word in a sentence, we add all topics assigned to all words in the sentence as additional features. As above, we use the output of three sampling chains. We report results for models with 50 topics and 100 topics.

+LDA-MAX: The baseline model with additional features derived from LDA models. Here each model assigns each sentence the single topic assigned to the greatest number of words in the sentence (ties are broken randomly).

5.3.3 Results

Results for the supervised evaluation are presented in Table 2. +BOILERPLATE-LDA is the only augmented feature set that consistently gives an improvement over the baseline features. The improvements in accuracy are statistically significant ($p < 0.01$). In every case but one (which is not statistically significant), the LDA models fail to improve on the baseline in either accuracy or F-Score, showing that the latent structure induced by BOILERPLATE-LDA captures aspects of rhetorical language that are not captured by topical word clustering.

6 Conclusion

We consider the work presented in this paper to be a first step towards the ambitious goal of inducing latent descriptions of the templates used by scientists and writers in other fields. We have shown how our hypothesis about the generality of rhetorical language allows the construction of models that can separate out topical and rhetorical language use. One focus for future work will be to enrich the model structure; an approach based on adaptor grammars (Johnson et al., 2006) could be used to break the reductive unigram assumption in BOILERPLATE-LDA and identify multiword collocations that carry rhetorical information. Another focus will be to broaden our understanding of how unsupervised rhetorical models trained on large corpora can improve the robustness of supervised systems. For example, we have observed that lexicalised AZ classifiers trained on texts from one scientific domain will often perform poorly on texts from another domain; unsupervised models have the potential to induce relevant lexical commonalities across domains.

Acknowledgements

This research is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center (DoI/NBC) contract number D11PC20153. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

Amr Ahmed and Eric P. Xing. 2010. Staying informed: Supervised and semi-supervised multi-view topical analysis of ideological perspective. In *Proceedings of EMNLP-10*, Cambridge, MA.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jill Burstein, Daniel Marcu, and Kevin Knight. 2003. Finding the WRITE stuff: Automatic identification of discourse structure in student essays. *IEEE Intelligent Systems*, 18(1):32–39.

Harr Chen, S.R.K. Branavan, Regina Barzilay, and David R. Karger. 2009. Global models of document structure using latent permutations. In *Proceedings of NAACL-09*, Boulder, CO.

Danish Contractor, Yufan Guo, and Anna Korhonen. 2012. Using argumentative zones for extractive summarization of scientific articles. In *Proceedings of COLING-12*, Mumbai, India.

Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

Thomas G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923.

Lan Du, Wray Buntine, and Mark Johnson. 2013. Topic segmentation with a structured topic model. In *Proceedings of NAACL-13*, Atlanta, GA.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of EMNLP-08*, Honolulu, HI.

Jianfeng Gao and Mark Johnson. 2008. A comparison of Bayesian estimators for unsupervised Hidden Markov Model POS taggers. In *Proceedings of EMNLP-08*, Honolulu, HI.

Thomas L. Griffiths, Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. 2004. Integrating topics and syntax. In *Proceedings of NIPS-04*, Vancouver, BC.

Amit Gruber, Michal Rosen-Zvi, and Yair Weiss. 2007. Hidden topic Markov models. In *Proceedings of AISTATS-07*, San Juan, Puerto Rico.

Yufan Guo, Anna Korhonen, Maria Liakata, Ilona Silins, Lin Sun, and Ulla Stenius. 2010. Identifying the information structure of scientific abstracts: An investigation of three different schemes. In *Proceedings of BioNLP-10*, Uppsala, Sweden.

Yufan Guo, Anna Korhonen, Maria Liakata, Ilona Silins, Johan Högberg, and Ulla Stenius. 2011a. A comparison and user-based evaluation of models of textual information structure in the context of cancer risk assessment. *BMC Bioinformatics*, 12:69.

Yufan Guo, Anna Korhonen, and Thierry Poibeau. 2011b. A weakly-supervised approach to argumentative zoning of scientific documents. In *Proceedings of EMNLP-11*, Edinburgh, UK.

Kenji Hirohata, Naoaki Okazaki, Sophia Ananiadou, and Mitsuru Ishizuka. 2008. Identifying sections in scientific abstracts using conditional random fields. In *Proceedings of IJCNLP-08*, Hyderabad, India.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of Classification*, 2(1):193–218.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2006. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In *Proceedings of NIPS-06*, Vancouver, BC.

Anna Korhonen, Ilona Silins, Lin Sun, and Ulla Stenius. 2009. The first step in the development of text mining technology for cancer risk assessment: identifying and organizing scientific evidence in risk assessment literature. *BMC Bioinformatics*, 10:303.

Nitin Madnani, Michael Heilman, Joel Tetreault, and Martin Chodorow. 2012. Identifying high-level organizational elements in argumentative discourse. In *Proceedings of NAACL-12*, Montreal, QC.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 2000. The rhetorical parsing of unrestricted texts: A surface-based approach. *Computational Linguistics*, 26(3):395–448.

Thomas P. Minka. 2003. Estimating a Dirichlet distribution. Available at `http://research.microsoft.com/en-us/um/people/minka/papers/dirichlet/`.

Radford M. Neal. 2010. MCMC using Hamiltonian dynamics. In Steve Brooks, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng, editors, *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC Press, Boca Raton, FL.

Matthew Purver, Konrad Körding, Tom Griffiths, and Josh Tenenbaum. 2006. Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of COLING-ACL-06*, Sydney, Australia.

Roi Reichart and Anna Korhonen. 2012. Document and corpus level inference for unsupervised and transductive learning of information structure of scientific documents. In *Proceedings of COLING-12*, Mumbai, India.

Joseph Reisinger and Raymond Mooney. 2010. A mixture model with sharing for lexical semantics. In *Proceedings of EMNLP-10*, Cambridge, MA.

Alan Ritter, Colin Cherry, and Bill Dolan. 2010. Unsupervised modeling of Twitter conversations. In *Proceedings of NAACL-HLT-10*, Los Angeles, CA.

Christina Sauper and Regina Barzilay. 2009. Automatically generating Wikipedia articles: A structure-aware approach. In *Proceedings of ACL-IJCNLP-09*, Singapore.

Advaith Siddharthan and Simone Teufel. 2007. Whose idea was this, and why does it matter? Attributing scientific work to citations. In *Proceedings of NAACL-07*, Rochester, NY.

Simone Teufel and Marc Moens. 2002. Summarizing scientific articles: Experiments with relevance and rhetorical status. *Computational Linguistics*, 28(4):409–445.

Simone Teufel. 2010. *The Structure of Scientific Articles: Applications to Citation Indexing and Summarization*. CSLI Publications, Stanford, CA.

Andrea Varga, Daniel Preoţiuc-Pietro, and Fabio Ciravegna. 2012. Unsupervised document zone identification using probabilistic graphical models. In *Proceedings of LREC-12*, Istanbul, Turkey.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11:2837–2854.

Hanna M. Wallach. 2008. *Structured topic models for language*. Ph.D. thesis, University of Cambridge.

Cross-lingual Coreference Resolution of Pronouns

Michal Novák and Zdeněk Žabokrtský
Charles University in Prague, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, CZ-11800
{mnovak,zabokrtsky}@ufal.mff.cuni.cz

Abstract

This work is, to our knowledge, a first attempt at a machine learning approach to cross-lingual coreference resolution, i.e. coreference resolution (CR) performed on a bitext. Focusing on CR of English pronouns, we leverage language differences and enrich the feature set of a standard monolingual CR system for English with features extracted from the Czech side of the bitext. Our work also includes a supervised pronoun aligner that outperforms a GIZA++ baseline in terms of both intrinsic evaluation and evaluation on CR. The final cross-lingual CR system has successfully outperformed both a monolingual CR and a cross-lingual projection system.

1 Introduction

Coreference resolution (CR) is a well-established task in the field of Natural Language Processing (NLP). The majority of papers published so far has focused on the monolingual CR, mostly experimenting on the English data. An important step towards multilingual CR was the CoNLL-2012 Shared Task in Modeling Multilingual Unrestricted Coreference in OntoNotes, where the participants were asked to build a CR system that could be applied on three typologically different languages contained in the OntoNotes corpus (Hovy et al., 2006): English, Chinese, and Arabic.

Same just as in other NLP tasks such as part-of-speech tagging or parsing, recent years have witnessed a rising interest in cross-lingual projection techniques, mostly aiming at under-resourced languages.

However, little attention is paid to leveraging cross-lingual information for CR in two resource-rich languages. This is probably due to lack of bilingual resources annotated with coreference since such techniques would require rich linguistic annotation on both sides of the bitext. Moreover, to solve this issue using a supervised learner, one needs the gold standard of coreference at least on the target side of the bitext. On the other hand, given such data, the typological differences in languages can be exploited to aid a CR system to perform better than if CR is performed independently for each language.

The motivation for solving this task is threefold. Firstly, even though Statistical Machine Translation (SMT) has been attracting interest of the community for years, most systems do not take information beyond the sentence boundary into account, leaving the issues of discourse coherence unresolved. Having a better-quality bitext with coreference resolved could drive research in discourse-aware SMT forward. Secondly, although inter-sentential relations are neglected in SMT, current phrase-based system unintentionally resolve some of the coreference links within the sentence, using just the power of phrases. This might be leveraged by using the SMT output instead of a human-translated output in a cross-lingual CR scenario. Finally, even monolingual CR may be improved by applying semi-supervised learning methods in a smart way on a large bilingual corpus with automatic rich annotations, such as CzEng 1.0 (Bojar et al., 2012).

Our work examines cross-lingual CR on the Czech-English language pair. We focus on CR of English pronouns, particularly the 3rd person *central pronouns*. Central pronouns is a term coined by Quirk (1985) embracing personal, possessive and reflexive pronouns. For the sake of simplicity, we will denote

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 14–24, Dublin, Ireland, August 23-29 2014.

3rd person central pronouns by the word *pronouns* in the following. We ignore noun phrase coreference for two reasons. First, there has been no data set available for the Czech-English language pair with noun phrase coreference annotated, yet. Second, the language differences between languages show more clearly on pronouns than on nouns, as pronouns tend to be more constrained by various grammar rules across different languages.

Czech and English are typologically distant languages, which is also reflected in different behavior of pronouns. A cross-lingual CR system could substantially benefit from the necessity of the anaphor and its antecedent to agree in gender. Czech uses grammatical genders which are more evenly distributed among nouns than the notional genders[1] used in English, where male and female gender[2] are solely allocated to living objects. However, benefiting from the pronoun's gender becomes problematic for personal pronouns in subject position which are usually dropped from the surface representation in Czech. If their governing verb is in the past tense, the correct gender can be reconstructed from its form. With the verb in present or future tense, the pronoun's gender remains hidden. Possessive pronouns are used to a greater extent in English than in Czech. Same as articles, they play the role of determiners whereas in Czech, the determination and possession must be understood from the context. A missing Czech counterpart of an English possessive pronoun may indicate its antecedent to be in the same sentence. Moreover, Czech uses reflexive possessive pronouns, whose antecedent is easier to detect than for non-reflexive pronouns. On the other hand, English reflexive pronouns, unlike the Czech, carry gender and number information the resolver can benefit from.

In this work, we make to our knowledge a first attempt to leverage the language differences using a machine learning approach to improve CR on bitexts. To achieve this goal, we create a supervised CR model, proposing two sets of cross-lingual features: projected features used for Czech CR and an indicator feature of a projected Czech coreference link obtained by a Czech CR system. Note that for the latter set (actually comprising only a single feature), the Czech CR system would require gold annotation of Czech coreference. We did not consider new features that would address specific Czech-English correspondences.

The fact that a Czech counterpart is missing for many English pronouns has a negative effect on traditional unsupervised alignment approaches. We address this issue by a supervised aligner of pronouns that incorporates the result of the traditional aligner as a feature and adds other features that help detect the true Czech counterparts of English pronouns.

The structure of this paper is as follows: After introducing related work in Section 2 and describing the data used in experiments in Section 3, we present the design of a supervised approach to improve English pronoun alignment in Section 4. Section 5 describes the cross-lingual CR system and the experiments conducted with it. Finally, we discuss the main observations made in the experiments in Section 6 and conclude the paper in Section 7.

2 Related work

The task of coreference resolution has been studied for a few decades, with supervised systems dominating the field. The most popular approaches have been thoroughly summarized by Ng (2010).

The system for English CR we use has been built for automatic coreference annotation in the Czech-English parallel treebank CzEng 1.0 (Bojar et al., 2012). It is an implementation of the so-called mention ranking model, first introduced by Denis and Baldridge (2007).

Parallel bilingual data is often exploited to solve well-known tasks such as part-of-speech tagging (Das and Petrov, 2011), named entity recognition (Kim et al., 2012), name tagging (Li et al., 2012), and semantic role labeling (Zhuang and Zong, 2010). Undoubtedly, this approach is most popular with parsing. Joint parsing of both the source and the target text along with searching for the best alignment between the trees has been approached in a more (Burkett et al., 2010) or less (Smith and Smith, 2004; Burkett and Klein, 2008) integrated approach. However, much closer to our work is the research on

[1] "Nouns are classified semantically according to their coreferential relations with personal, reflexive and wh-pronouns." (Quirk et al., 1985, p.314)

[2] Quirk (1985) uses these terms instead of terms masculine and feminine related to grammatical gender.

bilingually-informed parsing by Haulrich (2012), in which English trees are used to enrich the feature set for a Danish parser and vice-versa. Rosa et al. (2012) explored the same approach on the Czech-English language pair. Moreover, they adapted this technique to parse the output of an SMT system.

As for coreference resolution in a bilingual scenario, most works focus on coreference projection (de Souza and Orsan, 2011; Rahman and Ng, 2012; Ogrodniczuk, 2013). Research on cross-lingual CR has been inhibited by the lack of coreference-annotated parallel corpora. There are only few such corpora, for instance an English-Romanian corpus containing full hand-annotated coreference chains including noun phrase coreference (Postolache et al., 2006) and two corpora with pronoun coreference annotations – Prague Czech-English Dependency Treebank 1.0 (Hajič et al., 2012, PCEDT) and the recently published English-German corpus ParCor 1.0 (Guillou et al., 2014).

However, the only attempts at cross-lingual CR date back to the time before these corpora were released. Harabagiu and Maiorano (2000) designed a CR system for English-Romanian bitexts while Mitkov and Barbu (2003) focused on the English-French language pair. Both extended their rule-based monolingual CR systems to apply some high-precision rules from one language to enhance the result in the other language. They both reported an improvement of about 4% in precision compared to the monolingual systems.

As concerns a machine learning approach, in the work by Veselovská et al. (2012), PCEDT was employed in related tasks – to identifying types of the English personal pronoun *it* and Czech types of the unexpressed subject. The tasks have been addressed by the isolated monolingual systems as well as by taking advantage of the features from the other language.

3 Main source of the data

As mentioned in Section 2, Czech is one of a few languages for which a coreference-annotated parallel corpus has been built – The Prague Czech-English Dependency Treebank (Hajič et al., 2012, PCEDT).[3]

PCEDT is a manually annotated Czech-English parallel treebank comprising over 1.2 million words for each language in almost 50,000 sentence pairs. The English part contains the entire Penn Treebank–Wall Street Journal Section (Linguistic Data Consortium, 1999) transformed into dependency trees, whereas the Czech part comprises the translations of all the texts from the English part. The data from both parts are annotated on three layers of linguistic description following the Prague tectogrammatics theory (Sgall, 1967; Sgall et al., 1986) – the morphological layer (where each token from the sentence gets a lemma and a POS tag), the analytical layer (surface syntax in the form of a dependency tree, where each node corresponds to a token in the sentence) and the tectogrammatical layer. Tectogrammatical representation of a sentence is a dependency tree, where only content words have their own nodes; on the other hand, it contains additional nodes, e.g., for pronouns unexpressed on the surface. This is also the layer where the coreference relations are annotated. PCEDT includes annotation of pronoun coreference and the so-called grammatical coreference[4] for Czech as well as English.

For the purpose of this work, we ignore all annotations originally provided by PCEDT. Annotations on the tectogrammatical layer, which is in the center of this work's attention, are mostly manual there. But to truly simulate the real-world scenario when given just a pair of parallel texts, we need to replace them with ones carried out in a fully automatic manner. The only two exceptions, where we employ the gold annotations, are the relations we aim to model, i.e. coreference links and our own annotation of alignment for English personal pronouns (see Section 4.1).

3.1 Fully Automatic Annotation

We have conducted automatic linguistic analysis on both the English and the Czech part of PCEDT, transforming the individual sentences into multi-layer dependency tree structures based on the Prague tectogrammatics theory. The analysis was carried out within the Treex framework (Popel and Žabokrtský, 2010).

[3] http://hdl.handle.net/11858/00-097C-0000-0015-8DAF-4

[4] Its antecedent is imposed by the grammar of the language, e.g. coreference of relative pronouns.

Treex is a multi-purpose open-source framework for NLP applications development, which integrates a wide range of modules, such as tools for sentence splitting, tokenization, morphological analysis, part-of-speech tagging, shallow and deep syntax parsing, named entity recognition, anaphora resolution, among others.

Moreover, we performed an unsupervised word alignment on the complete PCEDT using the MGIZA++ tool (Gao and Vogel, 2008), which is a multi-threaded version of the popular GIZA++ (Och and Ney, 2000) that supports applying a saved model on a new sentence pair. We used a model trained on CzEng 1.0, which is about 300 times bigger in terms of the number of sentence pairs. The resulting alignment of the `intersection` and `grow-diag-final-and` types was subsequently projected onto the tectogrammatical layer. Furthermore, a simple heuristic was applied to find the English counterparts for reconstructed Czech personal pronouns. We denote this alignment as the *original* in the following.

4 Supervised alignment

The alignment described in the previous section is sufficiently accurate for content words, such as verbs, nouns, and adjectives. However, errors become more frequent as we move to pronouns. Some reasons for this have already been outlined in Section 1, i.e. dropped subject personal pronouns and omitted possessive pronouns in Czech. In addition, English uses a pleonastic variant of the pronoun *it*, which also has no correspondence in Czech. Personal pronouns function in a sentence as a replacement of nouns. Thus, it is no exception if a pronoun is translated into a noun. And finally, the translation may be reworded to such an extent that the pronoun would carry no valuable information, and it disappears. All these cases are difficult for GIZA++ to tackle.

The pronoun correspondence problem has been already faced concerning the alignment of the personal pronoun *it* by Novák et al. (2013). The authors tried to find the Czech counterpart of *it* by taking the node that is aligned to the parent of *it* on the Czech side and picking the argument of the aligned node that agrees on the semantic role with the particular *it*. This approach assumed that the unsupervised alignment of the parent, which is likely to be a content word, is of higher quality than the alignment of *it* itself. Furthermore, it relied on high-accuracy semantic role labeling, which could only be justified because the experiments were conducted on data manually annotated with semantic roles.

As we are working with fully automatic annotations (i.e., much less reliable) and a wider range of words to align, we cannot just copy this rule-based approach. However, we can take a more robust approach of supervised machine learning and transform Novák et al.'s rule to one of the features in our alignment model.

In Section 4.1, we describe the manual annotation of alignment, then introduce the supervised model in Section 4.2, using features described in Section 4.3. Finally, we show the evaluation results of the alignment model in Section 4.4.

4.1 Manual Annotation of the Data

Supervised learning requires that the training data are manually labeled with a target variable. For this purpose, we set aside the section 19 of PCEDT. In this data, all occurrences of English personal pronouns have been coupled with its Czech counterpart by one human annotator. If no suitable Czech expression was found, the annotator identified a possible cause of the missing counterpart. The causes were then categorized into three classes – pleonastic *it*, missing possessive pronoun and missing correspondence due to translation rewording. So far, we do not distinguish these classes in our models and treat them in the same manner.

We managed to align 471 occurrences of personal pronouns, which account for over 50% of all occurrences in the section. The overall statistics of how English personal pronouns are translated into Czech is shown in Table 1.

It shows that more than 55% of English personal pronouns are dropped from the surface representation of the Czech sentence, though still present in its deep structure. In contrast, English pleonastic pronouns are not present even there. An interesting observation is that more than half of English possessives are either translated as reflexive possessives or completely missing in the Czech sentence. All these

CS\EN	personal	possessive	reflexive	Total
personal unexpressed	147	1		148
personal	37	2		39
demonstrative	17	1		18
noun	15	6		21
possessive	3	78		81
reflexive possessive		68		68
reflexive	1	2	5	8
other	6	1	3	10
pleonastic	24			24
reword	12	4		16
no possessive		38		38
Total	262	201	8	471

Table 1: The statistics on the correspondence of English personal pronouns to their Czech counterparts. The last three Czech categories indicate the reason why there is no corresponding word in Czech for an English pronoun.

phenomena might in the end be a source of helpful information to the CR system.

4.2 Model

The nature of the task of aligning a given English pronoun to its Czech counterpart is to pick the best-fitting one from a bunch of candidates. The set of candidates consists of all tectogrammatical nodes in the aligned Czech sentence. To allow the system to select no correspondence for a pronoun, we add a special candidate representing the null alignment.

We represent the candidate ranking task as a discriminative log-linear model trained in a cost-sensitive, one-against-all strategy with label-dependent features (`csoaa-ldf`) provided by the Vowpal Wabbit[5] machine learning toolkit. The feature weights are optimized by running stochastic gradient descent in 40 passes over the training data.

4.3 Features

The feature set consists of the following types of features, which consider an English pronoun and a Czech candidate from the corresponding Czech tree:

- **Original alignment features:** presumably the most valuable set of features. It indicates if there is a link between the two nodes in the original alignment and if there is any between their parents.

- **Graph features:** we designed these features to somehow reflect the distance between the nodes. The pair of aligned tectogrammatical trees is treated as a bipartite graph and a shortest path between the nodes is found using a sequence of dependency edges and a single alignment link. We applied the Dijkstra algorithm to find the shortest path. We ensure that it only uses a single alignment link by setting large weights to alignments and small weights to dependency edges, i.e., 100 and 1, respectively. The features then comprise the length of the shortest path and the sequence of edge labels (parent, child, alignment).

- **Grammatical features:** these include lemmas, part-of-speech tags, reflexivity indicators, semantic role labels both for each of the nodes individually and as a concatenation of the two.

- **Combined features:** these features combine selected features from the types mentioned above. The concatenation of parents' alignment and semantic role correspondence mimics the rule Novák et al. (2013) used to get better Czech counterparts for English *it* (see Section 4). Furthermore, features combining lemmas with direct alignment or alignment through parents are included.

[5] `https://github.com/JohnLangford/vowpal_wabbit/wiki`

Method	Train				Test			
	A	P	R	F	A	P	R	F
ORIGINAL	–	–	–	–	73.04	75.55	82.40	78.83
SUPERVISED	88.37	90.18	90.34	90.26	84.50	88.52	86.40	87.45

Table 2: Evaluation results of English-to-Czech pronoun alignment. The quality is measured in terms of accuracy (A), precision (P), recall (R), and F1-score (F).

4.4 Experiments and Results

The small amount of manually annotated data led us to evaluate alignment models by 10-fold cross-validation, with the results on the train and test partitions averaged over all folds.

We measured the quality of produced the alignment links in terms of both accuracy and F1-score, i.e., as the harmonic mean of precision and recall. While accuracy positively scores also the cases when a node is correctly labeled as having no alignment, precision and recall neglect these cases at all, thus describing how good a method is in finding the correct counterpart for a node.

Table 2 shows the performance of the supervised model with the best combination of features and learning method parameters and compares it to the original alignment described in Section 3.1. It shows an improvement of about 9% absolute in terms of both accuracy and F-score.

5 Cross-lingual coreference resolver for English

In this section, we describe cross-lingual coreference resolution. The CR system we use definitely does not aim to compete with current state-of-the-art systems. However, for the purpose of research on cross-lingual CR, it can be employed as a reasonable baseline.

In Section 5.1, we describe the supervised CR model trained and tested on the data described in Section 5.2. We elaborate more on the design of English and aligned features in Section 5.3 and Section 5.4, respectively. Finally, several variants of the CR system are evaluated and compared in Section 5.5.

5.1 Coreference model

Our resolver employs a supervised model denoted as *mention ranker* by Ng (2010). Its advantage lies in judging all antecedent candidates simultaneously, and then picking the candidate with the highest score as the predicted antecedent. However, it is unable to exploit features that describe already formed clusters of mentions belonging to the same entity. A typical issue related to ranking models is how to deal with non-anaphoric mentions. We use the approach introduced by Rahman and Ng (2009) – adding a special candidate that indicates no anaphor.

Since this work focuses only on the so-called pronoun resolution, all the anaphor candidates are English 3rd person central pronouns, i.e. personal, possessive and reflexive pronouns.

For every anaphor, we collect in the set of its antecedent candidates all semantic nouns[6] from the previous sentence and the part of the current sentence prior to the anaphor.

CR can be treated as a ranking task, so we represent it in the same way as we handled alignment in Section 3.1 – as a discriminative log-linear model trained in the `csoaa-ldf` strategy by the Vowpal Wabbit tool. The feature weights are optimized by running stochastic gradient descent in 20-80 passes (the number differs across the experiments) over the training data.

5.2 Data

Models for coreference were trained on data extracted from sections 00–18 of the automatically analyzed PCEDT (as described in Section 3). Sections 20–21 have been employed as development testing data and Sections 22–24 as evaluation testing data. The development set has been used to select the best configuration, which was subsequently tested on the evaluation set. The training, development, and evaluation set consist of 19,294, 1,988 and 2,591 instances with 86%, 67%, and 73% anaphoric instances, respectively.

[6]Semantic nouns are all nouns as well as pronouns acting as a noun.

5.3 English Features

A wide range of features used by us had already been proven to be beneficial for the task of CR in multiple prior works. The majority of the features presented here have already been used in the CR system for Czech (Nguy et al., 2009); we keep just the language-independent. Furthermore, several grammatical and positional features proposed by Charniak and Elsner (2009) have been added. Finally, the feature set has been enriched with the information on named entities and WordNet[7] classes. All the features disregard dependent members of a mention, describing just the head of the mention. They can be divided into several categories:

- **Distance features:** number of sentences, clauses, and words between the anaphor and the antecedent candidate; the order of the candidate,
- **Grammatical features:** morphological number and gender of both the anaphor and the antecedent candidate, agreement in gender and number; part-of-speech tag,
- **Function features:** they exploit dependency labels on the analytical layer and semantic roles on the tectogrammatical layer; they also include an indicator of whether the mention plays a role of an argument or an adjunct in the governing phrase,
- **Parent features:** the features of both nodes' parents, e.g. their lemmas or semantic roles, are compared; an indicator of whether a mention is in coordination,
- **Semantic features:** WordNet classes the head word is assigned to,
- **Named entity features:** the named entity category and subcategory returned by Stanford named entity recognizer.[8] This includes also the indicator of whether the mention is a name of a person,
- **Charniak features:** anaphor type (pronoun in subject position, in object position, possessive pronoun, reflexive pronoun, other); antecedent type (noun, pronoun, other); antecedent syntactic type (subject, object, prepositional phrase, other).

We denote this feature set as EN in all our experiments.

5.4 Alignment features

The features from the Czech nodes aligned to the given English anaphor and antecedent candidate are obtained by moving to the corresponding Czech nodes and extracting the features as though we are trying to resolve a Czech coreference link. As outlined in Section 1, we designed two sets of features: CS and CS-COREF.

The CS set consists of features introduced by Nguy et. al (2009). Most of them, namely the categories of distance, function, and parent features, are extracted in the same manner as the English ones in the previous section. Grammatical features also contain the full positional morphological tag as designed by Hajič (2004). Semantic features employ a different knowledge base, replacing WordNet by the Czech portion of EuroWordNet (Vossen, 1998). In addition to the features more or less shared with the English side, the Czech feature set includes a probability estimate of the antecedent candidate co-occurring with its governing verb. This statistics has been collected on Czech National Corpus (CNC, 2005).

The CS-COREF set consists of a single binary feature indicating if there is a coreference relation between the nodes predicted by the Czech CR system (Nguy et al., 2009), or not.

[7] http://wordnet.princeton.edu

[8] http://nlp.stanford.edu/software/CRF-NER.shtml

5.5 Experiments and Results

The different feature sets proposed in the previous sections suggest an obvious set of experiments. The system trained only on the monolingual EN features is put as a baseline.

The rest our experimental setups use alignment features, forming three combinations with EN features: EN + CS, EN + CS-COREF, and EN + CS + CS-COREF. Moreover, these three experiments can be run on the data provided either with the original or supervised alignment, which serves as extrinsic evaluation of alignment approaches. This allows us to confirm or deny the hypothesis that the alignment plays a significant role in cross-lingual CR (see Section 4).

For comparison, we also evaluated the system that simply projects coreference links obtained by the Czech CR system to English.

The performance of a CR system is usually measured by scores that treat CR as a clustering problem, e.g., MUC, B^3, CEAF. As this work focuses merely on a subset of coreference expressions – pronouns – and we only compare different feature sets trained in the same framework, we resorted to the simplest metrics with a sufficient expression power. For each English pronoun we test if its predicted antecedent hits any of the true antecedents within the window of the current and the previous sentence. Given this indicator we calculate precision, recall, and F1-score, which takes into account only the nodes for which a relation with another node exists – referential pronouns in this case (similarly to the alignment evaluation in Section 4.4). Likewise, in order to assess quality of detecting non-referential pronouns, accuracy is computed as well.

The final results are shown in Table 3. The overall higher numbers on the evaluation set than on the development set probably result from a different proportion of non-anaphoric pronouns (see Section 5.2). The smaller difference in F1-score than in accuracy also supports this explanation.

The coreference projection scores a great deal below the baseline, which suggests that this approach is worth using only if manual annotation for at least a small amount of target language data (English in our case) is extremely expensive.

As for the cross-lingual CR on the original alignment, all three feature set combinations have beaten the baseline. The EN + CS-COREF system confirmed the added value of the CS-COREF feature, which, unlike the CS feature set, conveys latent information on true Czech coreference links. Even the combination of all features performs worse than CS-COREF alone.

Moving to the experiments with supervised alignment, we can see the findings from Section 4.4 confirmed also in the extrinsic evaluation. All three systems outperform not only the baseline, but also all the systems working on the original alignment. Moreover, both accuracy and F1-score order the three feature combinations in the expected way, where the overall winner improves over the baseline in more than 1% absolute. This improvement is significant[9] at p-level $p \leq 0.1$ but not at p-level $p \leq 0.05$.

6 Discussion

Using information from Czech parallel texts in English CR led to an improvement in terms of automatic measures. To see what the main aspects in which the Czech text positively impacts the CR performance are, we compared the output of the system trained only on the EN features with systems working on the EN + CS and EN + CS-COREF feature sets. We used the results of the experiments run on the development set with supervised alignment for this comparison.

Out of 1988 coreference instances in the development set, the EN + CS system improved the output in 49 cases, while it worsened the output in 23 cases. The rest remained unchanged. Likewise, the EN + CS-COREF system scored better than the EN one in 63 instances, while it failed in 39 instances.

The inspection of 10% instances for which the systems differed revealed that the cases when the cross-lingual system scored better than the monolingual concur with the language differences described in Section 1. We found that in these cases, the pronoun is often a pleonastic *it* or a possessive pronoun with a Czech reflexive possessive counterpart. Finally, we noticed improvements in cases where the Czech antecedent is easier to determine due to agreement in gender and number.

[9] Significance has been calculated by bootstrap resampling using 100,000 samples.

Setup	Train				Dev				Eval			
	A	P	R	F	A	P	R	F	A	P	R	F
EN	79.13	80.12	86.00	82.96	60.97	60.28	79.14	68.43	63.72	63.28	78.78	70.19
Original alignment												
CS-COREF projection	28.64	49.57	21.75	30.23	36.55	41.98	24.66	31.07	33.33	42.38	21.58	28.60
EN + CS-COREF	78.31	79.27	85.25	82.15	61.77	61.07	80.45	69.44	64.30	63.74	79.62	70.80
EN + CS	83.32	84.05	89.97	86.91	61.97	61.15	80.23	69.40	64.07	63.72	78.62	70.39
EN + CS + CS-COREF	80.75	81.52	87.61	84.46	62.27	61.33	80.96	69.79	64.03	63.59	79.57	70.69
Supervised alignment												
CS-COREF projection	30.74	49.91	24.87	33.20	36.60	41.38	27.61	33.12	33.60	41.85	23.98	30.49
EN + CS	83.19	83.98	89.73	86.76	62.27	61.42	80.60	69.72	64.53	64.13	79.09	70.83
EN + CS-COREF	79.27	80.20	85.89	82.95	62.17	61.27	81.11	69.81	64.65	64.11	79.67	71.05
EN + CS + CS-COREF	81.99	82.78	88.53	85.56	62.68	61.59	81.62	70.20	64.69	64.38	79.67	71.22

Table 3: Evaluation results of monolingual CR, CR via projection, and cross-lingual CR system trained and tested on the data with both the original and supervised alignment. Performance is measured in terms of accuracy (A), precision (P), recall (R) and F1-score (F).

We did not encounter an example of improvement for an English possessive pronoun having no Czech counterpart. We might have inspected too little data for it to appear. However, these cases may get covered after the features combining English and Czech features will be introduced.

7 Conclusion

This work introduced a largely unexplored task in the field of CR – cross-lingual CR. Given a Czech-English bitext, we sought to improve the performance of an English pronoun CR system by enriching the feature set with features from the aligned Czech text. Consistent improvements over the monolingual system confirmed that cross-language differences in pronoun behavior are big enough to affect the result. Furthermore, we have found that the quality of alignment is vital for this task.

In future work, we plan to apply this approach on a much larger parallel corpus and employ semi-supervised techniques to improve cross-lingual as well as monolingual CR. Moreover, human translation in the bitext can be replaced with the output of SMT system to see if we can produce valuable features for CR from the machine-translated source text.

Acknowledgments

This work has been supported by the EU FP7 project Khresmoi (contract no. 257528). This work has been using language resources developed and/or stored and/or distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2010013). We would like to thank our colleague Ondřej Dušek for language correction and three anonymous reviewers for their remarks and suggestions.

References

Ondřej Bojar, Zdeněk Žabokrtský, Ondřej Dušek, Petra Galuščáková, Martin Majliš, David Mareček, Jiří Maršík, Michal Novák, Martin Popel, and Aleš Tamchyna. 2012. The joy of parallelism with CzEng 1.0. In *Proceedings of LREC 2012*, Istanbul, Turkey. European Language Resources Association.

David Burkett and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Burkett, John Blitzer, and Dan Klein. 2010. Joint parsing and alignment with weakly synchronized grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eugene Charniak and Micha Elsner. 2009. EM works for pronoun anaphora resolution. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, Stroudsburg, PA, USA. Association for Computational Linguistics.

CNC. 2005. Czech national corpus – SYN2005. Prague, Czech Republic. Institute of the Czech National Corpus.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, Stroudsburg, PA, USA. Association for Computational Linguistics.

José Guilherme Camargo de Souza and Constantin Orsan. 2011. Can projected chains in parallel corpora help coreference resolution? In *Proceedings of the 8th International Conference on Anaphora Processing and Applications*, Berlin, Heidelberg. Springer-Verlag.

Pascal Denis and Jason Baldridge. 2007. A ranking approach to pronoun resolution. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Liane Guillou, Christian Hardmeier, Aaron Smith, Jrg Tiedemann, and Bonnie Webber. 2014. ParCor 1.0: A parallel pronoun-coreference corpus to support statistical MT. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC-2014)*, Reykjavik, Iceland. European Language Resources Association.

Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Ondřej Bojar, Silvie Cinková, Eva Fučíková, Marie Mikulová, Petr Pajas, Jan Popelka, Jiří Semecký, Jana Šindlerová, Jan Štěpánek, Josef Toman, Zdeka Urešová, and Zdeněk Žabokrtský. 2012. Announcing Prague Czech-English Dependency Treebank 2.0. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, Istanbul, Turkey. European Language Resources Association.

Jan Hajič. 2004. *Disambiguation of Rich Inflection (Computational Morphology of Czech)*. Nakladatelství Karolinum.

Sanda M. Harabagiu and Steven J. Maiorano. 2000. Multilingual coreference resolution. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Martin Wittorff Haulrich. 2012. *Data-driven bitext dependency parsing and alignment*. Ph.D. thesis, Copenhagen Business School.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: the 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sungchul Kim, Kristina Toutanova, and Hwanjo Yu. 2012. Multilingual named entity recognition using parallel data and metadata from Wikipedia. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qi Li, Haibo Li, Heng Ji, Wen Wang, Jing Zheng, and Fei Huang. 2012. Joint bilingual name tagging for parallel corpora. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, New York, NY, USA. ACM.

Linguistic Data Consortium. 1999. Penn Treebank 3. LDC99T42.

Ruslan Mitkov and Catalina Barbu. 2003. Using bilingual corpora to improve pronoun resolution. *Languages in contrast*, 4(2).

Vincent Ng. 2010. Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Giang Linh Nguy, Václav Novák, and Zdeněk Žabokrtský. 2009. Comparison of classification and ranking approaches to pronominal anaphora resolution in Czech. In *Proceedings of the SIGDIAL 2009 Conference*, London, UK. The Association for Computational Linguistics.

Michal Novák, Anna Nedoluzhko, and Zdeněk Žabokrtský. 2013. Translation of "it" in a deep syntax framework. In *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Workshop on Discourse in Machine Translation*, Sofija, Bulgaria. Omnipress, Inc.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Maciej Ogrodniczuk. 2013. Translation- and projection-based unsupervised coreference resolution for Polish. In *Language Processing and Intelligent Information Systems*, number 7912, Berlin / Heidelberg. Springer.

Martin Popel and Zdeněk Žabokrtský. 2010. TectoMT: Modular NLP framework. In *Lecture Notes in Artificial Intelligence, Proceedings of the 7th International Conference on Advances in Natural Language Processing (IceTAL 2010)*, volume 6233, Berlin / Heidelberg. Springer.

Oana Postolache, Dan Cristea, and Constantin Orasan. 2006. Transferring coreference chains through word alignment. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, Genoa, Italy. European Language Resources Association.

Randolph Quirk, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman.

Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Altaf Rahman and Vincent Ng. 2012. Translation-based projection for multilingual coreference resolution. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rudolf Rosa, Ondřej Dušek, David Mareček, and Martin Popel. 2012. Using parallel features in parsing of machine-translated sentences for correction of grammatical errors. In *Proceedings of the Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. D. Reidel Publishing Company, Dordrecht.

Petr Sgall. 1967. *Generativní popis jazyka a česká deklinace*. Academia, Prague, Czech Republic.

David A. Smith and Noah A. Smith. 2004. Bilingual parsing with factored estimation: Using English to parse Korean. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Kateřina Veselovská, Giang Linh Nguy, and Michal Novák. 2012. Using Czech-English parallel corpora in automatic identification of "it". In *The Fifth Workshop on Building and Using Comparable Corpora*, İstanbul, Turkey. European Language Resources Association.

Piek Vossen, editor. 1998. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers, Norwell, MA, USA.

Tao Zhuang and Chengqing Zong. 2010. Joint inference for bilingual semantic role labeling. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, Stroudsburg, PA, USA. Association for Computational Linguistics.

A Context-Aware NLP Approach For Noteworthiness Detection in Cellphone Conversations

Francesca Bonin *
Trinity College Dublin
Dublin, Ireland
boninf@tcd.ie

Jose San Pedro
Telefonica Research
Barcelona, Spain
jspw@tid.es

Nuria Oliver
Telefonica Research
Barcelona, Spain
nuriao@tid.es

Abstract

This papers presents a context-aware NLP approach to automatically detect noteworthy information in spontaneous mobile phone conversations. The proposed method uses a supervised modeling strategy which considers both features from the content of the conversation as well as contextual information from the call. We empirically analyze the predictive performance of features of different nature on a corpus of mobile phone conversations. The results of this study reveal that the context of the conversation plays a crucial role on boosting the predictive performance of the model.

1 Introduction

More than 6 billion people worldwide use their cellphones daily for a variety of purposes: contacting colleagues, relatives or friends, doing business, getting help in emergency situations, etc. Previous work (Carrascal et al., 2012) has shown that almost 40% of users frequently feel the need to recall bits of information from their phone conversations and that 27% of the users consider the recall task to be difficult, mainly because taking notes during a mobile phone call is not always possible (*e.g.* hands not free, lack of time or devices for note-taking). In a related user study, Cycyl *et al.* reveal that users are often engaged in concurrent tasks during mobile phone conversations (*e.g.* walking, jogging, driving, cooking, etc), which makes taking notes an unfeasible task (Cycil et al., 2013).

In this setting, information extraction techniques could be applied to automatically detect noteworthy information from mobile phone conversations. Related studies have focused on detecting noteworthiness from meeting transcripts (Banerjee and Rudnicky, 2009). However, very little work has been done to date to identify this kind of information in other types of human communication, such as spontaneous phone conversations.

In this paper, we present a data-driven information extraction approach aimed at automatically detecting fragments of phone conversations worth annotating for future recall, *i.e. noteworthy*. These *call notes* could then be presented to the users to enable fast browsing of their conversation history, and leveraged to design efficient information interaction techniques for supporting smart user interfaces.

Given the particular characteristics of mobile phone calls, detecting noteworthiness in them is challenging at many levels. First, the audio is captured in a natural environment rather than in controlled settings, which results in noisy signals, and consequently in noisy transcriptions. Second, the conversations are highly fragmented due to their spontaneous nature. Finally, at a conceptual level, judging which pieces of information are noteworthy is a very subjective task, as emerged in (Banerjee and Rudnicky, 2009), who investigated the feasibility of the task by conducting a Wizard of Oz-based user study.

Our noteworthiness modeling approach considers a supervised learning paradigm which takes into account two types of information: (1) Contextual information both from the call (*where, when, to whom,* ...) and the users (*gender, age,* ...); and (2) Content information of the conversation. The combination

* The work was conducted while the author was intern at Telefonica Research, Barcelona, Spain.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 25–36, Dublin, Ireland, August 23-29 2014.

of both sources of information enhances the flexibility of the model to accurately predict noteworthiness in different use scenarios.

The main contributions of this paper are:

i) We propose and evaluate a supervised machine learning model to automatically detect noteworthy segments of phone conversations. Our approach adopts a hybrid strategy to model conversations exploiting both *content* and *context*-related information.

ii) We propose a new set of content and context-based features specifically designed to detect noteworthy information in our corpus of real-world cellphone conversations, and compare their effectiveness

iii) We provide a discussion of the results, derived from our quantitative and qualitative analyses.

The paper is structured as follows. Relevant previous work is presented in Section 2. Section 3 describes the corpus of phone conversations and the annotations provided by the participants. In Section 4 we describe in-depth the extracted features. Our experimental validation and results are presented in Section 5. Finally, Section 6 summarizes our findings and highlights some lines of future research.

2 Related work

Noteworthiness detection in conversations can be considered to be a particular form of summarization: the aim is to summarize the conversation by keeping only the *relevant* pieces of information that the user would like to refer to at a later time. Although related, the main distinction between automatic summarization and detection of noteworthy information lays in the notion of *relevance*. The former aims at generating a comprehensive record of the conversation, while the latter considers only fragments worth registering for future recall.

Considerable research activity has recently been devoted to automatic text and speech summarization (Maskey and Hirschberg, 2003). Many approaches have been proposed in the literature, including cluster (Zhang et al., 2005) and graph-based methods (Garg et al., 2009; Wang and Liu, 2011) and machine learning techniques (Jian Zhang et al., 2007; Maskey and Hirschberg, 2006; Galley, 2006), where the task is tackled as a binary classification problem considering whether the sentence is a good candidate for a summary or not. In addition, different types of features have been used, including lexical, acoustic and structural characteristics (Xie et al., 2008; Maskey and Hirschberg, 2005). Recent works have been focused on adapting summarization to the social context, exploiting user generated contents associated with the documents (Yang et al., 2011; Hu et al., 2012). Implicit and explicit community feedback in online collaborative websites have also been leveraged to detect highlights of media assets (San Pedro et al., 2009).

However, few studies have focused on noteworthiness detection. Banerjee *et al.* investigate the feasibility of discovering noteworthy pieces of information in meetings by means of a Wizard of Oz-based user study where a human suggested notes to meeting participants during the meeting. The authors found that the human annotator obtained a precision of 35% and a recall of 41.5%. In the same work, Banerjee *et al.* reports a low inter annotator agreement (IAA) in noteworthiness discovery. In a related work –probably the most relevant prior-art to our work, the authors apply extractive meeting summarization techniques to automatically detect noteworthy utterances in meetings (Banerjee and Rudnicky, 2008). They train a Decision Tree classifier over a collection of 5 meetings, obtaining an F-score of 0.14. This result highlights the difficulty of the task at hand and motivates to explore alternative approaches.

To overcome the difficulties posed by this task we propose two main contributions: 1) the use of novel features engineered ad-hoc for this task, and 2) the use of contextual information. While the former adapts the document representation to the specific problem setting, the latter allows to enhance the representation with orthogonal information which many times provides a higher discriminative power. This approach has been used successfully in related fields; for instance, in information retrieval tasks rich multimodal queries have been shown to effectively boost the retrieval performance compared to pure textual queries (Yeh et al., 2011).

3 Corpus Collection

We used a corpus of cellphone conversations collected in a previous study (Carrascal et al., 2012). In this study, a large sample of mobile phone conversations was recorded, semi-automatically transcribed[1] and manually annotated for relevance by their participants. Over 64 days, 796 mobile phone conversations from 62 volunteering subjects (20 female) were recorded. All the participants were Spanish native speakers, and the conversations were recorded and transcribed in Spanish. Metadata about the call (*e.g.* duration, date, time) was also stored along with the actual conversation and its transcript. More details about the corpus collection process can be found in (Carrascal et al., 2012).

All the participants were first asked to fill out a pre-study questionnaire where they provided some personal information, including gender, marital status, education and income. Then they were asked to annotate what parts of their calls that they would like to take a note of: *i.e.* noteworthy fragments of conversations. To this end, participants used a Web-based interface that gave them access to their calls and allowed them to highlight with the mouse the parts of the transcript that they considered to be worth keeping for future reference.

We used these annotations as the ground truth for the studies presented in this paper, considering them as the ideal noteworthy parts of the calls. For privacy reasons, due to the sensible nature of the data (i.e. private phone conversations) we could not consider alternative ground truth generation schemes, for instance collecting annotations from users other than the callers themselves.[2]

Finally, the participants were asked to fill out a questionnaire after annotating each call, which was used to collect contextual information, including: location of the call (*i.e at work, at home, while commuting, while doing shopping, while exercising*), and category of the call (*i.e. discuss a topic, taking an appointment, give/receive information, asking a favor, social*).

3.1 Characteristics of the Corpus

The original conversation collection consists of a total of 796 conversations, of an average length of 178 seconds ($s = 384$ sec.). We pre-filtered this original set to exclude calls with problems in the transcript (*e.g.* empty transcript, only one speaker audible, etc). Out of the entire corpus we finally selected 659 conversations. We denote this subset of the corpus as the $\mathcal{G}$ dataset. The $\mathcal{G}$ dataset comprises $22,474$ turns, with an average of 34.10 ($s = 45$) turns per conversation. From these, only 671 are annotated as being noteworthy (2.98%), which represent an average of 1.02 turns ($s = 1.803$) per call. Given that the vast majority of turns (97.2%) are not annotated, this can be considered a highly unbalanced dataset, which makes the automatic modeling problem more challenging.

Hence, we considered a second dataset which included only the 295 calls from the $\mathcal{G}$ dataset containing at least one annotation. This second subset, denoted as $\mathcal{A}$ amounts for approximately 45% of the $\mathcal{G}$ dataset. The $\mathcal{A}$ dataset features $10,642$ turns, with an average of 36.07 ($s = 33$) turns per conversation. From these, again 671 (6.3%) are annotated, which represent an average of 2.275 ($s = 2.09$) per call. The $\mathcal{A}$ dataset is still highly unbalanced but significantly less than the $\mathcal{G}$ dataset. Table 1 summarizes the high level characteristics of each dataset.

		Turns		Annotated Turns	
	# Calls	Total	$avg.$ per call	Total	Fraction
$\mathcal{G}$	659	$22,474$	34.1 ($s = 45$)	671	2.9%
$\mathcal{A}$	295	$10,642$	36 ($s = 33$)	671	6.3%

Table 1: General statistics on $\mathcal{G}$ and $\mathcal{A}$ datasets.

Class	Annotations
I	*We are in front of the fruit shop*
RoA	*Tomorrow we go to look for the swimsuit*
RI	*Are you coming to eat? At what time*
O	*Sure, it's normal*

Table 2: Examples of annotations.

Given the complexity of the modeling problem, we studied the note taking behavior of participants to identify relevant patterns that would simplify the problem. To this end, we conducted a quantitative analysis of the note taking behavior of participants. We found that users tend to highlight complete

[1] Participants were given the opportunity to revise transcriptions during the annotation phase.

[2] Receivers of the calls were aware of the study and were given the possibility to not participate in the call, but were not directed involved in the study.

turns as relevant, instead of parts of the turns. On average, 66.57% ($s = 35.87$) of the words within an annotated turn are highlighted, with a median value of 80%. Hence, we decided to use *turns* –rather than individual words– as the unit to be automatically detected as noteworthy. Using this approach, a turn is considered to be noteworthy if it contains at least one annotated word.

3.2 Qualitative Analysis of the Corpus

Since our aim is to detect the noteworthy turns within a call, we conducted a preliminary qualitative analysis to understand the nature of the annotations entered by the participants in the study. We distinguished 4 types of annotations: *Giving Information (I)*, *Requesting Information (RI)*, *Reporting on an Action (RoA)* and *Other (O)*. Examples of these 4 types of annotations are presented in Table 2. We collected annotations from three collaborators of our lab for a total of 54 randomly selected turns from the $\mathcal{A}$ dataset (IAA, $Fleiss\ Kappa = 0.54$ (Fleiss, 1971)).

We found that 47% of the turns were classified as belonging to the *Giving Information* category, 22% of the turns to the *Request Information* category, 26% to the *Other* category, and only 3% were classified as *Report on an Action*. Intuitively, we had expected the *Giving Information* category to be the most common in the annotated turns. However, the results obtained show that the other types of annotations are also well represented in the data.

Two main interesting aspects emerge. First, while the vast majority of annotations correspond to turns where a piece of information is given (*e.g. We meet at 3pm*), turns where information is requested are also well represented in the sample. There are plausible explanations for this behavior, such as users trying to include more context in the annotations. Second, more than 25% of this manually annotated dataset was marked under the *Other* category, which includes turns with very diverse functionalities (*e.g.* greetings, statements of agreement). This reveals that participants tend to annotate turns with very diverse functional aspects, which poses a challenge to be added to the unbalanced nature of the dataset.

4 Feature Extraction

We follow a supervised machine learning approach to automatically detect noteworthy turns in conversations. In this section we describe the features that we compute to represent conversations and which have been engineered to capture information relevant to the problem at hand. We have divided the set of features into two categories: **C**ontent features, that we denote with the letter **C**, and conte**X**t features, that we denote with the letter **X**.

4.1 Content Features

Content features are computed by analyzing the content of the conversations. We use as input the textual information resulting from the semi-automatic transcription of the calls. Note that we do not make use of any conversational acoustic information. While the analysis of the acoustic signal may reveal additional cues useful for noteworthiness detection, it lies out of the scope of this work.

In order to extract features from the transcript, we first pre-proces the datasets (split in turns, lemmatized, PoS tagged). Also, we extract and classify Named Entities (NEs).[3] We extract 42 content-based features which include both variations of features previously used in the meeting summarization literature and novel features particularly adapted to our task. However, in contrast to related work on meeting summarization, we do not extract content features based on lexical similarity to the entire call or to the main topic of the call, under the intuition that the notion of noteworthiness depends on the user's needs rather than on the main topic of the conversation. In addition and for robusteness purposes, we decided not to rely on long distance dependency information (*e.g.* argument predicate relations) or deep syntactical parsing, which are sensitive to the quality of the transcription.

The resulting features are grouped into three main classes: **Turn-Based** (C-T), **Dynamic** (C-D), and **Conversational** (C-C). We compare them with a pure bag-of-words (BoW) representation. Table 3a provides a summary of all the content-based features used in our system. Where applicable, we experi-

[3] All pre-processing was performed using the Freeling Language Processing tools (Padro et al., 2010).

ment with two vector representations: binary and frequency-based. We will refer to these two different encoding schemes as **Bin** for the binary case, and **Freq** for the frequency case.

CONTENT FEATURES	
C-BoW (Bag of Words)	
BoW	BoW for all words (except hapax)
C-T (Turn-based)	
NE	Presence (or frequency) of NEs (Person, Location, Organization, Numbers, Dates, Misc.)
TLN	Turn length in # words normalized
PoS	PoS distribution
TF	Max and Mean term frequency
IDF	Max and Mean inverse document frequency
C-D (Dynamic)	
Rep	Repetition between t and t-1,t+1,t-2,t+2
Int	Presence (or total amount) of Int. pro./adj. in t-1
Q	Presence (or total amount) of question in t-1
C-C (Conversational)	
Dur	Duration of the call (# turns and # words)
Cent	Conversation centrality
Spk	Speaker
Dom	Speaker dominance

(a) Content Feature

CONTEXT FEATURES	
X-C (Call-based)	
X-C-T	Time of the call
X-C-Loc	Location of the call
X-C-Day	Day of the call
X-C-Obj	Objective of the call
X-U (User-based)	
X-U-G	Gender
X-U-A	Age
X-U -I	Income
X-U-E	Education
X-U-Ms	Marital Status

(b) Context Feature

Table 3: Content (a) and Context (b) based features.

4.1.1 Turn-Based Content features (C-T)

Turn-based content features take into account information related to individual turns. We distinguish lexical and non-lexical C-T.

Lexical content features: Lexical C-T features capture the lexical properties of a turn. We include NEs, such as *Locations, Organizations, Persons*, *Miscs* and *Numbers*, *Dates*, and temporal expressions. For each turn t, we detect the presence of any NE as well as the presence of individual classes of NEs. For each of these class of entities, we extract both a binary and a frequency feature vector. In the text summarization literature, the appearance of particular lexical phrases (*e.g. to summarize*) has been exploited to predict relevant sentences (Gupta and Lehal, 2010). In our study, attention has been given to the presence of temporal expressions under the intuition that temporal cues are good indicators of upcoming pieces of information (*e.g. The meeting is tomorrow*). We exploit temporal expressions, such as *today, tomorrow, etc*.[4]

Non-lexical content features: capture characteristics of the turn which do not involve lexical information, namely: turn length, Part-of-Speech (PoS) distributions and Tf-Idf descriptive statistics at the turn level.

In meeting summarization, the average length of a turn has been found to be a good feature to automatically create a summary of a meeting (Xie et al., 2008). In our dataset, preliminary analyses revealed that annotated turns tend to be longer in average. Hence, we include the turn length in the non-lexical content feature set. The turn length is given by the number of tokens per turn normalized over the average turn

[4]Note that, here and in the remainder of the paper, we report the English translations of the Spanish originals.

length (punctuation excluded). To further gauge discourse characteristics, we detect the distribution of PoS at the turn level: *i.e.* for each turn, the frequency of nouns, pronouns, adjectives, adverbs, interjections, verbs, prepositions and conjunctions is calculated. Finally, we compute the term frequency (Tf) and inverse document frequency (Idf) measures. In (Xie et al., 2008), authors report that Idf is among the most discriminative features in sentence selection for text summarization. We compute maximum and mean Tf and Idf values for each turn.

4.1.2 Dynamic content features (C-D)

Dynamic content features are designed to capture the semantic relationships between each turn and its precedent and subsequent turns. In particular we refer to relations such as lexical and topical cohesion, question-answer relationship, and the appearance of general cues that may anticipate relevant bits of information in the subsequent turn. We consider: *1)* the lexical and topical cohesion among consecutive turns (Repetitions); *2)* the appearance of general cues that may anticipate relevant bits of information in the subsequent turn (Interrogative Pronouns); *3)* the question-answer relationship among consecutive turns (Question).

Repetitions: words repeated by different speakers in consecutive turns. Participants of a conversation tend to align at several linguistic and paralinguistic levels in order to ease communication and increase mutual understanding (Pickering and Ferreira, 2008). This phenomenon has been investigated in terms of prosody, lexicon and syntax (Levitan and Hirschberg, 2011; Brennan, 1996; Bonin et al., 2013; Branigan et al., 2010). From a lexical point of view, the alignment mechanism, often referred to as priming, is realized by means of word repetitions among speakers. Many studies have investigated this phenomenon assessing correlation between priming and mutual understanding or dialogue success (Vogel, 2013; Reitter and Moore, 2007).

We exploit the priming phenomenon to detect concepts in the conversation that are considered important by both participants, relying on the fact that repeated words convey concepts that participants want to make sure they have been successfully communicated to their interlocutor. Given a dataset D, a turn in D, $t \in D$, and $t - i$ and $t + i$ turns in the context of t, we calculate the amount of repeated lemmas between t and $t-i$, and t and $t+i$ for $1 \leq i \leq 2$. In order to consider semantically meaningful repetitions, we take into account only content words (nouns, adjectives, adverbs, verbs) when they activate one of the C-T features described above. Being A the set of annotated turns, we noticed a significant difference in the amount of repeated lemmas between t,$t - i$ for $t \in A$ rather than for $t \notin A$. Find below an example of consecutive turns with repetitions:

```
Turn Utterance
  t-1: Starting at half past four.
  t:   Starting at half past four, yes.
```

Interrogative pronouns and questions: We also exploit indicators of an upcoming *giving information* act. As shown in Sec 3.2, 47% of the annotations were marked as *giving information*, which may have been triggered by a request of information in the precedent turn. Hence, in order to capture these cases, we identify linguistic elements that indicate a request of information in $t - 1$ (questions and interrogative pronouns/adjectives).

4.1.3 Conversational flow features (C-C)

They are designed to model information about the conversation's flow and speakers' interaction.

Centrality of the turn: Distance of a turn from the center of the conversation. This feature is inspired by the sentence location features used in text summarization (Chen et al., 2002). Chen *et al.* assign different weights to sentences in the first, middle and final part of a paragraph, in order to favor sentences that are in the central part of the paragraph as they are considered to be more informative for a summary. In our corpus, we noticed the tendency of users to annotate turns that are in the central portion of the conversation. Typically the first and the last quarters of the phone conversations are dedicated to social talk. Hence, we introduce a temporal feature, referred to as conversation centrality, that captures the distance of a turn from the center of the conversation. This distance is measured in terms of number of words, excluding punctuation.

Speaker: Who is uttering the turn (caller vs callee).

Conversation duration: Length of the conversation in number of turns and in number of words. The number of turns captures the dynamics of a dialogue (few longer turn vs a more dynamic exchange), while the number of words captures the overall duration.

Speaker dominance: We consider whether the speaker is the dominant speaker of the conversation, defining dominance in terms of amount of productions during the call. This is calculated by comparing the number of turns of speaker *a* vs speaker *b*, normalized over the total amount of turns per call.

4.1.4 Bag-of-Words (BoW)

Finally, we explore the performance of a naive bag-of-words scheme to represent the content at the turn level. Given the large vocabulary size of our corpus ($10,144$ tokens) and the sparsity organic to bag-of-word representations, we decided to use a trivial dimensionality reduction strategy filtering out the terms that appear only once in the corpus. We decided not to apply a stop-list of functional words for further reducing the feature space. This decision was based on the higher discrimintative power we observed when comparing classification accuracy with and without them. We discarded the use of more aggressive feature selection approaches (*e.g.* mutual information) to allow for a fair comparison of accuracy with the rest of feature representations described in the paper. In total, our BoW representation had $5,048$ dimensions in the $\mathcal{G}$ dataset, and $3,219$ dimensions in the $\mathcal{A}$ dataset.

4.2 Context Features

Context features are introduced under the assumption that noteworthy information may depend on the characteristics of the user and on the situation in which the call takes place. For example, people may not need to annotate pieces of information that are part of their daily lives. Whereas while taking an appointment, it is plausible the need to annotate the name of the doctor, in a social call with a friend, the name of the friend is part of the background knowledge of the user. Therefore, while from a content (and an NLP) point of view both names are Person NEs and carry the same amount of information, from the point of view of the user they might have different weight (no need of taking note vs need of taking note). Also, the current situation or location of the user may influence the necessity of taking notes: a user in a supermarket will not need to annotate to buy milk, (s)he will rather take it directly from the shelf. A user driving to the supermarket will need to keep in his/her mind the need to buy milk for later recall.

In line with this, we noted in Section 2 that pure NLP approaches applied to automatically detecting noteworthy information in meetings are able to achieve an F-score of only 0.14. This low F-score underlines the complexity of the task and the limitations of a pure content-based approach. Contextual cues may be used to increase the discriminative power of the classification model.

Since we consider the specific scenario of cellphone conversations, we can exploit contextual information derived from the use of the mobile network, such as geo-location, and temporal information. Other contextual features that we use, gathered during the pre-study questionaire, are organically much more challenging to infer. We still decided to consider these as a way to assess the potential of several types of contextual information with respect to the discriminative power of the classifier. We distinguish among Call-based (X-C) and User-based (X-U) contextual features. A schematic overview of these features is given in Table 3b.

4.2.1 Call-Based Features (X-C)

Call-based features are meant to capture contextual information at the call level. In particular, X-C features include information about *where*, *when* and *for what reason* a call is made, under the intuition that calls made, for example, during working hours may have different noteworthy information than calls made in the weekend. We distinguish six *location* categories: home, work place, while commuting, while exercising, while shopping, other. The location of the calls was provided by participants through the post-call questionnaire. However location information is typically available from the mobile network. In terms of *temporal* features, we consider the actual time of the call (over 24 hours). In addition, we classify the time in two classes: working vs non working hours, and the day in also two classes: weekday vs weekend. Finally, we also consider the *objective* of the call as described in Section 3. Note that,

although this information is not directly accessible from mobile data collected during the call, previous literature on conversation classification supports the feasibility of inferring this information from the content of the conversation (Koço et al., 2012).

4.2.2 User-Based Features (X-U)

Finally, we introduce a set of features that feed the model with information about the user. We exploit information that could be provided by users upon registration to such a *note-taking* service. We capture age, gender, educational level, income and marital status. Gender is represented as a binary feature, while age is categorized in 5 groups: below 20 years old, between 20 and 30, between 30 and 40, between 40 and 50 and above 50. The education status is represented by the following categories: Primary education, Secondary education, Bachelor degree or a Postgraduate education (Master or PhD). Yearly income is categorized by: up to 10k, 20k, 30k, 40k and more than 40k. Finally, marital status is categorized as: single, in a couple (married, with a stable partner), other.

5 Experiments

The goal of our system is to automatically identify information annotated by users in terms of its potential need for future recall. We frame this problem as a binary classification task (noteworthy or not) at the turn level. This task presents two main challenges. First, our dataset is extremely unbalanced, with less than 3% of the corpus labeled as relevant by the participants. Second, the subjectivity of the task leads to high variability of annotation behaviours, (see Sec. 3.2). In this section we describe the experimental setting that we used to empirically evaluate the performance of different features sets and present the results obtained using the ground truth data collected (Section 3) to provide classification performance scores. In order to fully investigate the predictive performance of the different feature sets, we conducted our experiments using both the entire corpus $\mathcal{G}$, which includes all the selected conversations, and its subset $\mathcal{A}$, which considers only the calls with at least one annotation. Both sets are described in Sec. 3. We experimented using both encoding schemes described in Section 4: binary based (**Bin**) and frequency based (**Freq**).

We used Support Vector Machines (SVMs) with RBF kernel, as this classification approach yielded the most consistent results throughout all the evaluated configurations. We used the same random split of training and test sets for all the experiments, accounting for 70% and 30% of the dataset respectively. We tune the hyperparameter C of the SVM model using a 3-fold cross-validation approach on the training data only, where we chose F-score as the quality metric to optimize. Given the nature of our task, recall is preferred to precision from a user-centric perspective: it is preferable to avoid missing any relevant information than to include some non-relevant fragments. For this reason, we also report precision and recall values.

5.1 Classification Results

This section presents the results obtained in our binary classification task (turns being noteworthy or not). We study the performance of different combinations of features and present the results obtained using only content information (C), and the combination of content and context information (CX).

5.1.1 Content features

We present a comparison of the different content feature sets using the naming scheme of Sec.4. We considered four classification scenarios: C-T only, C-D only, the combination of C-T and C-D (C-TD), and the combination of C-T, C-D and C-C (C-TDC). The results of these feature sets are shown in Tables 4a and 4b for the $\mathcal{G}$ and $\mathcal{A}$ collections, respectively.

As shown in Table 4a, the maximum F-score for the $\mathcal{G}$ dataset is achieved for the combination of all content features included the BoW. The low score ($F = 0.18$) is a direct consequence of the low precision obtained ($P = 0.11$). For the $\mathcal{A}$ dataset (Table 4b) we observe a better F-score ($F = 0.296$), still obtained by the combination of all content features, with a much higher precision ($P = 0.18$) due to the significant amount of noise removed by considering only annotated calls. Note that in both the $\mathcal{G}$ and the $\mathcal{A}$ datasets the C-TDC feature set outperforms the pure BoW approach ($F = 0.158$ vs. $F = 0.14$

Features	Precision		Recall		F-score	
Encoding	Bin	Freq	Bin	Freq	Bin	Freq
BoW	0.081	0.083	0.730	0.720	0.150	0.150
C-T	0.087	0.088	0.53	0.32	0.15	0.139
C-D	0.03	0.26	0.26	0.12	0.15	0.05
C-TD	0.087	0.09	**0.754**	0.33	0.1505	0.1419
C-TDC	0.09	0.093	0.58	0.37	0.158	0.149
C-TDC+BoW	**0.11**	**0.11**	0.52	0.51	**0.18**	**0.18**

(a) Results for the $\mathcal{G}$ dataset.

Features	Precision		Recall		F-score	
Encoding	Bin	Freq	Bin	Freq	Bin	Freq
BoW	0.14	0.14	0.54	0.54	0.22	0.22
C-T	0.135	0.147	0.56	0.555	0.218	0.23
C-D	0.149	0.11	0.24	0.15	0.18	0.12
C-TD	0.143	0.143	0.506	0.546	0.223	0.227
C-TDC	0.165	0.159	0.626	**0.693**	0.254	0.267
C-TDC+BoW	**0.188**	0.1659	0.57	0.568	0.283	**0.296**

(b) Results for the $\mathcal{A}$ dataset.

Table 4: Classification performance of Content features, BoW and their combination.

for $\mathcal{G}$ and $F = 0.267$ vs. $F = 0.22$ for $\mathcal{A}$), using a fraction (about 1%) of the number of BoW features, which leads to a considerably simpler model. On the other hand, the combination of C-TDC and BoW features improves the results up to $F = 0.18$ for $\mathcal{G}$ ($P = 0.11$, $R = 0.52$) and $F = 0.296$ ($P = 0.20$, $R = 0.57$) for the $\mathcal{A}$ subset. This result highlights how the lexical representation comprised by the BoW provides the model with orthogonal information to the one provided by the C-TDC features set.

To the best of our knowledge no previous work has been done in noteworthy detection from telephone conversations. For this reason, we report as a reference the results of the more similar prior art to our work, (Banerjee and Rudnicky, 2008), where the authors implement an SVM classifier for the detection of noteworthy information in meetings.[5] Although aware of the different nature of the dataset, these results are reported to get a sense of the potentiality of the system. The best performance of our model on the $\mathcal{A}$ dataset improves in 15% the F-score of $F = 0.14$ reported in (Banerjee and Rudnicky, 2008).

5.1.2 Combining Content and Context Features

In this section we report the performance of the model trained using both content and context features. For simplicity, in the remainder of this section we refer to the entire set of content features, (C-TDC) as C, to the entire set of context features as X, and to their combination as CX. When we test adding BoW features the *+BoW* naming is used. The results are shown in Table 5 and Figure 1. We observe that the fusion of content and context features (CX and CX+BoW) provides a noticeable overall increase in the F-score for both datasets. This increase is particularly high for the $\mathcal{G}$ dataset, where the F-score gets increased by almost a factor of 2, from $F = 0.18$ to $F = 0.28$. On the $\mathcal{A}$ dataset, the combination of content and context features improves the F-score from $F = 0.29$ to $F = 0.32$, given by a better precision ($P = 0.24$ vs $P = 0.18$) with similar recall.

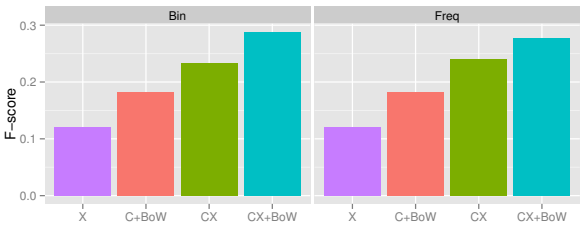
Features	Precision		Recall		F-score	
Rep.	B	F	B	F	B	F
C+BoW	0.11	0.11	0.52	0.51	0.18	0.18
X	0.068	0.068	**0.665**	**0.665**	0.124	0.124
CX	0.169	**0.20**	0.38	0.286	0.2354	0.2394
CX+BoW	0.189	0.1919	0.524	0.5022	**0.288**	0.277

(a) Results for the $\mathcal{G}$ dataset.

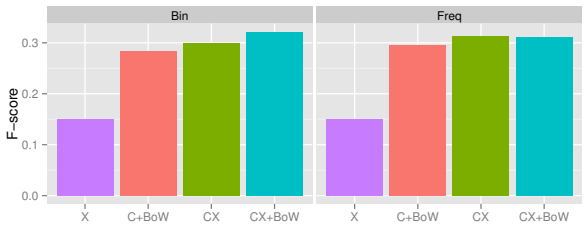
Features	Precision		Recall		F-score	
Rep.	B	F	B	F	B	F
C+BoW	0.188	0.1659	0.57	0.568	0.283	0.296
X	0.087	0.087	0.56	0.56	0.15	0.15
CX	0.2075	0.212	0.5866	**0.595**	0.3066	0.3130
CX+BoW	0.223	**0.2455**	0.573	0.426	**0.3212**	0.3116

(b) Results for the $\mathcal{A}$ dataset.

Table 5: Classification performance using the combination of context and context-based features



(a) Results for the $\mathcal{G}$ dataset.



(b) Results for the $\mathcal{A}$ dataset.

Figure 1: Classification performance using Content, Context features and their combination

This result gives empirical evidence that these two sets of features convey complementary information

[5] In their experimental settings all the meetings have at least one annotation as in our $\mathcal{A}$ scenario.

that is relevant for the task at hand. That is, the same words can carry different relevance depending on the contextual information of the conversation.

Note that the BoW features add discriminative information in the $\mathcal{G}$ scenario, but have a minimal effect in the less noisy $\mathcal{A}$ scenario where the combination of content and context features, without BoW, provides already an F-score of $F = 0.31$.

An interesting remark about this combined model is that the difference in performance between the $\mathcal{G}$ and the $\mathcal{A}$ dataset is vastly reduced. While in the pure content model the difference in F-score value between both datasets was 0.10, in the combined model this difference is just 0.03. This result shows that the combination of content and context features boosts considerably the results in the more noisy and realistic dataset $\mathcal{G}$, while its effect is weaker in the cleaner dataset.

5.2 Qualitative Analysis

In order to better understand the failure cases in our system, we carried out a qualitative analysis of both false positives, *i.e.* turns annotated by the system but not by the user, and false negatives, *i.e.* turns annotated by the user but not by the system. Table 6 illustrates a few representative examples. Note how the proposed system does not perform well when detecting a *request for information* as something worth annotating (*e.g. What are you doing?*). We noticed that in these cases, the model tended to annotate turns where the information was actually provided (*e.g.That is the package has arrived*). We can hypothesize that users annotate the *request for information* to give context to the a-priori more relevant information, *i.e.* the answer to the question. However, in some cases, participants did not annotate the answer as relevant. This counter-intuitive observation reflects the subjectivity and variability of the task.

False Positive	False Negative
I am leaving soon, I start at 3 o'clock or [..]	*How are you? Can you hear me?*
Let's see if we can tell him.	*What are you doing?*
That is, the package has arrived.	*Did you buy beautiful things for me?*

Table 6: Examples of false positive and false negative turns.

5.3 Comparative Analysis and Discussion

To the best of our knowledge there are no previous works of similar nature to the study presented in this paper. Yet, it is important to give a sense of the merits and limitations of the proposed approach in the context of the state-of-the-art. For this reason, we compare our results with (Banerjee and Rudnicky, 2009), which is the most similar prior art to our work. In (Banerjee and Rudnicky, 2009), Barnejee *et al.* perform a Wizard-of-Oz experiment and report a performance of the human annotator of $P = 0.35$ precision, $R = 0.42$ recall, leading to an F-score of $F = 0.38$. This result highlights the difficulty of the task, even for a human annotator. When comparing our proposed system with this Wizard-of-Oz experiment, we obtain an F-score of $F = 0.32$ against the human annotator's F-score of $F = 0.38$, with a significantly higher recall (0.57 vs 0.42) yet lower precision (0.245 vs 0.35). Given this human-based prediction performance, the proposed approach represents a good first step towards realizing an intelligent annotation system for mobile phone conversations.

6 Conclusions and Future Work

In this paper we have proposed and empirically evaluated a machine learning-based approach to automatically detect noteworthy information in spontaneous mobile phone conversations. The subjectivity of this task leads to a challenging classification problem even for human assessors. Our approach adopts a hybrid strategy that exploits the content and the context of the conversation. We have shown that information about the context of the conversation improves the predictive performance of the system over a pure content based approach.

In the future, we plan to extend the model by including acoustic features which could improve the performance by adding orthogonal information to the current model. To tackle the subjectivity of the task we also intend to investigate the performance of personalization techniques, creating individual models per user. Finally, we plan to conduct a study to evaluate our system from a user-centric perspective.

Acknowledgments

This work was partially supported by the Innovation Bursary of Trinity College Dublin (project: 'Technology for Harmonising Interpersonal Communication').

References

Satanjeev Banerjee and Alexander I. Rudnicky. 2008. An extractive-summarization baseline for the automatic detection of noteworthy utterances in multi-party human-human dialog. In *Proceeding of SLT*, pages 177–180.

Satanjeev Banerjee and Alexander Rudnicky. 2009. Detecting the noteworthiness of utterances in human meetings. In *Proceedings of the SIGDIAL 2009 Conference*, pages 71–78, London, UK, September. Association for Computational Linguistics.

Francesca Bonin, Celine De Looze, Sucheta Ghosh, Emer Gilmartin, Carl Vogel, Anna Polychroniou, Hugues Salamin, Alessandro Vinciarelli, and Nick Campbell. 2013. Investigating fine temporal dynamics of prosodic and lexical accommodation. In *Proceedings of Interspeech 2013*, Lyon, France, August.

H.P. Branigan, M.J. Pickering, J. Pearson, and J.F. McLean. 2010. Linguistic alignment between people and computers. *Journal of Pragmatics*, 42(9):2355–2368.

Susan E. Brennan. 1996. Lexical entrainment in spontaneous dialog. *Proceedings of ISSD*, pages 41–44.

Juan Pablo Carrascal, Rodrigo de Oliveira, and Mauro Cherubini. 2012. A note paper on note-taking: understanding annotations of mobile phone calls. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, MobileHCI '12, pages 21–24, New York, NY, USA. ACM.

Fang Chen, Kesong Han, and Guilin Chen. 2002. An approach to sentence-selection-based text summarization. In *TENCON'02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering*, volume 1, pages 489–493. IEEE.

Chandrika Cycil, Mark Perry, Eric Laurier, and Alex Taylor. 2013. 'eyes free' in-car assistance: parent and child passenger collaboration during phone calls. In *Proceedings of the 15th international conference on Human-computer interaction with mobile devices and services*, MobileHCI '13, pages 332–341, New York, NY, USA. ACM.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 364–372, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nikhil Garg, Benoit Favre, and Dilek Hakkani-Tür. 2009. Clusterrank: a graph based method for meeting summarization. In *Technical Report, IDIAP*.

Vishal Gupta and Gurpreet Singh Lehal. 2010. A survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence*, 2(3):258–268.

Po Hu, Dong-Hong Ji, Chong Teng, and Yujing Guo. 2012. Context-enhanced personalized social summarization. In *COLING*, pages 1223–1238.

J Jian Zhang, Ho Yin Chan, and Pascale Fung. 2007. Improving lecture speech summarization using rhetorical information. In *Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on*, pages 195–200. IEEE.

Sokol Koço, Cécile Capponi, and Frédéric Béchet. 2012. Applying multiview learning algorithms to human-human conversation classification. In *INTERSPEECH*.

Rivka Levitan and Julia Hirschberg. 2011. Measuring acoustic-prosodic entrainment with respect to multiple levels and dimensions. In *Twelfth Annual Conference of the International Speech Communication Association*.

Sameer Maskey and Julia Hirschberg. 2003. Automatic summarization of broadcast news using structural features. In *INTERSPEECH*.

Sameer Maskey and Julia Hirschberg. 2005. Comparing lexical, acoustic/prosodic, structural and discourse features for speech summarization. In *INTERSPEECH*, pages 621–624.

Sameer Maskey and Julia Hirschberg. 2006. Summarizing speech without text using hidden markov models. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 89–92. Association for Computational Linguistics.

Lluis Padro, Samuel Reese, Eneko Agirre, and Aitor Soroa. 2010. Semantic services in freeling 2.1: Wordnet and ukb. In Pushpak Bhattacharyya, Christiane Fellbaum, and Piek Vossen, editors, *Principles, Construction, and Application of Multilingual Wordnets*, pages 99–105, Mumbai, India, February. Global Wordnet Conference 2010, Narosa Publishing House.

Martin J. Pickering and Victor S. Ferreira. 2008. Structural priming: a critical review. *Psychological bulletin*, 134(3):427.

David Reitter and Johanna D Moore. 2007. Predicting success in dialogue. In *Annual Meeting-Association for Computational Linguistics*, volume 45, page 808.

Jose San Pedro, Vaiva Kalnikaite, and Steve Whittaker. 2009. You can play that again: Exploring social redundancy to derive highlight regions in videos. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pages 469–474, New York, NY, USA. ACM.

Carl Vogel. 2013. Attribution of mutual understanding. *Journal of Law & Policy*, pages 101–145.

Dong Wang and Yang Liu. 2011. A pilot study of opinion summarization in conversations. In *Annual Meeting-Association for Computational Linguistics, ACL*, pages 331–339.

Shasha Xie, Yang Liu, and Hui Lin. 2008. Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*, pages 157–160. IEEE.

Zi Yang, Keke Cai, Jie Tang, Li Zhang, Zhong Su, and Juanzi Li. 2011. Social context summarization. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 255–264, New York, NY, USA. ACM.

Tom Yeh, Brandyn White, Jose San Pedro, Boriz Katz, and Larry S. Davis. 2011. A case for query by image and text content: Searching computer help using screenshots and keywords. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 775–784, New York, NY, USA. ACM.

Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. 2005. Narrative text classification for automatic key phrase extraction in web document corpora. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*, WIDM '05, pages 51–58, New York, NY, USA. ACM.

Hierarchical Topical Segmentation with Affinity Propagation

Anna Kazantseva & Stan Szpakowicz
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Ontario, Canada
{ankazant,szpak}@eecs.uottawa.ca

Abstract

We present a hierarchical topical segmenter for free text. Hierarchical Affinity Propagation for Segmentation (*HAPS*) is derived from a clustering algorithm Affinity Propagation. Given a document, *HAPS* builds a topical tree. The nodes at the top level correspond to the most prominent shifts of topic in the document. Nodes at lower levels correspond to finer topical fluctuations. For each segment in the tree, *HAPS* identifies a segment centre – a sentence or a paragraph which best describes its contents. We evaluate the segmenter on a subset of a novel manually segmented by several annotators, and on a dataset of Wikipedia articles. The results suggest that hierarchical segmentations produced by *HAPS* are better than those obtained by iteratively running several one-level segmenters. An additional advantage of *HAPS* is that it does not require the "gold standard" number of segments in advance.

1 Introduction

When an NLP application works with a document, it may benefit from knowing something about this document's high-level structure. Text summarization (Haghighi and Vanderwende, 2009), question answering (Oh et al., 2007) and information retrieval (Ponte and Croft, 1998) are some of the examples of such applications. Topical segmentation is a lightweight form of such structural analysis: given a sequence of sentences or paragraphs, split it into a sequence of *topical segments*, each characterized by a certain degree of topical unity. This is particularly useful for texts with little structure imposed by the author, such as speech transcripts, meeting notes or literature.

The past decade has witnessed significant progress in the area of text segmentation. Most of the topical segmenters (Malioutov and Barzilay, 2006; Eisenstein and Barzilay, 2008; Kazantseva and Szpakowicz, 2011; Misra et al., 2011; Du et al., 2013) can only produce single-level segmentation, a worthy endeavour in and of itself. Yet, to view the structure of a document linearly, as a sequence of segments, is in certain discord with most theories of discourse structure, where it is more customary to consider documents as trees (Mann and Thompson, 1988; Marcu, 2000; Hernault et al., 2010; Feng and Hirst, 2012) or graphs (Wolf and Gibson, 2006). Regardless of the theory, we hypothesize that it may be useful to have an idea about fluctuations of topic in documents beyond the coarsest level. It is the contribution of this work that we develop such a hierarchical segmenter, implement it and do our best to evaluate it.

The segmenter described here is *HAPS* – Hierarchical Affinity Propagation for Segmentation. It is closely based on a graphical model for hierarchical clustering called *Hierarchical Affinity Propagation* (Givoni et al., 2011). It is a similarity-based segmenter. It takes as input a matrix of similarities between atomic units of text in the sequence to be segmented (sentences or paragraphs), the desired number of levels in the topical tree and a preference value for each data point and each level. This value captures *a priori* belief about how likely it is that this data point is a segment centre at that level. The preference values also control the granularity of segmentation: how many segments are to be identified at each level. The output is a topical tree. For each segment at every level, *HAPS* also finds a segment centre, a data point which best describes the segment.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 37–47, Dublin, Ireland, August 23-29 2014.

The objective function maximized by the segmenter is net similarity – the sum of similarities between all segment centres and their children for all levels of the tree. This function is similar to the objective function of the well-known *k-means* algorithm, except that here it is computed hierarchically.

It is not easy to evaluate *HAPS*. We are not aware of comparable hierarchical segmenters other than that in (Eisenstein, 2009) which, unfortunately, is no longer publicly available. Therefore we compared the trees built by *HAPS* to the results of running iteratively two state-of-the-art flat segmenters. The results are compared on two datasets. A set of Wikipedia articles was automatically compiled by Carroll (2010). The other set, created to evaluate *HAPS*, consists of nine chapters from the novel *Moonstone* by Wilkie Collins. Each chapter was annotated for hierarchical structure by 3-6 people.

The evaluation is based on two metrics, *windowDiff* (Pevzner and Hearst, 2002) and *evalHDS* (Carroll, 2010). Both metrics are less then ideal. They do not give a complete picture of the quality of topical segmentations, but the preliminary results suggest that running a global model for hierarchical segmentation produces better results then iteratively running flat segmenters. Compared to the baseline segmenters, *HAPS* has an important practical advantage. It does not require the number of segments as an input; this requirement is customary for most flat segmenters.

We also made a rough attempt to evaluate the quality of the segment centres identified by *HAPS*. Using 20 chapters from several novels of Jane Austen, we compared the centres identified for each chapter against summaries produces by a recent automatic summarizer *CohSum* (Smith et al., 2012). The basis of comparison was the ROUGE metric (Lin, 2004). While far from conclusive, the results suggest that segment centres identified by *HAPS* are rather comparable with the summaries produced by an automatic summarizer.

A Java implementation of *HAPS* and the corpus of hierarchical segmentations for nine chapters of *Moonstone* are publicly available. We consider these to be the main contributions of this research.

2 Related work

Most work on topical text segmentation has been done for single-level segmentation. Contemporary approaches usually rely on the idea that topic shifts can be identified by finding shifts in the vocabulary (Youmans, 1991). We can distinguish between local and global models for topical text segmentation. Local algorithms have a limited view of the document. For example, TextTiling (Hearst, 1997) operates by sliding a window through the input sequence and computing similarity between adjacent units. By identifying "valleys" in similarities, TextTiling identifies topic shifts. More recently, Marathe (2010) used lexical chains and Blei and Moreno (2001) used Hidden Markov Models. Such methods are usually very fast, but can be thrown off by small digressions in the text.

Among global algorithms, we can distinguish generative probabilistic models and similarity-based models. Eisenstein and Barzilay (2008) model a document as a sequence of segments generated by latent topic variables. Misra et al. (2011) and Du et al. (2013) have similar models. Malioutov and Barzilay (2006) and (Kazantseva and Szpakowicz, 2011) use similarity-based representations. Both algorithms take as input a matrix of similarities between sentences of the input document; the former uses graph cuts to find cohesive segments, while the latter modifies a clustering algorithm to perform segmentation.

Research on hierarchical segmentation has been more scarce. Yaari (1997) produced hierarchical segmentation by agglomerative clustering. Eisenstein (2009) used a Bayesian model to create topical trees, but the system is regrettably no longer publicly available. Song et al. (2011) develop an algorithm for hierarchical segmentation which iteratively splits a document in two at a place where cohesion links are the weakest. A second pass transforms a deep binary tree into a shallow and broad structure.

Any flat segmenter can certainly be used iteratively to create trees of segments by subdividing each segment, but this may be problematic. Topical segmenters are not perfect, so running them iteratively is likely to compound the error. Most segmenters also require the number of segments as an input. This estimate is feasible for flat segmentation. To know in advance the number of segments and sub-segments at each level is not a realistic requirement when building a tree.

This work describes a hierarchical model of text segmentation. It takes a global view of the document and of the topical hierarchy. Each iteration attempts to find the best assignment of segments for the

whole tree. It does not need to know the exact number of segments. Instead, it takes a more abstract parameter, preference values, to specify the granularity of segmentation at each level. For each segment it also outputs a segment centre, a unit of text which best captures the contents of the segment.

3 Creating a corpus of hierarchical segmentations

Before embarking on the task of building a hierarchical segmenter, we wanted to study how people perform such a task. We also needed a benchmark corpus which could be used to evaluate the quality of segmentations produced by *HAPS*.

To this end, we annotated nine chapters of the novel *Moonstone* for hierarchical structure. We settled on these data because it is a subset of a publicly available dataset for flat segmentation (Kazantseva and Szpakowicz, 2012). In our study, each chapter was annotated by 3-6 people (4.8 on average). The annotators, undergraduate students of English, were paid $50 dollars each.

Procedure. The instructions asked the annotator to read the chapter and split it into top-level segments according to where there is a perceptible shift of topic. She had to provide a one-sentence description of what the segment is about. The procedure had to be repeated for each segment all the way down to the level of individual paragraphs. Effectively, the annotators were building a detailed hierarchical outline for each chapter.

Metrics. Two different metrics helped estimate the quality of our hierarchical dataset: *windowDiff* (Pevzner and Hearst, 2002) and *S* (Fournier and Inkpen, 2012).

windowDiff is computed by sliding a window across the input sequence and checking, for each window position, whether the number of reference breaks is the same as the number of breaks in the hypothetical segmentation. The number of erroneous windows is then normalized by the total number of windows. In Equation 1, N is the length of the input sequence and k is the size of the sliding window.

$$windowDiff \;=\; \frac{1}{N-k}\sum_{i=1}^{N-k}(|ref - hyp| \neq 0) \tag{1}$$

windowDiff is designed to compare sequences of segments, not trees. That is why we compute it for each level between each pair of annotators who worked on the same chapter. It should be noted that *windowDiff* is a penalty metric: higher values indicate less agreement (*windowDiff* $= 0$ corresponds to two identical segmentations).

The *S* metric allows us to compare trees and take into account situations when the segmenter places a boundary at a correct position but at a wrong level. *S* is an edit-distance metric. It computes the number of operations necessary to turn one segmentation into another. There are three types of editing operations: add/delete, transpose and substitute (change the level in the tree). The sum is normalized by the number of possible boundaries in the sequence. *S* has an unfortunate downside of being too optimistic, but it allows the breakdown of error types and it explicitly compares trees.

Unlike *windowDiff*, *S* is a similarity metric: higher values correspond to more similar segmentations. The value of *S* between two identical segmentations is 1.

$$S(bs_a, bs_b, n) = \frac{1 - |boundary\_distance(bs_a, bs_b, n)|}{pb(D)} \tag{2}$$

Here $boundary\_distance(bs_a, bs_b, n)$ is the total number of edit operations needed to turn a segmentation bs_a into bs_b, n is the threshold defining the maximum distance of transpositions. $pb(D)$ is the maximum possible number of edits. Segmentations bs_a and bs_a are represented as strings of sets of boundary positions. For example bs_a = ({2}, {1,2}, {1,2}) corresponds to a hierarchical segmentation of a three-unit sequence in the following manner: a segment boundary at level 1 after the first unit, segment boundaries at levels 1 and 2 after the second unit and the third unit.

Corpus Analysis. On average, the annotators took 3.5 hours to complete the task (σ = 1.6). The average depth of the tree is 3.00 levels (σ = 0.65), suggesting that the annotators prefer shallow but broad structures. Table 1 reports the average breadth of the tree at different levels. In the Table and further

in this paper we refer to the bottom level of the tree (i.e., the leaves of the tree or the most fine-grained level of segmentation) as level 1. In Table 1, level 4 refers to the top level of the tree (the coarsest segmentations). The values were computed using only the breaks explicitly specified by the annotators (*i.e.*, we did not assume that a break at a coarse level implies a break at a more detailed level).

The average breadth of the trees at the bottom (level 1) is lower than that at level 2, indicating that only a small percentage of the entire tree was annotated more than three levels deep. The table also shows the average values of *windowDiff* computed for each possible pair of annotators. The values worsen toward the bottom of the tree, suggesting that the annotators agree more about top-level segments and less and less about finer fluctuations of topic.

We hypothesize that these shallow broad structures are due to the fact that it is difficult for people to create deep recursive structures in their mental representations. We do not, however, have any hard data to support this hypothesis. Many of the annotators specifically commented on the difficulty of the task. 9 out of 23 people included comments ranging from notes about specific places to general comments about their lack of confidence. 4 annotators found several (specific) passages they had trouble with.

The average value of pairwise *S* is 0.79. We have noted earlier that the *S* metric tends to be optimistic (that is due to its normalization factor) but it provides a breakdown of disagreements between the annotators. According to *S*, 46.14% of disagreements are errors of omission (some of the annotators did not include segment breaks where others did), 47.56% are disagreements about the level of segmentation (the annotators placed boundaries in the same place but at different levels) and only 6.31% are errors of transposition (the annotators do not agree about the exact placement but place boundaries within 1 position of each other). This distribution is more interesting than the overall value of *S*. Among other things, it shows why it is so important to take into account adjacent levels when evaluating topical trees.

4 The *HAPS* algorithm[1]

4.1 Factor graphs

The *HAPS* segmenter is based on factor graphs, a unifying formalism for such graphical models as Markov or Bayesian networks. A factor graph is a bi-partite graph with two types of nodes, *factor* or *function nodes* and *variable nodes*. Each factor node is connected to those variable nodes which are its arguments. Running the well-known Max-Sum algorithm (Bishop, 2006) on a factor graph finds a configuration of variables which maximizes the sum of all component functions. This is a message-passing algorithm. All variable nodes send messages to their factor neighbours (functions in which those nodes are variables) and all factor nodes send messages to their variable neighbours (their arguments). A message $\mu_{x\to f}$ sent from a variable node x to a function node f is computed as a sum of all incoming messages to x, except the message from the recipient function f:

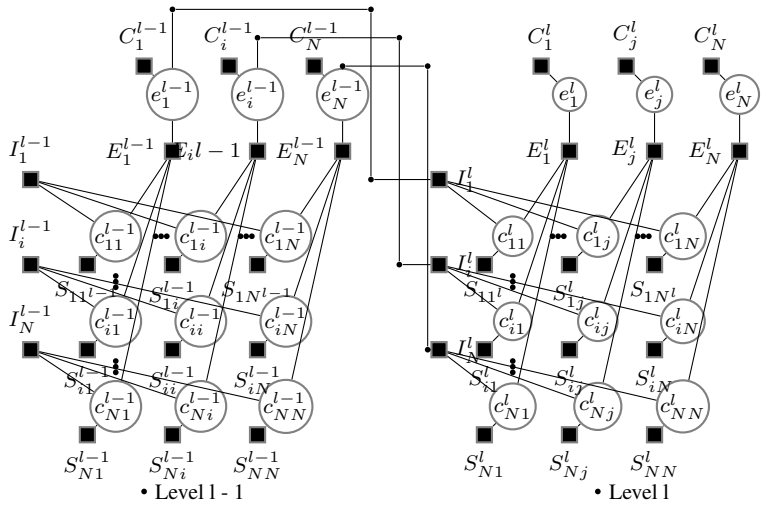
$$\mu_{x\to f} = \sum_{f'\in N(x)\setminus f} \mu_{f'\to x} \tag{3}$$

$N(x)$ is the set of all function nodes which are x's neighbours. Intuitively, the message reflects evidence about the distribution of x from all functions which have x as an argument, except the function corresponding to the receiving node f. A message $\mu_{f(x,\ldots)\to x}$ sent from the factor node $f(x,\ldots)$ to the

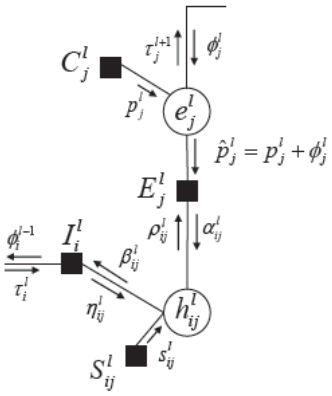
[1]The derivation of the *HAPS* algorithm, quite involved, is unlikely to interest many readers. We only present the bare minimum of facts about the algorithm, the framework of factor graphs and the derivation of *HAPS* from the underlying model of Affinity Propagation. A detailed account appears in (Kazantseva, 2014).

Table 1: Average breadth of manually created topical trees and *windowDiff* value across different levels

Level	Average breadth	*windowDiff*
4 (top)	6.53	0.35
3	17.55	0.46
2	17.63	0.47
1 (bottom)	8.80	0.50



(a) Fragment of the factor graph for levels $l-1$ and l



(b) Types of messages sent in the *HAPS* model

Figure 1: Factor graph for *HAPS* – Hierarchical Affinity Propagation for Segmentation

variable node x is computed as a maximum of the value of $f(x)$ plus all messages incoming to $f(x, ...)$ other than the message from the recipient node x:

$$\mu_{f \to x} = \max_{N(f) \setminus x} \left(f(x_1, \ldots, x_m) + \sum_{x' \in N(f) \setminus x} \mu_{x' \to f} \right) \tag{4}$$

$N(f)$ is the set of all variable nodes which are f's neighbours. The message reflects the evidence about the distribution of x from function f and its neighbours other than x.

4.2 Hierarchical Affinity Propagation for Segmentation

This work aims to build trees of topical segments. Each segment is characterized by a centre which best describes its content. The objective function is net similarity, the sum of similarities between all centres and the data points which they exemplify. The complete sequence of data points is to be segmented at each level of the tree, subject to the following constraint: centres at each level l, $l > 1$, must be a subset of the centres from the previous level $l-1$. Figure 1a shows a fragment of the factor graph describing *HAPS* corresponding to levels l and $l-1$. The tree has L levels, from the root ($l = L$) down to the leaves ($l = 1$). The superscripts of factor and variable nodes denote the level.

At each level, there are N^2 variable nodes c_{ij}^l and N variable nodes e_j^l (N is the number of data points in the sequence to segment). A variable's value is 0 or 1: $c_{ij}^l = 1 \Leftrightarrow$ the data point i at level l belongs to the segment centred around data point j; $e_j^l = 1 \Leftrightarrow$ there is a segment centred around j at level l.

Four types of factor nodes in Figure 1a are I, E, C and S. The I factors ensure that each data point is assigned to exactly one segment *and* that segment centres at level l are a subset of those from level $l-1$. The E nodes ensure that segments are centred around the segment centres in solid blocks (rather than unordered clusters). The values of I and E are 0 for valid configurations and -∞ otherwise. The S factors capture similarities between data points. $S_{ij}^l = sim(i,j)$ if $c_{ij}^l = 1$; $S_{ij}^l = 0$ if $c_{ij}^l = 0$.[2] The C factors handle preferences in an analogous manner. Running the Max-Sum algorithm on the factor graph in Figure 1a maximizes the net similarity between all segment centres and their children at all levels:

$$\max_{\{c_{ij}^l\},\{e_j^l\}} S(\{c_{ij}^l\},\{e_j^l\}) = \sum_{i,j,l} S_{i,j}^l(c_{ij}^l) + \sum_{i,l} I_i^l(c_{i1}^l, \ldots, c_{iN}^l, e_i^{l-1}) + \sum_{j,l} E_j^l(c_{1j}^l, \ldots, c_{Nj}^l, e_j^l) + \sum_{j,l} C_j^l(e_j^l) \tag{5}$$

[2]The value $sim(i,j)$ is specified in the input matrix.

Figure 1b shows a close-up view of the messages that must be sent to find the optimizing configuration of variables. Messages β, η, $\hat{\rho}$ do not need to be sent explicitly: their values are subsumed by other types of messages. We only need to compute explicitly and send four types of messages: α, ρ, ϕ and τ.

Algorithm 1 shows the pseudo-code for the *HAPS* algorithm.[3] Intuitively, different parts of the update messages in Algorithm 1 correspond to likelihood ratios between two hypotheses: whether a data point i is or is not part of a segment centred around another data point j at a given level l. For example, here is the availability (α) message sent from a potential segment centre j to itself at level l:

$$\alpha_{ij}^{l} = p_j^l + \phi_j^l + \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}^l) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}^l) \tag{6}$$

Here p_j^l incorporates the information about the preference value for the data point j at the level l. ϕ_j^l brings in the information from the coarser level of the tree. The summand $\max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}^l)$ encodes the likelihood that there is a segment starting before j given the values of responsibility messages for all data points i such that $i < j$ — hence the information from a more detailed level of the tree as well as the similarities between all data points i ($i < j$) and j. The summand $\max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}^l)$ does the same for the tail-end of the segment (all data points i such that $i > j$).

Complexity analysis. The *HAPS* model contains N^2 c_{ij}^l nodes at each level. In practice, however, the matrix of similarities $\mathcal{SIM}$ does not need to be fully specified. It is customary to compute this matrix with a large sliding window; the size should be at least twice the anticipated average length. On each iteration, we need to send *L*M*N* messages α and ρ, resulting in the complexity *O(L*M*N)*. Here L is the number of levels, N is the number of data points in the sequence and M ($M \leq N$) is the size of the sliding window used for computing similarities. The computation of ρ and α messages is independent for each row and column respectively, so the algorithm would be easy to parallelize.

Parameter settings. An important advantage of *HAPS* is that it does not require the number of segments in advance. Instead, the user needs to set the preference values for each level. However, *HAPS* is fairly resistant to changes in preferences and this generic parameter is a convenient knob for fine-tuning the desired granularity of segmentation, as opposed to specifying the exact number of segments at each level of the tree. In this work we set preferences uniformly, but it is possible to incorporate additional knowledge through more discriminative settings.

In all our experiments, preference values are set uniformly for each level of the tree, so effectively all data points are equally likely to be chosen as segment centres at each level. As a starting point, the preference value for the most detailed level of the tree should be about approximately equal to the median similarity value (as specified in the input matrix). A near-zero preference value tends to result in a medium number of segments and is thus suitable to the middle levels of the tree. A negative preference value results in a small number of segments and is appropriate for identifying the most pronounced segment breaks.

5 Experimental evaluation

In order to evaluate the quality of topical trees produced by *HAPS*, we ran the system on two datasets. We compared the results obtained by *HAPS* against topical trees obtained by iteratively running two high-performance single-level segmenters.

Datasets. We used the *Moonstone* corpus described in Section 2, and the Wikipedia dataset compiled by Carroll (2010). Created automatically from metadata on Web pages, the dataset consists of 66 Wikipedia entries on various topics; the annotations and the results concern sentences. In the *Moonstone* corpus we work with paragraphs. To simplify evaluation and interpretation, we produced three-tier trees. This is in line with the average depths of manual annotations in the *Moonstone* data.

[3] It is not possible to include a detailed derivation of the new update messages in the space allowed here. The interested reader can find these details in (Kazantseva, 2014). The derivation follows the same logic as (Givoni et al., 2011) and (Kazantseva and Szpakowicz, 2011).

Algorithm 1 Hierarchical Affinity Propagation for Segmentation

input: 1) L pairwise similarity matrices $\{\mathcal{SIM}^l(i,j)\}_{(i,j)\in\{1,\ldots,N\}^2}$; 2) L preferences p^l (one per level l) indicating *a priori* likelihood of point i being a segment centre at level l

initialization: $\forall i,j : \alpha_{ij} = 0$ (set all availabilities to 0)

repeat

iteratively update ρ, α, ϕ and τ messages

$$\forall i,l : \phi_i^{l-1} = \max[0, \alpha_{ii} - \max_{k\neq i}(s_{ik}^l + \alpha_{ik}^l)]$$

$$\forall i,j,l : \rho_{ij}^l = \begin{cases} \min(0, \tau_i^l) - \max_{k\neq i}(s_{ik}^l + \alpha_{ik}^l) & \text{if } i = j \\ s_{ij}^l + \min[\max(0, -\tau_i^l) - \alpha_{ii}^l, -\max_{k\notin i,j}(s_{ik}^l + \alpha_{ik}^l)] & \text{if } i \neq j \end{cases}$$

$$\forall i,j,l : \alpha_{ij}^l = \begin{cases} p_j^l + \phi_j^l + \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}^l) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}^l) & \text{if } i = j \\ \alpha_{ij,i<j}^l = \min[(\max_{s=1}^{i}\sum_{k=s}^{i-1}\rho_{kj}^l + \sum_{k=i+1}^{j}\rho_{kj}^l + \max_{e=j}^{N}\sum_{k=j+1}^{e}\rho_{kj}^l) + p_j^l + \phi_j^l, & \\ \quad \max_{s=1}^{i}\sum_{k=s}^{i-1}\rho_{kj}^l + \min_{s=i+1}^{j}\sum_{k=i+1}^{s-1}\rho_{kj}^l] & \text{if } i < j \\ \min[(\max_{s=1}^{j}\sum_{k=s}^{j-1}\rho_{kj}^l + \sum_{k=j}^{i-1}\rho_{kj}^l + \max_{e=i}^{N}\sum_{k=i+1}^{e}\rho_{kj}^l) + p_j^l + \phi_j^l, & \\ \quad \min_{e=j}^{i-1}\sum_{k=e+1}^{i-1}\rho_{kj}^l + \max_{e=i}^{N}\sum_{k=i+1}^{e}\rho_{kj}^l] & \end{cases}$$

$$\forall j,l : \tau_j^{l+1} = p^l(j) + \rho_{jj}^l + \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}^l) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}^l)$$

until convergence

compute optimal configuration: $\forall i,j$ i is in the segment centred around j iff $\rho_{ij} + \alpha_{ij} > 0$

output: segment centres and segment boundaries

Baselines. Regrettably, we are not aware of another publicly available hierarchical segmenter. That is why we used as baselines two recent flat segmenters: *MCSeg* (Malioutov and Barzilay, 2006) and *BSeg* (Eisenstein and Barzilay, 2008). Both were first run to produce top-level segmentations. Each segment thus computed was a new input document for segmentation. We repeated the procedure twice to obtain three-tiered trees. *MCSeg* cannot be run without knowing the number of segments in advance. Therefore, on each iteration, we had to specify the correct number of segments in the reference segmentation. *BSeg* does not need the exact number of segments, so we had two settings: with and without knowing the number of segments.

Evaluation metrics. We did our best to obtain a realistic picture of the results, but each metric has its shortcomings. We compared topical trees using *windowDiff* and *evalHDS* (Carroll, 2010). Both metrics are penalties: the higher the values, the worse the hypothetical segmentation. *evalHDS* computes *windowDiff* for each level of the tree in isolation and weighs the errors according to their prominence in

the tree. We computed *evalHDS* using the publicly available Python implementation (Carroll, 2010).[4]

When computing *windowDiff*, we treated each level of the tree as a separate segmentation and compared each hypothetical level against a corresponding level in the reference segmentation.

To ensure that evaluations are well-defined at all levels, we propagated the more pronounced reference breaks to lower levels (in both annotations and in the results). In effect, the whole sequence is segmented at each level – otherwise *windowDiff* would not be not well-defined. Conceptually this means that if there is a topical shift of noticeable magnitude (*e.g.,* at the top level), there must be at least a shift of less pronounced magnitude (*e.g.,* at an intermediate level).

The *Moonstone* dataset has on average 4.8 annotations per chapter. It is not obvious how to combine these multiple annotations. We evaluated separately each hypothetical segmentation against each available gold standard. We report the averages across all annotators – for both *evalHDS* and *windowDiff* – per level.

Preprocessing. The representations used by *HAPS* and the *MCSeg* are very similar. Both systems compute a matrix of similarities between atomic units of the document (sentences or paragraphs). Each unit was represented as a bag of words. The vectors were further weighted by the *tf.idf* value of the term and also smoothed in the same manner as in (Malioutov and Barzilay, 2006). We computed cosine similarity between vectors corresponding to each sentence or paragraph. We used tenfold cross-validation on the Wikipedia dataset and fourfold cross-validation on the smaller *Moonstone* data.

The quality of the segment centres. In addition to finding topical shifts, *HAPS* identifies segment centres – sentences or paragraphs which best capture what each segment is about. In order to get a rough estimate of the quality of the centres, we extracted paragraphs identified as segment centres at the second (middle) level of *HAPS* trees. These pseudo-summaries were then compared to summaries created by an automatic summarizer *CohSum*. We used ROUGE-1 and ROUGE-L metrics (Lin, 2004) as a basis for comparison. *CohSum* identifies the most salient sentences in a document by running a variant of the TextRank algorithm (Mihalcea and Tarau, 2004) on the entire document. In addition to using lexical similarity, the summarizer takes into account coreference links between sentences. We ran *CohSum* at 10% compression rate.

The summarization experiment was performed on the *Moonstone* corpus. We also collected 20 chapters from several other XIX century novels and used it in a separate experiment. The ROUGE package requires manually written summaries to compare with the automatically created ones. We obtained the summaries from the SparkNotes website.[5]

6 Results and discussion

Table 2 shows the results of comparing *HAPS* with two baseline segmenters using *windowDiff* and *evalHDS*. *HAPS* was run without knowing the number of segments. *MCSeg* required that the exact number be specified. *BSeg* was tested with and without that parameter. Therefore, rows 3 and 4 in Table 2 correspond to baselines considerably more informed than *HAPS*. This is especially true of the bottom levels where sometimes knowing the exact number of segments unambiguously determines the only possible segmentation.

The results suggest that *HAPS* performs well on the *Moonstone* data even when compared to more informed baselines. This applies to both metrics, *windowDiff* and *evalHDS*. *BSeg* performs slightly better at the bottom levels of the tree when it has the information about the exact number of segments. We hypothesize that the advantage may be due to this additional information, especially when segmenting already small segments at level 1 into a predefined number of segments. Another explanation may be that when using *windowDiff* as the evaluation metric, *HAPS* was fine-tuned so as to maximize the value of *windowDiff* at the top level, effectively disregarding lower levels of segmentation.

[4]When working with the *Moonstone* dataset, we realized that the software produces very low values, almost too good to be true. That is because the bottommost annotations are very fine-grained. Sometimes each paragraph corresponds to a separate segment. This causes problems for the software. So, when we report *evalHDS* values for the *Moonstone* dataset, we only consider two top levels of the tree, disregarding the leaves. We also remove the "too good to be true" outliers, though the "bad" tail is left intact. We applied the same procedure to all three segmenters, only for the *Moonstone* dataset.

[5]`http://www.sparknotes.com/`

	Level	*Moonstone windowDiff*	*Wikipedia windowDiff*	*Moonstone evalHDS*	*Wikipedia evalHDS*
HAPS	3 (top) 2 (middle) 1 (bottom)	0.337 (± 0.060) 0.422 (± 0.060) 0.556 (± 0.070)	0.421 (± 0.060) 0.447 (± 0.070) 0.617 (± 0.080)	0.353 (± 0.072)	0.450 (± 0.015)
MinCutSeg-iter. segm. known	3 (top) 2 (middle) 1 (bottom)	0.375 0.541 0.601	0.440 (± 0.075) 0.424 (± 0.064) 0.471 (± 0.057)	0.377 (± 0.002)	0.444 (± 0.002)
BayesSeg-iter. segm. known	3 (top) 2 (middle) 1 (bottom)	0.353 (± 0.071) 0.406 (± 0.053) 0.504 (± 0.064)	0.391 (± 0.070) 0.344 (± 0.033) 0.354 (± 0.033)	0.367 (± 0.089)	0.370 (± 0.019)
BayesSeg-iter. segm. unknown	3 (top) 2 (middle) 1 (bottom)	0.600 (± 0.071) 0.447 (± 0.053) 0.545 (± 0.064)	0.637 (± 0.070) 0.877 (± 0.033) 0.952 (± 0.033)	0.453 (± 0.089)	0.437 (± 0.022)

Table 2: Evaluation of *HAPS* and iterative versions of *APS*, *MCSeg* and *BSeg* using *windowDiff* per level (mean *windowDiff* and standard deviation for cross-validation)

	Moonstone corpus		Austen corpus	
	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L
Segment centres	0.341 (0.312, 0.370)	0.321 (0.298, 0.346)	0.291 (0.272, 0.311)	0.301 (0.293, 0.330)
CohSum summaries	0.294 (0.243, 0.334)	0.269 (0.226, 0.306)	0.305 (0.290, 0.320)	0.307 (0.287, 0.327)

Table 3: *HAPS* segment centres compared to *CohSum* summaries: ROUGE scores and 95% confidence intervals

All segmenters perform worse on the Wikipedia dataset. Using that scale, informed *BSeg* performs the best, but it is interesting to note a significant drop in performance when the number of segments is not specified.

Overall, *HAPS* appears to perform better than, or comparably to, the more informed baselines, and much better than the baseline not given information about the number of segments.

We also made a preliminary attempt to evaluate the quality of segment centres by comparing them to the summaries created by the *CohSum* summarizer. In addition to working with the *Moonstone* corpus, we collected a corpus of 20 chapters from various novels by Jane Austen.

Table 3 shows the results. They are not conclusive because there is no evidence that ROUGE scores correlate with the quality of automatically created summaries for literature. According to the scores in Table 3, however, the summaries created by *CohSum* cannot be distinguished from simple summaries composed of segment centres identified by *HAPS*. We interpret this as a sign that the centres identified by *HAPS* are approximately as informative as those created by an automatic summarizer.

7 A brief conclusion

This paper presented *HAPS*, a hierarchical segmenter for free text. Given an input document, *HAPS* creates a topical tree and identifies a segment centre for each segment. One of the advantages of *HAPS* is that it does not require the exact number of segments in advance. Instead, it estimates the number of segments given information on generic preferences with regard to segmentation granularity. We also created a corpus of hierarchical segmentations which has been annotated by 3-6 people per chapter.

A Java implementation of *HAPS* and the *Moonstone* corpus are publicly available.[6]

Acknowledgements

We thank Chris Fournier (for computing *S* values using a beta version of SegEval software for hierarchical datasets), Lucien Carrol (for help and discussion of the *evalHDS* software and representation) and Christian Smith (for allowing us to use his implementation of *CohSum*).

[6] `http://www.eecs.uottawa.ca/~ankazant/`

References

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

David Blei and Pedro Moreno. 2001. Topic segmentation with an aspect hidden Markov Model. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 343–348.

Lucien Carroll. 2010. Evaluating Hierarchical Discourse Segmentation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 993–1001.

Lan Du, Wray Buntine, and Mark Johnson. 2013. Topic Segmentation with a Structured Topic Model. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200, Atlanta, Georgia.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian Unsupervised Topic Segmentation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 334–343, Honolulu, Hawaii.

Jacob Eisenstein. 2009. Hierarchical Text Segmentation from Multi-Scale Lexical Cohesion. In *Proceedings of the 2009 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 353–361. The Association for Computational Linguistics.

Vanessa Wei Feng and Graeme Hirst. 2012. Text-level Discourse Parsing with Rich Linguistic Features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 60–68, Jeju Island, Korea, July. Association for Computational Linguistics.

Chris Fournier and Diana Inkpen. 2012. Segmentation Similarity and Agreement. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 152–161, Montréal, Canada.

Inmar E. Givoni, Clement Chung, and Brendan J. Frey. 2011. Hierarchical Affinity Propagation. In *Uncertainty in AI, Proceedings of the Twenty-Seventh Conference (2011)*, pages 238–246.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring Content Models for Multi-Document Summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, Boulder, Colorado, June.

Marti A. Hearst. 1997. TextTiling: segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

Hugo Hernault, Helmut Prendinger, David A. duVerlea, and Mitsuru Ishizuka. 2010. HILDA: A Discourse Parser Using Support Vector Machine Classification. *Dialogue and Discourse*, 3:1–33.

Anna Kazantseva and Stan Szpakowicz. 2011. Linear Text Segmentation Using Affinity Propagation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 284–293, Edinburgh, Scotland.

Anna Kazantseva and Stan Szpakowicz. 2012. Topical Segmentation: a Study of Human Performance and a New Measure of Quality. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 211–220, Montréal, Canada.

Anna Kazantseva. 2014. *Topical Structure in Long Informal Documents*. Ph.D. thesis, University of Ottawa. ⟨`http://www.eecs.uottawa.ca/~ankazant/`⟩.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of summaries. In *Text Summarization Branches Out, Proceedings of the ACL Workshop*, pages 74–81, Barcelona, Spain.

Igor Malioutov and Regina Barzilay. 2006. Minimum Cut Model for Spoken Lecture Segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, Sydney, Australia.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Meghana Marathe. 2010. Lexical Chains Using Distributional Measures of Concept Distance. Master's thesis, University of Toronto.

Daniel Marcu. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, Mass.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In Dekang Lin and Dekai Wu, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing 2004*, pages 404–411, Barcelona, Spain.

Hemant Misra, François Yvon, Olivier Cappé, and Joemon M. Jose. 2011. Text segmentation: A topic modeling perspective. *Information Processing and Management*, 47(4):528–544.

Hyo-Jung Oh, Sung Hyon Myaeng, and Myung-Gil Jang. 2007. Semantic passage segmentation based on sentence topics for question answering. *Information Sciences, an International Journal*, 177:3696–3717.

Lev Pevzner and Marti A. Hearst. 2002. A Critique and Improvement of an Evaluation Metric for Text Segmentation. *Computational Linguistics*, 28(1):19–36.

Jay M. Ponte and W. Bruce Croft. 1998. A Language Modeling Approach to Information Retrieval. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, Melbourne, Australia.

Christian Smith, Henrik Danielsson, and Arne Jnsson. 2012. A more cohesive summarizer. In *24th International Conference on Computational Linguistics, Proceedings of COLING 2012: Posters*, pages 1161–1170, Mumbai, India.

Fei Song, William M. Darling, Adnan Duric, and Fred W. Kroon. 2011. An iterative approach to text segmentation. In *Proceedings of the 33rd European Conference on Advances in Information Retrieval*, ECIR'11, pages 629–640, Berlin, Heidelberg. Springer-Verlag.

Florian Wolf and Edward Gibson. 2006. *Coherence in Natural Language: Data Structures and Applications*. MIT Press, Cambridge, MA.

Yaakov Yaari. 1997. Segmentation of Expository Texts by Hierarchical Agglomerative Clustering. In *Proceedings of International Conference on Recent Advances in Natural Language Processing RANLP97*, pages 59–65, Tzigov Chark, Bulgaria.

Gilbert Youmans. 1991. A new tool for discourse analysis: The vocabulary-management profile. *Language*, 67(4):763–789.

Capturing Cultural Differences in Expressions of Intentions

Marc T. Tomlinson **David B. Bracewell** **Wayne Krug**
Language Computer
Richardson TX 75080
marc, david, wayne@languagecomputer.com

Abstract

The intersection of psychology and computational linguistics is capable of providing novel automated insight into the language of everyday cognition through analysis of micro-blogs. While Twitter is often seen as banal or focused only on the who, $what$, $when$ or $where$ tweets can actually serve as a source for learning about the language people use to express complex cogntive states and their cultural identity. In this contribution we introduce a novel model which captures latent cultural dimensions through an individual's expressions of intentionality. We then show how these latent cultures can be used to create a culturally-sensitive model which provides enahnced detection of signals of intentionality in tweets. Finally, we demonstrate how these models reveal interesting cross-cultural differences in the goals and motivations of individuals from different cultures.

1 Introduction

Social media platforms have enabled new forms of discourse and have also provided enormous quantities of data on these communications. For instance, the popular microblogging service Twitter provides an exceptionally useful source of user-generated content which has attracted considerable interest from researchers in computational linguistics (Ritter et al., 2009; Gimpel et al., 2011). Most of the language processing on tweets has involved the identification of sentiment (Davidov et al., 2010), summarization (Sharifi et al., 2010), conversational models of Dialogue acts (Ritter et al., 2009), or lexical and semantic processing. In this effort we expand on these previous approaches and show how individuals express their cultural identity through expressions revealing their intentionality towards events and provide a way of capturing this information.

We define intentionality as the amount of effort an individual is willing to expend to achieve a goal(Ajzen, 1991). Goals represent future states or events which an individual wishes to happen. Accordingly, intentions are goals for which an individuals is willing to expend at least some minimal amount of effort to bring about. While people express goals throughout the day, intentions are the goals that they are willing to follow through with. Identifying when a goal is actually an intention requires the successful recognition of many distinct cognitive factors that can be revealed through the individual's use of language.

There is a long history of studies that have worked towards identifying a set of factors that underly an individual's intentions (Ajzen and Fishbein, 1977; Ajzen, 1991; Malle and Knobe, 1997; Sloman et al., 2012) of which, the setting of goals is one important factor. These studies have concentrated on identifying the factors that affect an individual's motivation. The studies have also identified a set of factors that people use to gauge the intentionality of other individuals. However, these factors have always been manually identified by an expert from an individual's speech or writing. It is not clear that these features can actually be detected automatically in language.

Intentions have also been considered in computational linguistics. In their seminal work entitled, "Attention, Intentions, and the Structure of Discourse" (Grosz and Sidner, 1986), Grosz and Sidner point out the fundamental role of intentions and their effect on the theory and processing of discourse structure. They even define a set of intentions that can be held by individuals that are relevant to discourse theory. In contrast, we focus on understanding intentions outside of the discourse. In addition, we work with a more general definition of intentions taken from psychology, defining intentionality as the amount of effort an individual is willing to expend to achieve a goal.

Culture refers to the set of beliefs, norms, and customs shared by a group of people. Beliefs and culture are inseparably tied to intentions and language (Ajzen and Fishbein, 1977; Tomasello et al., 2005). Culture affects an author's proclivity to have a particular intention, for example Hofstede's dimension of power distance (Hofstede, 1980) would suggest that individuals from high power-distance cultures have a lower likelihood of performing

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 48–57, Dublin, Ireland, August 23-29 2014.

actions with the intention of overriding the actions of an individual of higher status. Culture can also affect the way in which individuals reason about other agents' intentions and the set of actions that are used to realize an individual's intentions. While considerable work has looked at the link between cultures and intentions, here we show how a latent representation of an individual's culture derived from their intentions can be utilized to explore the intersection between culture and intentions using the vast amount of written expressions present on Twitter.

In this contribution, instead of focusing on the discourse meaning of intentions, we look at how personal intentions can be understood through Twitter posts by focusing on the language of those posts contain. We briefly discuss previous work showing how it is possible to capture language that reveal cognitive factors of intentionality which could be used to capture broader intentions. Critically, we then augment the models of the cognitive factors of intentionality by accounting for the culture of the authors on Twitter. Twitter contains an immense number of authors covering a variety of different cultures definable at different levels, for example women, college-students, or fitness buffs.

We have evaluated the models on a very large set of over 7.5 million tweets which cover a sampling of Twitter from early 2011 to the middle of 2013. Our sample includes just over 900,000 authors. We found very promising results for identifying the factors of intentionality, but by considering culture we were able to provide a significant improvement of those results. We have shown that cognitive factors of intentionality, including goals, control and skill, and rewards can be recognized through the use of simple language models. Similarly, our cultural models were based on traditional techniques for latent variable modeling through principal component analysis enabling an understanding of the cultural distribution of intentions.

The remainder of the paper is organized as follows. We first present the cognitive factors of intentionality that we have used for this contribution. We then present a new cultural model of authors on Twitter and compare it to existing approaches in the literature. We then present a series of models which capture the cultural variation of the cognitive factors of intentionality. Finally, we present a look at some of the cultural differences identified through our approach.

2 Factors of Intentionality

While there are numerous factors that affect an individual's intentionality (Ajzen, 1991; Malle and Knobe, 1997; Sloman et al., 2012), in this contribution we focus on investigating the most historically central factors: goals, perceptions of control, and rewards. Below we provide brief examples of the three factors before detailing our approach for identification of latent cultures.

2.1 Factor 1: Goals

The first factor that we consider is evidence that an individual has a goal. Goals are expressions of a desire for a change of state or rewards which could require an action on the part of the individual. The setting of goals for both action and inaction have been linked to many different motivational and long-term outcomes (Albarracin et al., 2011; Locke, 1968). Examples of goals are

(1) I want to finish my paper

(2) I want to be famous

The first example of a goal expresses an intention to perform an action which could result in a positive reward for the individual, however it doees not mention the reward. In contrast, the second example expresses a clear expectation for a reward (fame), but does not describe the actions that will lead to that reward. Does the individual want to be President or the next Kardashian? Additionally, in contrast to explicit goals stated by an individual, goals can also be inferred by other people based on an analysis of actions (perceived intended events) carried out by the individual. For example, it is presumed that an individual has a goal to win the lottery when they buy a lottery ticket, or that the occupants of a car full of beach toys is headed to or from a beach. Goals represent the factor that has seen the most recent attention in terms of the creation of automatic methods for their recognition (Chen et al., 2013; Banerjee et al., 2012).

2.2 Factor 2: Perception of Control

Intentions are revealed not just through goals, but also through words expressing skill or a level of control. Individuals that feel that they have more control over a situation will expend more effort on their actions (Ajzen, 1991). Individuals are also perceived by others as having greater intentionality for actions that they have control over or exhibit skill at. We considered multiple ways in which an individual can express their perceived control over an event, subdividing this factor into three sub-factors. The first sub-factor captures expressions which indicate skill.

(3) Just helped some guy push his gas-less car to the garage #iamwoman #hearmeroar

Table 1: Example Hash Tags and Tweets.

Cognitive Factor	Sample Tags	Sample Tweets
F1: Goal	#goalinlife, #mywish	"3 more days of studying"
F2: Control	#dowhatisay, #kissmyfeet	"I defy the law of gravity"
F2: Skill	#madskillz, #iamapro	"you are flat out amazing to watch"
F2: Lack of Control	#oops, #cantstop	"cannot believe I said that"
F3: Negative Reward Self	#fml, #crap	"I just locked the keys in my car"
F3: Negative Reward Other	#worstdriverever, #awkward	"It does make me cringe"
F3: Positive Reward Self	#whyismile, #victoryismine	"my cats make me smile"
F3: Positive Reward Other	#ff, #thatsbadass	"Solar panels on the white house"

The second sub-factor captures expressions of control.

(4) I'm in control here!

The third sub-factor captures expressions of lack-of-control.

(5) i'm a little nervous for tomorrow

While several linguistic theories exist that could be utilized to create systems detecting control, such as agency (Dowty, 1991), there is no prominent work on automatically identifying control directly in an individual's expressions.

2.3 Factor 3: Reception of Rewards

Intentions can also be inferred when an individual receives a reward.

(6) I'm so proud of what I did

(7) Your work sucks!

Rewards can be positive (increasing the likelihood of the action being repeated, Example 6) or negative (decreasing the likelihood of the action in the future, Example 7). In addition, rewards can come from the individual (self-directed rewards, Example 6) or from other individuals (other-directed rewards, Example 7). This establishes four sub-factors for rewards. Knowing that an individual received a reward increases the likelihood that they had effortful participation in the event. In addition, evidence of negative rewards are strongly inferential for intentionality (Knobe, 2003). Interpretations of rewards are very culturally sensitive. For example, a comment such as "That is disgusting" would have a good chance of being interpreted as a positive reward when it was made as a comment to a user-generated contribution on the website DeviantArt.com. Additionally, the effect of rewards on motivation is not always clear-cut. Experts seek out and are actually motivated by criticism (Finkelstein and Fishbach, 2012).

2.4 Linking Hashtags and Factors of Intentionality

The factors and sub-factors described above capture expressions which can be used to infer an individual's intentionality towards a future action. In Tomlinson et al. (2014) we showed that it is possible to link particular hashtags used by people on Twitter to these cognitive factors. Our approach utilized two annotators. The first annotator, through trial and error, identified a large number of potential candidate tags for each sub-factor. The annotator then rated each hashtag for how well tweets containing that hashtag exhibited each sub-factor (on a scale of 1-5). The second annotator then separately rated each tag which scored a 4 or 5. The two annotators had an agreement rate of 87%. 178 tags in all were agreed to be a 4 or 5 by both annotators and considered representative of the particular sub-factor. Examples of the hashtags utilized and tweets with those tags are shown in Table 1. The tweets have been modified slightly to preserve anonymity.

3 Identification of Latent Cultures

In the preceding section we discussed examples of goals, control, and rewards, and discussed how hashtags are used on Twitter to mark a tweet expressing one of these factors. Some of these examples require cultural knowledge in order to correctly interpret. In this section we present the latent model of culture that is used for learning the cultural specific expressions of the factors of intentionality.

3.1 SVD-Model of Culture

A considerable amount of work has demonstrated how particular social characteristics of individuals can be identified on Twitter, such as gender, age, and political orientation (Zamal et al., 2012; Pennacchiotti and Popescu, 2011). While superb results can be obtained for identifying these characteristics of authors using a complex set of features, this approach does not neccesarily allow for generalization to other data sets. Therefore we settled on an approach utilizing a specially trained latent variable model. Instead of utilizing Latent-Dirichlet Allocation (LDA, Blei, Ng, and Jordan, 2003) as Pennacchiotti and Popescu we utilized a spectral analysis based on singular-value decomposition (SVD). This approach has been shown to be generally superior to LDA on the domain of topic modeling (Chen et al., 2011), but has not been tested for cultural modeling.

3.1.1 Data

We randomly sampled 1.6 million tweets from a Twitter dataset that had been generated by retrieving tweets that carried at least one of the hashtags linked to a cognitive factor of intentionality (and other posts by that author). In addition, we restricted the set to authors for which we had at least 20 posts in our dataset. For this dataset, all of the markup was left in the tweet (e.g. hashtags, urls, etc.).

3.1.2 Model

From our dataset we created a set of documents, $D = \{a_1, a_2, \ldots, a_A\}$. Where each a_i represents the entire collection of tweets for a single author that contain mentions of goals, skill/control, or rewards. This set of documents contains N words and hashtags. We then create a matrix, $X \in \Re^{N,A}$, where each author represents a row in the matrix and the columns are the number of times that the corresponding word or hashtag was used by that author. Then we perform a singular value decomposition of the matrix to solve

$$X = VSC^T \tag{1}$$

Where S is a $k\ x\ k$ matrix whose off-diagonal entries equal 0 and the on-diagonal entries are the k singular values for the matrix X. For our approach we set k equal to 100. V represents a mapping of the words into our reduced space $\Re^{n,k}$, and $C\ \Re^{i,k}$ contains a weighting for each author with respect to the k^{th} latent cultural dimension. The cultural model can be used to identify the culture of an unseen author through the creation of a projection matrix, P.

$$P = VS^{-1} \tag{2}$$

This matrix projects the tweets that make up the author into our latent cultural space C. This allows us to map each author in our complete data set into our latent space which can then be used for training and testing. The latent cultural space can be used to characterize the culture of an author as a distribution over the dimensions. Below we evaluate our latent cultures on the shared dataset provided by Zamal et al. 2012.

3.2 Evaluating the Latent Cultures

Culture is a system of shared beliefs and actions. Culture is often shared between individuals based on social similarity, this can be within a language, nation, gender, age-group or other social distinction. Thus, being able to identify an individual's culture should facilitate detection of socio-demographic information. To test this we looked at using the latent cultural dimensions to predict socio-demographics on Twitter. We looked at the systems ability to identify gender (male vs. female), age (young vs. old), and political orientation (Democrat vs. Republican) of individuals based on their exhibition of particular latent cultural dimensions. In this model we first represented an individual's tweets as a distribution over the latent dimensions. We then utilized two different statistical approaches to find associations between particular dimensions and the relevant socio-demographic information. For a comparison, we tested our SVD-culture model against a similarly trained LDA model and a model based on n-grams.

3.2.1 Data

We utilized the publicly available dataset from Zamal, Liu, and Ruths (2012). The dataset consisted of Twitter user names and associated meta-data identifying their gender (Male or Female), age (two classes, young and old), and political orientation (Liberal or Conservative). Unfortunately, many of the identified tweets were no longer available from the Twitter API, but we successfully retrieved 2.6 million tweets from authors identified in the dataset with 310 users identified for gender, 320 identified for their age, and 380 for their political affiliation. The tweets in our dataset are substantially different from the original dataset because of the time over which they were collected. Zamal, et al.'s tweets were from 2012 and before, whereas our tweets covered much of 2013. This suggests that comparisons of the raw numbers should be made with caution, particularly in the political area.

Table 2: Results for identifying user demographics based on latent cultural dimensions compared to linguistic style and an ensemble method utilized by Zamal et al (2012).

		Zamal et al.	N-Grams	LDA	SVD
	N	F	F	F	F
Gender	310	.80	.57	.71	.70
Age	320	.75	.63	.66	.67
Political Orientation	380	.89	.73	.66	.68

3.2.2 Modeling & Results

To provide a comprehensive view of the strengths and weakness of our approach we compared several models for their ability to correctly predict the cultural demographics of individuals on Twitter. We first established a base-line model which was an n-gram language model created from the language used by each individual in their tweets. This model learned to identify the cultural demographics based on the frequency with which individual's in that demographic used sequences of words, called n-grams. This approach is consistently ranked as one of the single best approaches to authorship identification and performs well on a large variety of datasets.

We also tested the SVD-Culture model introduced above on this dataset. For this experiment, we trained a logistic-regression based classifier to identify the demographic information of an author based on the vector created by projecting that author into our latent space.

Finally, to look for a difference in the performance between an SVD-based latent representation and one based on LDA (Pennacchiotti and Popescu, 2011), we also trained and tested an LDA-based Culture model. The model was trained on the same data as the SVD-based model and utilized the same number of dimensions.

All of the models were tested and trained utilizing 10-fold cross validation. It is very important to point out that the data sets used to generate the underlying latent representational models did not include any of the tweets from the data used for the 10-fold cross validation. That data was only utilized for the supervision of the logistic regression.

The results of the base model, the SVD model, the LDA model, and the original results presented by Zamal et al. (2012) are shown in Table 2. The latent models are clearly superior to the language model, on average outperforming it by a significant margin of 4%. As expected, the SVD-based model does outperform the LDA model on average, though it is only by 1%, on average.

The strength of this approach is in its simplicity. The latent cultural dimensions have been learned on a wholly different dataset than that used for testing, this supports good generalization performance. While the latent SVD-cultural model does not reach the performance of the system created by Zamal et al. (2012). Zamal et al.'s results were obtained using a plethora of different feature types, which were specifically trained to solve each individual problem. As pointed out in Cohen and Ruths (2013) this causes some issues on transfer to a novel dataset, because the selected features were not representative of differences between liberals and conservatives in the second dataset. In contrast, we suggest that the latent cultural model learns a more general representation utilizing only the set of features provided by the underlying latent cultural models, which were not trained on any of the data in the test set. Additionally, the latent SVD model is easy to implement and train.

Importantly, these results indicate that the latent cultural dimensions capture similarities in the ways in which individuals of similar socio-demographics express themselves on Twitter. The model is able to easily identify the gender, age, and political affiliation of individuals based on their tweets. In the next section we show how we can utilize these latent cultural dimensions to facilitate learning of expressions conveying factors of intentionality.

4 Cultural Sensitive Identification of Cognitive Factors of Intentionality in Language

Recognizing language that expresses factors of intentionality is complicated because of the wide variety of ways in which they can be expressed as shown in the examples in the previous section. While some work has explored automatic goal recognition, most recently by (Chen et al., 2013) and (Banerjee et al., 2012), little work has been done automatically characterizing the other factors, though work in detecting social implicatures in language is similar (Bracewell et al., 2012b). We first present a general framework for learning to model the content of tweets that express a given factor from our cognitive model, we then show how this approach can be enhanced with the addition of latent cultural dimensions.

4.1 Culture Agnostic Model

Here we introduce the $General\ Model$ that serves as the basis for the culture specific models. It is so named because it applies to all cultures. We utilized an n-gram based language model to identify the factors in tweets.

We first constructed a vocabulary of all n-grams between 2 and 4 words in length. Each tweet, j, which is labeled with a hashtag linked to a sub-factor f, is represented as a vector, X^j. Entries in X^j correspond to the number of occurrences in the tweet of the i^{th} n-gram from the vocabulary. We examined two different mathematical approaches to modeling the cognitive factors to gain a better understanding of the problem.

The first approach utilized a Naive-Bayes based classifier (NB) where

$$p(F = f|X) = \frac{p(F = f) \cdot \ p(X|F = f)}{p(X)} \tag{3}$$

The second approach utilized an L2-loss logistic regression model (L2):

$$p(F = f|X, W) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)} \tag{4}$$

In which the weights, W, are learned by maximizing Equation (4)

$$\sum_{j}^{m_f} log\ p(y^j|X^j; W) - \alpha||W||_2^2 \tag{5}$$

where m_f represents a balanced training set created by randomly sampling the training tweets that are tied to sub-factor f and an equal number of tweets that express one of the other factors. For solving the maximization problem we utilized the LibLinear package (Fan et al., 2008).

4.2 Culture Sensitive Models

We compared two different methods for integrating the culture information from the SVD-based culture model into the models for identifying the cognitive factors of intentionality. Both models assume that the authors have been partitioned into a set of cultures, L, but differ in their modeling of the link between language and cognitive factors.

In order to identify the cultures of the authors we utilize a clustering of the latent dimensions produced by the SVD model, a spectral clustering (Kannan et al., 2004). We utilized a simple hierarchical clustering that capitalizes on the y largest singular values. We create a set of hierarchical clusters based on a median split of each of the first y columns in our latent space. When $y = 1$ we have two clusters where the authors have been split based on the median value of the first latent dimension, with $y = 2$ each cluster is then independently split by the author's value along the second latent dimension, giving four clusters, and so on.

4.2.1 Culture-Specific Model

Our first method, which we call the culture specific model uses a separate model of each factor for each latent culture, $l \in L$. We first identify a tweet x, as belonging to a given culture, l. We then determine whether or not the language it contains expresses a particular cognitive factor based on

$$p(F = f|x_l, L_l) \tag{6}$$

To learn the function we utilize a linear classifier, Logistic-Regression with an L2 regularization term, and limit the training data to authors that belong to the particular culture.

4.2.2 Joint-Culture Model

Our second model, which we call the joint culture model utilizes an ensemble based approach. For each tweet, x_l, we calculate both a culture specific view of the language in the tween $p(F = f|x_l, L_l)$ and a culture agnostic view $p(F = f|x_l)$, taking the classification is that is most confident. This joint approach utilizes the culture-agnostic model to smooth deficiencies caused by insufficient culture-specific data.

4.2.3 Number of Cultures

We explored settings of $y = \{2, 3, 4, 5\}$ latent dimensions which equates to $\{2, 4, 8, 16, 32\}$ latent cultures. Authors are first split according into their cultural group and then tweets from each culture are broken into a training and testing set. Because of the amount of data we utilized only a 5-fold cross validation procedure. In addition, we also tested a random culture model that randomly assigned authors to cultures instead of utilizing the spectral clustering. When creating these random cultures we balanced the number of authors in each random culture with the corresponding spectral cultures.

Table 3: Accuracies for modeling each sub-factor of intentionality. L2 represents results obtained using an L2-regularized linear regression, NB represents naive-Bayes, #Cultures signifies the number of latent cultural dimensions used for clustering.

	General		L2 - Culture Specific				L2 -Joint Culture				
#Cultures	NB-0	L2-0	2	3	4	5	2	3	4	5	NB-5
F1:Goals	79.8	80.9	81.1	80.9	79.8	78.8	82.1	**82.2**	82.1	82.0	79.1
F2:Control	70.1	75.5	75.5	75.3	74.4	73.8	76.4	**76.4**	76.4	76.4	72.3
F2:Lack of Control	69.1	73.7	75.2	74.9	74.1	72.9	**75.9**	75.7	75.8	75.6	71.6
F2:Skill	73.2	76.2	77.6	77.0	76.4	75.5	78.2	**78.2**	78.1	78.1	75.3
F3:Positive Other	78.3	82.9	84.3	84.1	83.7	83.4	**84.5**	84.4	84.4	84.5	81.9
F3:Positive Self	66.0	69.1	70.6	70.3	69.4	68.7	71.3	71.3	71.3	**71.4**	68.4
F3:Negative Other	68.7	72.3	73.6	73.4	72.5	71.6	74.1	74.0	**74.1**	74.0	70.8
F3:Negative Self	69.3	72.4	73.6	73.3	71.9	71.3	**74.4**	74.3	73.9	73.7	71.1

4.3 Data

Testing was done on a large number of tweets (7.5 million) that contained tweets from individuals that used any of the representative hashtags. In our collection hashtags exhibiting the sub-factor of control contained the largest number with approximately 575,000 tweets, while we only collected 110,000 tweets which were marked with a hashtags indicating positive rewards for the actions of other individuals. For training and testing purposes we removed all URLs, hashtags, and @users from the tweets. We then discarded tweets that were less than two words long. This approach is conservative, because we removed the classifier's ability to directly learn co-occurring hashtags, however we wanted to ensure that we would minimize deficient solutions and maximize the ability of the models to transfer from Twitter to other genres of text.

4.4 Results and Discussion

The accuracy of the classifiers for identifying each sub-factor are shown in Table 3. The accuracies reflect the classifiers ability to separate tweets that have a hashtag representing the given sub-factor from those that do not. The results suggest that all of the models are adequately capturing the differences between the cognitive factors. On average, the logistic regression based classifier achieves a 3.5 percent advantage in accuracy over the Naive-Bayes model, showing a clear advantage for the improved feature selection of the L2-loss logistic regression. Both models required a similar amount of time to train and test.

To conserve space Table 3 shows only the results for the 5-dimension Joint Culture Naive-Bayes model. The results for the Naive-Bayes model match the pattern exhibited by the logistic-regression Joint Culture model, except that the Naive-Bayes Joint-Culture model increases steadily as more groups are added with a maximum performance with 5 latent dimensions. With 5 latent dimension the gap between the two ML approaches shrinks to 2.8 percent (73.2 to 76.0).

On average the Joint Culture model shows a 1.8 percent improvement (74.3 to 76.1) over the culture neutral model for the L2-Logistic Regression, while it is a larger 2.2 percent for the Naive Bayes based approach (71.2 to 73.4). A comparison of the error reduction shows that the cultural integration is very promising. While the L2-loss logistic regression provides an 11 percent error reduction over the Naive-Bayes, the joint culture model achieves a comparable 7-9 percent reduction in error over the L2-loss regression and the Naive-Bayes model.

The improvements are strongest for positive self directed reward factor, skill factor, and lack of control factor. Interestingly, the models also exhibited considerable variation in accuracies across the different cultures, for example utilizing 3 dimensions positive rewards for others in one culture is recognized at 92 percent (this group contains 43,556 tweets), while for another culture of approximately the same size it is only recognized at 77 percent. Unfortunately, when moving to 4 dimension our clustering algorithm splits the group at 92 percent into two groups where the factor can only be recognized with an average of 88 percent accuracy. This suggests that more complex clusterings strategies within the latent space would be beneficial.

While not shown in the table for space reasons, we also tested the joint culture model utilizing a random assignment of authors to cultures, instead of relying on the assignment produced by the SVD-model. As expected the random model performed, on average, at approximately the same level as the general model, 74.6% compared to 74.5% respectively. Though the random culture model exhibited considerable variation in relation to the real joint culture model across the different factors. This evidence reinforces the idea that the latent cultures are coherent and that individuals within those cultures express the factors of intentionality in similar ways.

Table 4: Example cultures and the tags that are commonly associated with that factor.

Cultural Label	Cognitive Factor	Common Tags
Alterantive Medicine Health	Positive Rewards Self	#almond, #radish, #curd
Geek Interest	Positive Rewards Self	#theobroma, #freefiction, #nanotech
Teenagers	Positive Rewards Self	#bored, #me, #cute
Urban Hip/Hop	Positive Rewards Self	#bosslife, #teamfastfollow, #indiecharts
Martial Fitness	Goals	#healthynews, #fitnessimages, #fitso
Hip/Hop	Goals	#soundcloud, #support, #dl
General Religion	Goals	#singer, #jesus, #judas

Inspections of tweets where the cognitive factors have been discovered suggest that many times the hashtags are used sarcastically. Anecdotally, we also examined a list of the top hashtags associated with instances labeled by our approach and found good generalization to novel hashtags. We looked at a list of the hashtags based on the average confidence of the labels being applied to the tweets containing those tags, we found many reasonable candidate tags. For example, tweets containing the hashtags #day1 and #day2 were among the most likely to be labeled as exhibiting a goal even though neither were identified by our annotators initially. These two tags are used by individuals on the first and second day of pursuing a goal.

The results presented in this section suggest that breaking down the authors by culture before learning models linking the hashtags marking expressions of the cognitive factors of intentionality to language provides a significant benefit. It also hints at some interesting differences between the groups. In the next section we briefly explore some of those differences.

5 Investigating Cultural Differences in the Language of Intentionality

We investigated the cultural discriminations made by the model by looking at the hashtags that were the most popular for each culture. Two annotators provided labels for each of the cultures based on the most frequent hashtags for that culture. We found that some of the cultures could easily be labeled based on their differential use of topical hashtags. Many of the latent cultures reflected notions of distinctions between cultural (or sub-cultural) groups, such as along political orientation or socio-demographics (urban, hipster, university students, single mothers, and political activist). In addition to the latent cultures that weighed on group identity, some of the other clusters captured more topical information, such as being fitness oriented or discussions focused around sex.

The cultural distinctions allowed us to quantify the differences in the event and intentionality associations across the cultures and differences in expressions indicating cognitive factors of intentionality. For instance, activists and urban individuals were most likely to produce tweets expressing control over situations. There were also groups, such as the camaraderie group where individuals typically set goals that will benefit a group in some way as well as the individual. In most of these cases, the author is the member of a team or some other group that will be engaging in a cooperative or competitive activity. Some authors from this cultural group express goals of providing direct or moral support to specific teams or groups of which they are not members. Others have goals of attending group events or gatherings with no particular membership. In most cases, goals in this culture are associated with positive rewards or defeating an opponent.

Table 4 shows the most probable tags by cognitive factor for some of the more interesting groups. These lists were generated by first eliminating all tags from the culture that were not predictive of the culture. To do this, we generated an estimate of the mean and variance for each hashtag in our dataset across all of the different cultures. We then eliminated all tags where the probability of the tag given the culture was not significantly different than its estimate given the general population. This has the effect of removing that hashtags that signaled the cognitive factors because they had a fairly general distribution across the cultures.

6 Conclusion

In this paper we presented a novel approach for identifying factors of intentionality in tweets. Further, we showed how a latent cultural model could be used to enhance those identifications through an improved understanding of how these factors are expressed across the various cultures. The latent cultural dimensions identified by the model correspond well with real cultural demographic information.

This work presents several exciting possibilities, while Twitter is notoriously difficult for traditional natural language processing work because it doesn't follow established syntactic and semantic conventions, models learned over Twitter data are able to transfer to other types of social media, such as user-generated content sites (Tomlinson et al., 2014a). Hashtags provide a very interesting form of distant annotation that could reduce the amount of time and effort required to create models which capture a nuanced understanding of social or psychological pragmatics,

such as social acts (Bender et al., 2011; Bracewell et al., 2012a), thus making the exploration of a richer language understanding more tractable.

Lastly, we have also shown that the models provide an ability to look at differences between cultures in the how and when of their expressions of factors relating to intentionality. People express lots of goals, but what affects when they actually intent them. These models should be able to provide a novel view on the pulse of a city (Rios and Lin, 2013) or citizens' cognitive responses to events (Dodds et al., 2011). We can use these techniques to identify what events make people establish new goals or instill feelings of a loss of control?

Acknowledgment

This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-12-C-0063. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

References

Icek Ajzen and Martin Fishbein. 1977. Attitude-behavior relations: A theoretical analysis and review of empirical research. *Psychological Bulletin*, 84(5):888–918.

Icek Ajzen. 1991. The Theory of Planned Behavior. *Organizational Behavior and Human Decision Processes*, 50:179–211.

D. Albarracin, J. Hepler, and M. Tannenbaum. 2011. General Action and Inaction Goals: Their Behavioral, Cognitive, and Affective Origins and Influences. *Current Directions in Psychological Science*, 20(2):119–123, April.

Nilanjan Banerjee, Dipanjan Chakraborty, Anupam Joshi, Sumit Mittal, Angshu Rai, and B. Ravindran. 2012. Towards Analyzing Micro-Blogs for Detection and Classification of Real-Time Intentions. *ICWSM*.

E.M. Bender, J.T. Morgan, Meghan Oxley, Mark Zachry, Brian Hutchinson, Alex Marin, Bin Zhang, and Mari Ostendorf. 2011. Annotating social acts: Authority claims and alignment moves in wikipedia talk pages. *ACL HLT 2011*, (June):48.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

David B Bracewell, Marc T Tomlinson, Mary Brunson, Jesse Plymale, Jiajun Bracewell, and Daniel Boerger. 2012a. Annotation of Adversarial and Collegial Social Actions in Discourse. In *6th Linguistic Annotation Workshop*, number July, pages 184–192.

David B Bracewell, Marc T. Tomlinson, and Hui Wang. 2012b. Identification of Social Acts in Dialogue. In *COLING*, number December 2012, pages 375–390.

Xi Chen, Bing Bai, Qihang Lin, and Jaime G Carbonell. 2011. Sparse Latent Semantic Analysis. In *SDM*.

Zhiyuan Chen, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Identifying Intention Posts in Discussion Forums. In *NAACL-HLT*, number June, pages 1041–1050.

Raviv Cohen and Derek Ruths. 2013. Classifying Political Orientation on Twitter : It s Not Easy ! In *ICWSM-2013*, pages 91–99.

Dmitry Davidov, Oren Tsur, and Ari Rappaport. 2010. Enhanced Sentiment Learning Using Twitter Hashtags and Smileys. In *Coling*, number August, pages 241–249.

PS Dodds, KD Harris, and IM Kloumann. 2011. Temporal patterns of happiness and information in a global social network: Hedonometrics and Twitter. *PloS one*, 6.

David Dowty. 1991. Thematic Proto-Roles and Argument Selection. *Linguistic Society of America*, 67(3):547–619.

RE Fan, KW Chang, CJ Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9(2008):1871–1874.

Stacey R. Finkelstein and Ayelet Fishbach. 2012. Tell Me What I Did Wrong: Experts Seek and Respond to Negative Feedback. *Journal of Consumer Research*, 39(1):22–38, June.

Kevin Gimpel, Nathan Schneider, Brendan O Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2011. Part-of-Speech Tagging for Twitter : Annotation , Features , and Experiments. In *Proceedings of the Association for Computational Linguistics*, number 2.

B Grosz and C Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3).

Geert Hofstede. 1980. *Culture's consequences: International differences in work-related values*. Sage Publications, Inc.

R. Kannan, S. Vempala, and A. Vetta. 2004. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3):497–515.

J. Knobe. 2003. Intentional action and side effects in ordinary language. *Analysis*, 63(3):190–194, July.

E. A. Locke. 1968. Toward a theory of task motivation and incentives. *Organizational behavior and human performance*, 3(2).

Bertram Malle and J. Knobe. 1997. The Folk Concept of Intentionality. *Journal of Experimental Psychology*, 33(2):101–121.

Marco Pennacchiotti and Ana-maria Popescu. 2011. to Twitter User Classification. In *ICWSM'11*, pages 281–288.

Miguel Rios and Jimmy Lin. 2013. Visualizing the" Pulse" of World Cities on Twitter. *Seventh International AAAI Conference on Weblogs ...*, pages 717–720.

Alan Ritter, Colin Cherry, and Bill Dolan. 2009. Unsupervised Modeling of Twitter Conversations. In *HTL-NAACL*.

Beaux Sharifi, Mark-anthony Hutton, and Jugal Kalita. 2010. Summarizing Microblogs Automatically. In *ACL-HLT*, number June, pages 685–688.

Steven a. Sloman, Philip M. Fernbach, and Scott Ewing. 2012. A Causal Model of Intentionality Judgment. *Mind & Language*, 27(2):154–180, April.

Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne, and Henrike Moll. 2005. Understanding and sharing intentions: the origins of cultural cognition. *The Behavioral and brain sciences*, 28(5):675–91; discussion 691–735, October.

Marc T Tomlinson, David Bracewell, Wayne Krug, and David Hinote. 2014a. # impressme : The Language of Motivation in User Generated Content. In *CICLING*.

Marc T Tomlinson, David Bracewell, Wayne Krug, David Hinote, and Mary Draper. 2014b. # mygoal : Finding Motivations on Twitter. In *LREC - 2014*. ELRA.

Faiyaz Al Zamal, Wendy Liu, and Derek Ruths. 2012. Homophily and Latent Attribute Inference : Inferring Latent Attributes of Twitter Users from Neighbors. In *ICWSM-2012*.

Identification of Implicit Topics in Twitter Data Not Containing Explicit Search Queries

Suzi Park **Hyopil Shin**
Department of Linguistics, Seoul National University
1 Gwanak-ro, Gwanak-gu, Seoul, 151-745, Republic of Korea
{mam3b,hpshin}@snu.ac.kr

Abstract

This study aims at retrieving tweets with an implicit topic, which cannot be identified by the current query-matching system employed by Twitter. Such tweets are relevant to a given query but do not explicitly contain the term. When these tweets are combined with a relevant tweet containing the overt keyword, the "serialized" tweets can be integrated into the same discourse context. To this end, features like reply relation, authorship, temporal proximity, continuation markers, and discourse markers were used to build models for detecting serialization. According to our experiments, each one of the suggested serializing methods achieves higher means of average precision rates than baselines such as the query matching model and the tf-idf weighting model, which indicates that considering an individual tweet within a discourse context is helpful in judging its relevance to a given topic.

1 Introduction

1.1 Limits of the Twitter Query-Matching Search

Twitter search was not a very crucial thing in the past (Stone, 2009a), at least for users in its early stages who read and wrote tweets only within their curated timelines real-time (Dorsey, 2007; Stone, 2009b; Stone, 2009c). Users' personal interests became one of the motivations to explore a large body of tweets only after commercial, political and academic demands, but it triggered the current extension of the Twitter search service. The domain of Twitter search was widened, for example, from tweets in the recent week to older ones (Burstein, 2013), and from accounts that have a specific term in their name or username to those that are relevant to that particular subject (Stone, 2007; Stone, 2008; Twitter, 2011; Kozak, Novermber 19, 2013). However, the standard Twitter search mechanism is based only on the presence of query terms.

Even though the Twitter Search API provides many operators, the current query matching search does not guarantee retrieving a complete list of all relevant tweets.[1][2] The 140-character limit sometimes forces a tweet not to contain a term, not because of its lack of relevance to the topic represented by the term, but due to one of the following:

Reduction the query term is written in an abbreviated form or in form of Internet slang,

Expansion the query term is in external text that can be expanded through other services such as TwitLonger (http://twitlonger.com) and twtkr (http://twtkr.olleh.com), while the part exceeding 140 characters is shown only as a link on twitter.com, or

Serialization the query term is contained as an antecedent in some previous tweet.

[1] "[T]he Search API is focused on relevance and not completeness." 2 October 2013. Using the Twitter Search API. https://dev.twitter.com/docs/using-search

[2] "[T]he Search API is not meant to be an exhaustive source of Tweets." 7 March 2013. GET search/tweets https://dev.twitter.com/docs/api/1.1/get/search/tweets

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 58–68, Dublin, Ireland, August 23-29 2014.

If these cases are frequent enough, the current query matching search in Twitter will get a low recall rate. Considering that tweets are usually used to obtain as various views on a topic as possible, in addition to accurate and reliable information about it, this setback would block attempts to collect diverse opinions in Twitter.

These three different cases require different approaches. First, *reduction*, one of the most significant characteristics of Twitter data in natural language processing, can be solved by building a dictionary of Internet slang terms or learning them. Second, in case of *expansion* tweets are always accompanied with short URLs (`http://tl.gd` for TwitLonger and `http://dw.am` for twtkr) and the full text is reachable through them. In these two cases, tweets correspond one-on-one with documents, whether reduced internally or expanded externally. This study will focus on the third case, *serialization*, where several tweets may be interpreted as a single document.

1.2 Serialization of Tweets: An Overlooked Aspect of Twitter

Though little reported before, serialization of tweets is frequently observed in Korean data.[3] Influential users like famous journalists, columnists and critics as well as ordinary users often publish multiple tweets over a short period of time instead of using other media such as blogs or web magazines. Types of tweets published in this way by Korean users include reports, reviews, and analysis on political or social affairs, news articles, books, films and dramas. The content users intend to express is longer than a tweet but shorter than a typical blog post. Examples from our dataset will be introduced in Section 3.

This study aims at retrieving tweets on a topic, which cannot be found by the current query-matching system. Such tweets are relevant to a given query but do not contain the necessary words. Under the hypothesis that a considerable number of these tweets not containing the query term are serialized with one containing it overtly, and that serialized segments are integrated into the same discourse context, we built a model that allows us, when given a tweet that includes a query or a mentioned topic, to find the other tweets serialized with it and count them as relevant to the topic. We primarily focused on Korean Twitter data, but we believe that the methods developed here are also applicable to other languages with similar phenomena.

2 Previous Studies

Our study is based on the observation that a tweet in a "serialization" does not necessarily correspond to a full document. In fact, it has already been reported (Hong and Davison, 2010; Weng et al., 2010; Mehrotra et al., 2013) that a single tweet is too short to be treated as an individual document, especially considering that word co-occurrence in a tweet is hardly found. Studies proved that performance of Latent Dirichlet Allocation (LDA) models for Twitter topic discovery can be improved by aggregating tweets into a document. In these studies, a "document " consists either of all tweets under the same authorship (Hong and Davison, 2010; Weng et al., 2010), all tweets published in a particular period, or all tweets sharing a hashtag (Mehrotra et al., 2013). These criteria are useful for finding topics, into which tweets can be classified, but our purpose requires a different degree of "documentness." Our study deals with a fixed topic and is interested in whether or not only tweets relevant to the topic can be pooled. All tweets merged into the same document as constructed in the previous studies are not necessarily coherent or related to the same topic because it is not usually expected that ordinary users devote their Twitter accounts to a single topic. In this study, we will develop more detailed criteria for the aggregation of tweets by combining authorship with time intervals and adopting features such as sentiment consistency and discourse markers.

A method of using discourse markers for microblog data was proposed by Mukherjee and Bhattacharyya (2012). They noted that a dependency parser, on which opinion analyses using discourse information (Somasundaran, 2010) are usually based, is inadequate for small microblog data, and instead used a "lightweight" discourse analysis, considering the existence of a discourse marker on each tweet. The list of discourse markers used in their study was based on the list of conjunctions representing discourse relations presented by Wolf et al. (2004). This method was successful for sentiment

[3] Some Korean users sarcastically call this a "saga" of tweets.

analysis on Twitter data assuming that the relevance of each tweet to a certain topic was already known. We will take a similar approach of using discourse markers, but with a different assumption and for a different purpose. In our study, we treat unknown topic relevance of tweets with missing query terms by aggregating them with a topic-marked tweet using discourse markers.

3 Features

3.1 Properties of Tweet Serialization

Multiple tweets are likely to be consistent with a topic if they form a discourse as in the following situations, with examples of tweets in Korean translated into English. In each tweet, topic words are in boldface.

Conversation This is the most typical case.

> **U1:** Wow the neighborhood theater is packed; will ***Snowpiercer*** hit ten million?
>
> **U2:** @U1 My parents and my boss are all gonna watch, and they watch only one film a year. This is the measure for ten million.

Comment after retweet Users retweet and comment.

> **U3** RT @U4: Today's quote. "It is stupid to concentrate on symbolic meaning in Wang Kar Wai's *Happy Together.* That would be like trying to find political messages and signs in ***Snowpiercer***." — Jung Sung-Il
>
> **U3** Master Jung's sarcasm.....☆

On-the-spot addition Because a published tweet cannot be edited, users can elaborate or correct it only by writing a new tweet or deleting the existing tweet.

> **U5** Is Curtis the epitome of **Director Bong**'s[4] sinserity
>
> **U5** Sincerity, shit

True (intentional) serialization Some users begin to write tweets with a text of more than 140 characters in mind. They arrive at the length limit and continue to write in a new tweet.

> **U6** (1) Watched ***Snowpiercer***. It was more interesting than I thought. It felt more like black comedy than SF. On another note, I was surprised by several oddities, making the film feel more like a Korean film with foreign actors in it rather than Director Bong's Hollywood debut.
>
> **U6** (2) In many ways the film was "nineties"... like watching *The City of Lost Children* all over again... and the trip from the tail-car to the first car, though I expected some kind of level-up for each car,
>
> **U6** (3) the world connected car to car was not an organic world (a sideways pyramid?) but worlds too separate car by car, and the front-car people were so lifeless that I was surprised. The scale of the "charge" after 17 years felt shrunken.

If this is a characteristic feature of Korean Twitter data, this may be due to reasons such as personal writing style, the writing system of the Korean language, and Korean Web platforms. First, it may be simply because these users prefer formal language and are reluctant to use short informal expressions even in Web writing. Second, it is possibly because CJK writing systems including Hangul, the Korean alphabet, have more information per character than the Roman alphabet (Neubig and Duh, 2013). Since a 140-character text in Hangul has generally more information than that in the Roman alphabet, a Korean (or Japanese) user can more readily tweet about content which an English (or other European) language user would consider too long to write about on Twitter. Third, for many Korean users Twitter is the most available medium for publishing their opinions online, as a number of standard blogs have been replaced

[4]Director of the film *Snowpiercer*

by microblogs. Some users divide a long public text into multiple length-limited tweets simply because they do not have a blog to write in.

While Internet slang and abbreviations are common in tweets, "Serializers" tend to use 1) fully-spelled forms (unlike "reducers"), 2) usually without hashtags and emoticons, 3) which are all visible on `twitter.com` itself (unlike "expanders"), so it is not guaranteed that all serialized tweets will contain the topic word, as in the examples above. This implies that some tweet segments in a single discourse may not be retrieved even if the discourse is relevant to a given query. Search results may include a partial document for which it is difficult the full version of which is difficult to find.

3.2 Extralinguistic Criteria

Two tweets are more likely to be a part of a larger document consisting of a series of tweets if

Reply-relation one of them is a reply to the other,

Temporal proximity they are published immediately one after the other, or

Continuation markers they share such markers as numbers, arrows >> and continuation marker '(continued).'

Figure 1 shows examples of each case.

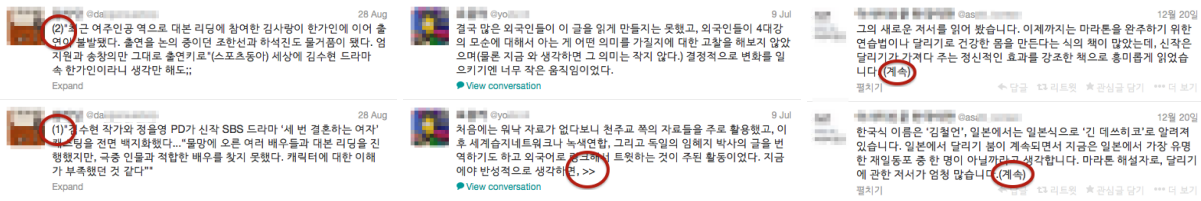


Figure 1: Serialized tweets with numbers, an arrow, or a continuation marker '(continued)'

3.3 Linguistic Clues

Semantic similarity to the query In order to determine the relatedness of two documents, the similarity between their term distributions is mainly considered. Based on this idea, one of our baseline methods will represent each tweet as a bag-of-words vector and retrieve a tweet containing no query term if its tf-idf weighted vector has a high cosine similarity with at least one vector from a tweet containing a query term.

Discourse markers Users may add a discourse marker when writing a new sentence in a new tweet. If a tweet begins with a marker that indicates continuation of a discourse, it is likely to be a part of a larger document. A sentiment analysis in Twitter by Mukherjee and Bhattacharyya (2012) adopted discourse relations from Wolf et al. (2004). In this paper, we use linguistic characteristics described by Kang (1999) in order to classify Korean texts, listing their English translations in Table 1. The *discourse marker* feature refers to whether or not any marker on the list occurs in the first N words (set $N = 5$) of the tweet.

4 Experiments

4.1 Data

We collected 173,271 tweets posted or retweeted by 105 Korean users, including film critics, film students, and amateur cinephiles from 27 July to 26 September 2013. Out of the 105 users, 17 users who had mentioned the film *Snowpiercer*[5] most often were singled out. In addition, the highest overall occurrence of the keyword was found to be between 1 to 15 August, probably due the film's release on 31

[5] `http://www.imdb.com/title/tt1706620/`

Demonstratives	*this, that, it, here, there*
Proverbs	*be so, do so*
Discourse	*well, now*
Conj-Reasoning	*because, so, therfore, thus, hence*
Conj-Conditional	*then, as long as, in the case, under*
Conj-Coordinate	*and, nor*
Conj-Adversative	*but, yet, however, still, by contrast*
Conj-Discourse	*meanwhile, anyway, by the way*

Table 1: List of selected Korean discourse markers used for classifying text types in Kang (1999), translated into English

July in South Korea. Then we kept all 8,543 tweets posted by those 17 users from the period between 1 to 15 August 2013, in order to construct a labeled data set. This set includes 189 tweets that explicitly contain the word *Snowpiercer*. Each tweet in the filtered set was labeled as *related* or *not related* to the movie by three annotators who were Twitter users already following most of the above 17 users and thus aware of the context of most tweets, and a tweet was considered relevant if two or more of the annotators agreed. Inter-annotator agreement was evaluated by using Fleiss's kappa statistic flei:71, which was $\kappa = 0.749$ ($p \approx 0$). Table 2 shows the annotation results.

	Related	Not related	Total
Explicit	173	15	188
Not explicit	207	8,148	8,355
Total	380	8,163	8,543

Table 2: The number of annotated tweets classified by explicitness and relatedness

Table 2 shows that 8163/8543 = 95.55% of the tweets in the dataset are not relevant to the movie *Snowpiercer*. Additional topics are induced from 7–9 manually collected seed words among the 200 most frequently occurring nouns in the dataset, in which each tweet text was POS-tagged by the Korean morphological and POS tagger Hannanum[6]. Induced topics and their seed words are listed in Table 3.

Topic	Seed words
Movie	Movie, Snowpiercer, director, The Terror Live, actor, stage, audience, film, theater
Literature	Story, book, writing, author, novel, character, work
Gender/relationship	Men, women, female, marriage, male, wife, lover
Politics	Politics, state, Park Geun-hye, government, president, party, Ahn Cheol-soo

Table 3: Four topics from manually collected seed words

As described in 3.1, it should be noted again that hashtags are not always useful for finding information in Korean tweets, particularly in this dataset. Among the seed words above, only *Snowpiercer* was ever used as a hashtag, and happened only three times (twice in English and once in Korean). Only nine types of hashtags occurred more than twice in the full dataset (they are presented in Table 4 with their respective frequencies). This predicts that hashtag-based tweet aggregation would not be very useful to find tweets relevant to *Snowpiercer* or one of the four induced topics.

Table 5 shows the number of tweets containing seed words for each topic, where a tweet is allowed to belong to more than one topic. Since only 1853/8543 = 21.69% of the tweets explicitly contain a topic or seed word, it is not plausible that each of the remaining 80% tweets belongs to one of the four topics. Many of the tweets may be related to a topic which was of a too small portion to be induced, or to no topic at all. So, instead of classifying all of the tweets into the given topics, the experiment seeks to retrieve any tweet that is relevant to a certain topic, which allows each tweet to belong to more than one topic at once. In every experiment we regarded tweets that contain a topic or seed word as relevant to the topic, and restricted the test set to those tweets which did not contain them.

[6] `http://sourceforge.net/projects/hannanum/`

#make_people_cry_with_a_story_of_two_words	13
#lgtwins	10
#quote	7
#changing_zero0_to_fatty_makes_things_totally_depressing	6
#EBSbookcafe	4
#today_i_feel	4
#blow_the_whistle_on_chun_doo-hwan	3
#chosundotcom	3
#the_name_of_your_bias_followed_by_the_name_of_the_food_you_just_ate_feels_nice	3

Table 4: Korean hashtags occurring more than twice in the dataset, translated into English

Movie	Literature	Gender	Politics	Total
716	452	379	306	1853

Table 5: Number of tweets including at least one of the seed words for each induced topic

4.2 Measures

For all models, the authors judged the relevance of each of the retrieved tweets for induced topics until ten relevant tweets were retrieved. In the *Snowpiercer* case, precision scores were calculated for all recall scores. We built a ranking retrieval system for each model and evaluated its performance by average precision. For models including a randomizing process, we used the mean of average precisions over 1,000 replicated samples. Precision was computed at every percentile of recall levels for *Snowpiercer* case and after each retrieved relevant tweet (up to top 10) for induced topics. In sum, the performance of a model m was defined in two ways as

$$\text{meanAP@percent}(m) := \frac{1}{1000}\sum_{i=1}^{1000}\text{AP@percent}(m_i)$$

and

$$\text{meanAP@10}(m) := \frac{1}{1000}\sum_{i=1}^{1000}\text{AP@10}(m_i)$$

, where m has 1,000 replicates $m_1, \cdots m_{1000}$ whose measures are

$$\text{AP@percent}(m_i) := \frac{1}{100}\sum_{j=1}^{100}\text{prec@}j\%(m_i)$$

and

$$\text{AP@10}(m_i) := \frac{1}{10}\sum_{k=1}^{10}\text{prec@}k\%(m_i).$$

When m is a tf-idf model, which has a unique ranking without replication, average precision was used.

4.3 Baselines

Query matching method The most obvious baseline method for this study is the current Twitter search system that treats topic words and seed words as queries and finds documents, or tweets, that are relevant to the topic. Since only tweets not containing the query terms remained in the test set, there are no tweets matching them. As the set of retrieved tweets is empty, relevance rank is randomly assigned to each tweet of the test set.

Tf-idf weighting method One may predict that a tweet is likely to be relevant to a topic if it shows a similar word distribution to some explicitly relevant tweets. Under this assumption, we represented each tweet as a tf-idf weighted vector (Salton and Buckley, 1988) after removing all punctuation marks and user-mention markers (`@username`). Stopwords were not removed and tf-idf values were length-normalized. Relevance of each tweet in the test set was defined as the maximum of its cosine-similarities with all tweets containing a query term.

4.4 Tweet Serialization

Examples of Tweet Serialization in Section 3 indicate clues between related tweets other than distributional similarity. When 1) a tweet is a reply to another one, 2) two tweets are written one after another by the same user, 3) one tweet following another includes some discourse marker, or 4) two tweets share a marker, such as numbers, they can be considered to be serialized into a single document rather than being two separate ones. Tweets serialized together are treated as a single document, and if this document contains a a tweet with a query term, then all tweets lacking it but belonging to the same the same document are retrieved. All retrieved tweets are first ranked in random order, followed by the others also in random order.

We suggest four criteria for Tweet Serialization:

Reply Two tweets are serialized if one is a reply to the other.

Continuation markers Two tweets are serialized if they are written successively by the same user and share a marker, such as a number or a phrase "(cont.)"

Discourse markers Two tweets are serialized if they are written successively by the same user, the latter contains one of the discourse markers listed in Table 1 in its first 5 words, and neither of them is a reply to another user.

Time Two tweets are serialized if they are written successively by the same user within a given interval and neither of them is a reply to another user. The upper boundary for intervals is set in one of the following ways:

Constant 30 or 60 seconds

User-specific Users may show different densities in their tweets, depending on their tweeting environment. Distribution of time intervals between successive tweets over users is presented in Table 6. The smallest 5% and 15% quantiles were selected, corresponding to 30 and 60 seconds respectively.

Quantile	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17
0%	3	19	2	1	1	5	10	3	2	3	9	3	2	3	3	2	16
5%	20	42	18	16	13	30	21	18	13	8	43	23	18	15	13	12	110
10%	33	45	25	35	20	43	38	38	28	13	71	35	28	35	21	21	130
15%	47	52	33	57	30	56	67	57	40	23	89	51	37	61	27	40	161
20%	62	67	41	79	41	73	92	74	53	31	111	65	50	84	33	58	197
25%	81	86	55	100	55	92	145	95	69	43	138	84	68	105	38	77	275
50%	237	298	164	322	151	242	1060	297	167	159	297	317	178	258	90	266	725

Table 6: Time intervals (in seconds) by cumulative percentile between consecutive pairs of tweets for each user

For all criteria, Tweet Serialization is transitive, that is, if t_i and t_j are serialized and t_j and t_k are serialized, then t_i and t_k are serialized. Table 7 shows the distribution of serialization sizes (number of serialized tweets) over criteria. Time value of 60 seconds serializes most tweets, as many as (8543-6464)/8543=24.33%, while continuation markers serialize only (8543-8511)/8543 = .37%. Assuming all serializations are correct, the relevance of retrieved documents is judged.

4.5 Results

The average precision values of all models are summarized in Table 8 (means calculated over recall levels) and Figure 2 (means calculated over 1,000 replications). In both Tables 8 and 9, differences between the tf-idf weighting model and each of the Serialization methods were statistically significant according to t-test. Figure 2 compares the results of the serialization methods, among which *continuation marker* model has the highest precision over 0.8 at the 1% recall level, and *Time with 15% quantile* has the average precision score showing the slowest decrease. Even though for all serialization methods average precision values converge to zero as recall levels increase, each of the method gets higher precision rates than baselines until some part of relevant tweets are retrieved.

Size	Repl.	Disc.	Cohe.	T:30s	T:60s	T:5%	T:15%
1	8137	8169	8511	7314	6464	7845	6849
2	88	166	6	465	664	298	610
3	34	14	2	76	149	31	109
4	9	0	0	6	40	1	19
5	3	0	1	5	12	1	8
6	5	0	0	1	6	0	2
7	3	0	0	0	3	0	0
8	1	0	0	2	2	0	1
9	2	0	1	0	0	0	0
10	0	0	0	0	0	0	0
11	0	0	0	0	1	0	1

Table 7: Distribution of serialization size (number of serialized tweets) under each criterion

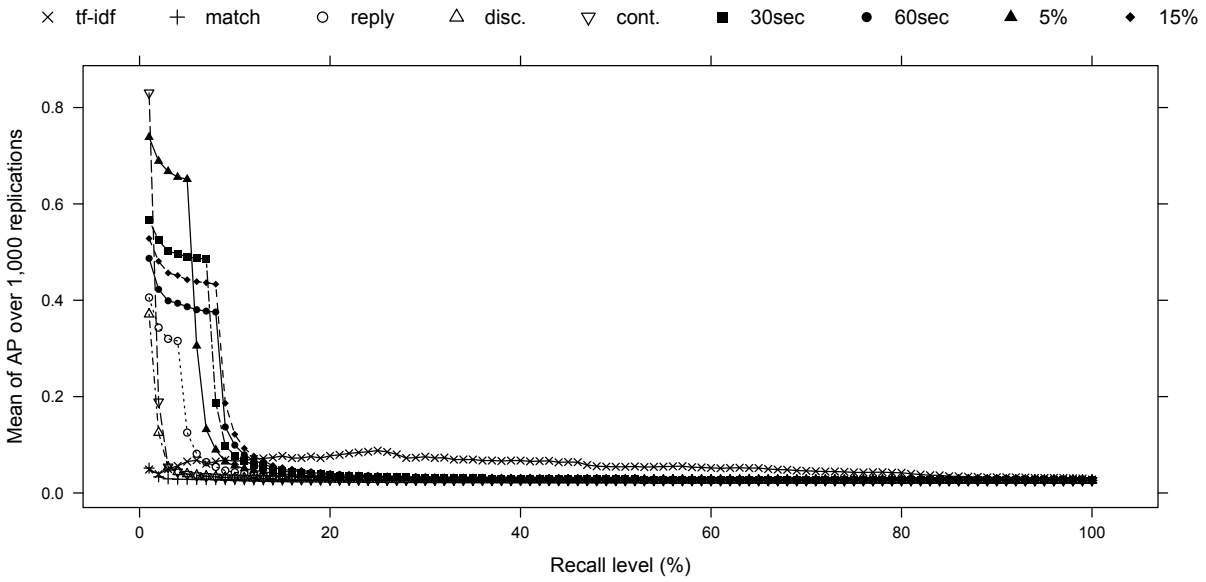


Figure 2: Means of average precision rates of all methods for the topic *Snowpiercer*

Recall	Baselines:		Repl.	Disc.	Cont.	Time difference threshold			
level $\leq$	Match	tf-idf	rela.	mark.	mark.	30sec	60sec	5%	15%
5%	.0342	.0518	.3019	.1266	.2313	.5158	.4178	**.6804**	.4720
10%	.0309	.0588	.1798	.0801	.1324	.3916	.3459	**.4050**	.3976
25%	.0284	.0695	.0920	.0494	.0702	.1824	.1665	.1847	**.1894**
50%	.0273	.0685	.0602	.0382	.0486	.1062	.0986	.1070	**.1103**
100%	.0268	.0556	.0434	.0322	.0375	.0666	.0628	.0669	**.0687**

Table 8: Means of average precision rates (at recall level up to 5%, 50%, and 100%) on various serialization criteria for the topic *Snowpiercer* (Results in boldface represent the best results among the methods.)

Serialization methods also perform better than the tf-idf baseline for induced topics, as shown in Figure 3 and Table 9. In particular, *Reply* and *Discourse markers*, which were far from the best for *Snowpiercer*, serve well for other topics such as *Movie* in general, *Politics*, and *Gender/Relationships*.

The precision of *Reply* for the topic *Movie* is exceptionally high, partly because the data were initially collected from users who were interested in films. *Reply relation* is dependent on the choice of the data, in that it is determined by interaction between users, not by a single user's tweets. If data are collected from users friendly with each other, *Reply* will serialize many tweets. On the contrary, if data contains some users while leaving out their friends, replies to these friends are not serialized by *Reply* criteria.

Discourse markers give a precision of higher than 50% for the topic *Politics*, which is likely to be discussed in more formal expressions using various conjunctions.

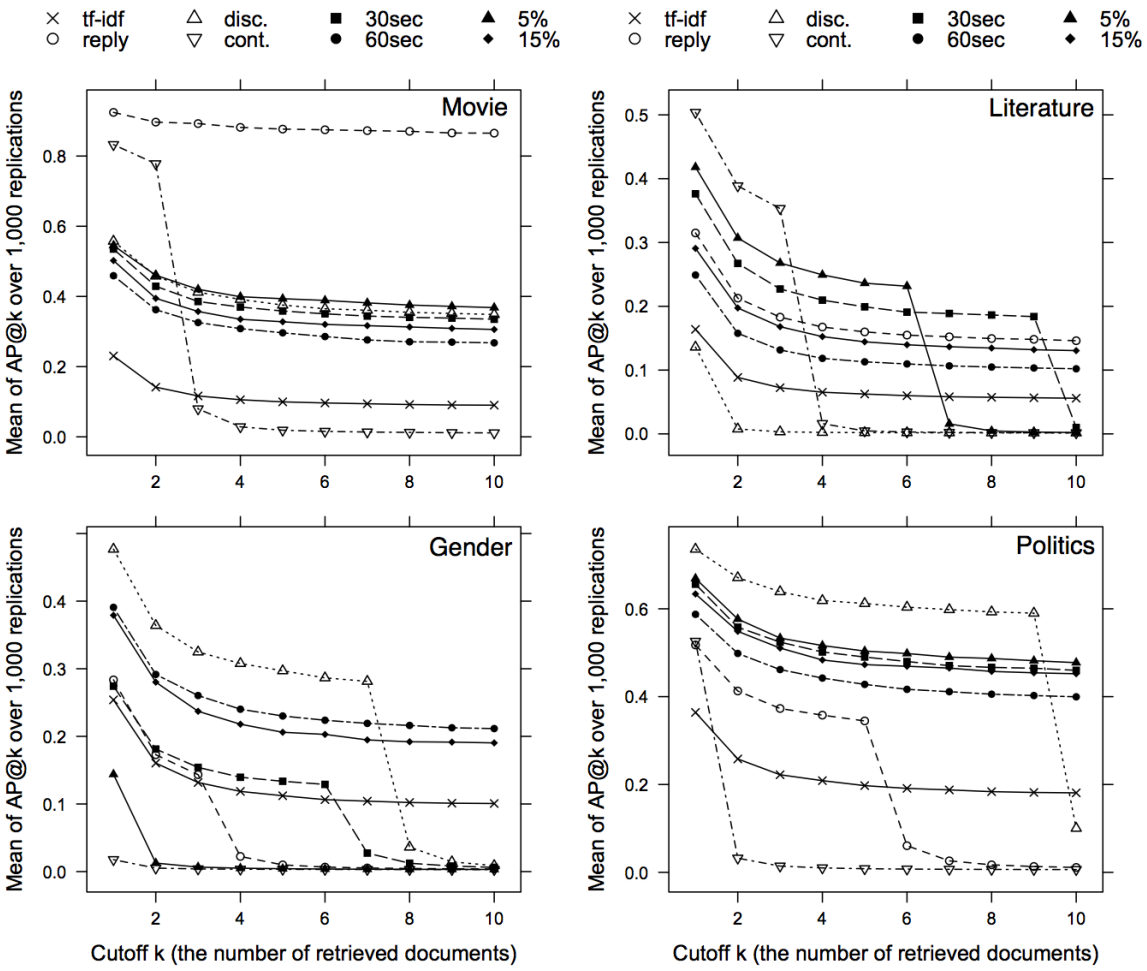


Figure 3: Means of average precision rates of all methods for the induced topics

	Baselines:		Repl.	Disc.	Cont.	Time difference threshold			
	Match	tf-idf	rela.	mark.	mark.	30sec	60sec	5%	15%
Mov.	.0134	.1139	**.8855**	.3925	.1791	.3787	.3123	.4161	.3435
Lit.	.0026	.0759	.1804	.0171	.1293	**.2005**	.1287	.1719	.1601
Gen.	.0048	.1287	.0653	.2424	.0050	.1092	**.2476**	.0187	.2297
Pol.	.0090	.2176	.2135	**.5762**	.0625	.5072	.4453	.5234	.4948

Table 9: Means of Average Precision rates at cutoff $k = 10$ of baselines and different serialization criteria for induced topics (Results in boldface represent the most accurate results of the topic among the methods.)

In the topics *Literature* and *Gender/Relationships*, average precision scores are at most 25%, which possibly results from the fact that the seed words for these topics consist of general terms only, while those of the other two topics include proper nouns such as movie titles or politicians' names. This is less a problem of the topic itself but rather one of data selection, which focused on users tweeting about films, and so the set of seed words will vary according to differences in data collection.

5 Conclusion

In this paper, we found that tweets with an implicit topic can be found more effectively by considering whether or not they are serialized with some tweet containing the overt keyword. Our experiments show that Tweet Serialization can be detected using various criteria such as reply relations between users, presence of discourse or continuation markers, and temporal proximity under the same authorship. Our

original purpose was to find as various opinions on a given topic as possible, but we expect the methods used here will be helpful for other tasks, including topic discovery and sentiment analysis, by setting more exact document boundaries in microblog data. The method we proposed is for Korean Twitter data, where tweet serialization is observed frequently, particularly among influential users, but it is also applicable to other languages with similar phenomena.

In future work, we will investigate methods for the evaluation of the results of Tweet Serialization and combine tf-idf methods with Tweet Serialization criteria. Furthermore, we aim at verifying the applicability of the results of this study with regard to more various users and more topics.

References

Paul Burstein. February 7, 2013. Older Tweets in search results. *The Official Twitter Blog.* https://blog.twitter.com/2013/now-showing-older-tweets-in-search-results.

Jack Dorsey. September 25, 2007. Tracking Twitter. *The Official Twitter Blog.* https://blog.twitter.com/2007/tracking-twitter.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin.* 76(5): 378–382.

Liangjie Hong and Brian D. Davison. 2010. Empirical study of topic modeling in Twitter. *SOMA 2010: The Proceedings of the First Workshop on Social Media Analytics.* 80–88.

Beom-mo Kang. 1999. *Hankukeui theksuthu cangluwa ene thukseng* [Text genres and linguistic characteristics in Korean]. Korea University Press, Seoul, Korea.

Esteban Kozak. November 19, 2013. New ways to search on Twitter. *The Official Twitter Blog.* https://blog.twitter.com/2013/new-ways-to-search-on-twitter.

Subhabrata Mukherjee and Pushpak Bhattacharyya. 2012. Sentiment analysis in Twitter with lightweight discourse analysis. *COLING 2012: The 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers.* 1847–1864.

Rishabh Mehrotra, Scoot Sanner, Wray Buntine and Lexing Xie. 2013. Improving LDA topic models for microblogs via tweet pooling and automatic labeling. *SIGIR '13; The 36th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 889–892.

Graham Neubig and Kevin Duh. 2013. How much is said in a Tweet? A multilingual, information-theoretic perspective. *AAAI Spring Symposium: Analyzing Microtext, Volume SS-13-01 of AAAI Technical Report*.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*. 24(5): 513–523.

Swapna Somasundaran. 2010. *Discourse-level Relations for Opinion Analysis*. Ph.D Thesis, University of Pittsburgh.

Biz Stone. August 22, 2007. Searching Twitter. *The Official Twitter Blog.* https://blog.twitter.com/2007/searching-twitter.

Biz Stone. December 23, 2008. Finding Nemo — Or, name search is back! *The Official Twitter Blog.* https://blog.twitter.com/2008/finding-nemo%E2%80%94or-name-search-back.

Biz Stone. February 18, 2009. Testing a more integrated search experience. *The Official Twitter Blog.* https://blog.twitter.com/2009/testing-more-integrated-search-experience.

Biz Stone. April 03, 2009. The discovery engine is coming. *The Official Twitter Blog.* https://blog.twitter.com/2009/discovery-engine-coming.

Biz Stone. April 30, 2009. Twitter search for everyone! *The Official Twitter Blog.* https://blog.twitter.com/2009/twitter-search-everyone.

Twitter. April 4, 2011. Discover new accounts and search like a pro. *The Official Twitter Blog.* https://blog.twitter.com/2011/discover-new-accounts-and-search-pro.

Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. 2010. TwitterRank: Finding topic-sensitive influential twitterers. *WSDM '10: Proceedings of the Third ACM International Conference on Web Search and Data Mining*. 261–270.

Florian Wolf, Edward Gibson and Timothy Desmet. 2004. Discourse coherence and pronoun resolution. *Language and Cognitive Processes*, 19(6): 665–675.

Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts

Cícero Nogueira dos Santos
Brazilian Research Lab
IBM Research
cicerons@br.ibm.com

Maíra Gatti
Brazilian Research Lab
IBM Research
mairacg@br.ibm.com

Abstract

Sentiment analysis of short texts such as single sentences and Twitter messages is challenging because of the limited contextual information that they normally contain. Effectively solving this task requires strategies that combine the small text content with prior knowledge and use more than just bag-of-words. In this work we propose a new deep convolutional neural network that exploits from character- to sentence-level information to perform sentiment analysis of short texts. We apply our approach for two corpora of two different domains: the Stanford Sentiment Treebank (SSTb), which contains sentences from movie reviews; and the Stanford Twitter Sentiment corpus (STS), which contains Twitter messages. For the SSTb corpus, our approach achieves state-of-the-art results for single sentence sentiment prediction in both binary positive/negative classification, with 85.7% accuracy, and fine-grained classification, with 48.3% accuracy. For the STS corpus, our approach achieves a sentiment prediction accuracy of 86.4%.

1 Introduction

The advent of online social networks has produced a crescent interest on the task of sentiment analysis for short text messages (Go et al., 2009; Barbosa and Feng, 2010; Nakov et al., 2013). However, sentiment analysis of short texts such as single sentences and and microblogging posts, like Twitter messages, is challenging because of the limited amount of contextual data in this type of text. Effectively solving this task requires strategies that go beyond bag-of-words and extract information from the sentence/message in a more disciplined way. Additionally, to fill the gap of contextual information in a scalable manner, it is more suitable to use methods that can exploit prior knowledge from large sets of unlabeled texts.

In this work we propose a deep convolutional neural network that exploits from character- to sentence-level information to perform sentiment analysis of short texts. The proposed network, named Character to Sentence Convolutional Neural Network (CharSCNN), uses two convolutional layers to extract relevant features from words and sentences of any size. The proposed network can easily explore the richness of word embeddings produced by unsupervised pre-training (Mikolov et al., 2013). We perform experiments that show the effectiveness of CharSCNN for sentiment analysis of texts from two domains: movie review sentences; and Twitter messages (tweets). CharSCNN achieves state-of-the-art results for the two domains. Additionally, in our experiments we provide information about the usefulness of unsupervised pre-training; the contribution of character-level features; and the effectiveness of sentence-level features to detect negation.

This work is organized as follows. In Section 2, we describe the proposed the Neural Network architecture. In Section 3, we discuss some related work. Section 4 details our experimental setup and results. Finally, in Section 5 we present our final remarks.

2 Neural Network Architecture

Given a sentence, CharSCNN computes a score for each sentiment label $\tau \in T$. In order to score a sentence, the network takes as input the sequence of words in the sentence, and passes it through

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 69–78, Dublin, Ireland, August 23-29 2014.

a sequence of layers where features with increasing levels of complexity are extracted. The network extracts features from the character-level up to the sentence-level. The main novelty in our network architecture is the inclusion of two convolutional layers, which allows it to handle words and sentences of any size.

2.1 Initial Representation Levels

The first layer of the network transforms words into real-valued feature vectors (embeddings) that capture morphological, syntactic and semantic information about the words. We use a fixed-sized word vocabulary V^{wrd}, and we consider that words are composed of characters from a fixed-sized character vocabulary V^{chr}. Given a sentence consisting of N words $\{w_1, w_2, ..., w_N\}$, every word w_n is converted into a vector $u_n = [r^{wrd}; r^{wch}]$, which is composed of two sub-vectors: the *word-level embedding* $r^{wrd} \in \mathbb{R}^{d^{wrd}}$ and the *character-level embedding* $r^{wch} \in \mathbb{R}^{cl_u^0}$ of w_n. While word-level embeddings are meant to capture syntactic and semantic information, character-level embeddings capture morphological and shape information.

2.1.1 Word-Level Embeddings

Word-level embeddings are encoded by column vectors in an embedding matrix $W^{wrd} \in \mathbb{R}^{d^{wrd} \times |V^{wrd}|}$. Each column $W_i^{wrd} \in \mathbb{R}^{d^{wrd}}$ corresponds to the word-level embedding of the i-th word in the vocabulary. We transform a word w into its word-level embedding r^{wrd} by using the matrix-vector product:

$$r^{wrd} = W^{wrd} v^w \tag{1}$$

where v^w is a vector of size $\left|V^{wrd}\right|$ which has value 1 at index w and zero in all other positions. The matrix W^{wrd} is a parameter to be learned, and the size of the word-level embedding d^{wrd} is a hyper-parameter to be chosen by the user.

2.1.2 Character-Level Embeddings

Robust methods to extract morphological and shape information from words must take into consideration all characters of the word and select which features are more important for the task at hand. For instance, in the task of sentiment analysis of Twitter data, important information can appear in different parts of a hash tag (e.g., "*#SoSad*", "*#ILikeIt*") and many informative adverbs end with the suffix "*ly*" (e.g. "*beautifully*", "*perfectly*" and "*badly*"). We tackle this problem using the same strategy proposed in (dos Santos and Zadrozny, 2014), which is based on a convolutional approach (Waibel et al., 1989). As depicted in Fig. 1, the convolutional approach produces local features around each character of the word and then combines them using a max operation to create a fixed-sized character-level embedding of the word.

Given a word w composed of M characters $\{c_1, c_2, ..., c_M\}$, we first transform each character c_m into a character embedding r_m^{chr}. Character embeddings are encoded by column vectors in the embedding matrix $W^{chr} \in \mathbb{R}^{d^{chr} \times |V^{chr}|}$. Given a character c, its embedding r^{chr} is obtained by the matrix-vector product:

$$r^{chr} = W^{chr} v^c \tag{2}$$

where v^c is a vector of size $\left|V^{chr}\right|$ which has value 1 at index c and zero in all other positions. The input for the convolutional layer is the sequence of character embeddings $\{r_1^{chr}, r_2^{chr}, ..., r_M^{chr}\}$.

The convolutional layer applies a matrix-vector operation to each window of size k^{chr} of successive windows in the sequence $\{r_1^{chr}, r_2^{chr}, ..., r_M^{chr}\}$. Let us define the vector $z_m \in \mathbb{R}^{d^{chr} k^{chr}}$ as the concatenation of the character embedding m, its $(k^{chr} - 1)/2$ left neighbors, and its $(k^{chr} - 1)/2$ right neighbors[1]:

$$z_m = \left(r^{chr}_{m-(k^{chr}-1)/2}, ..., r^{chr}_{m+(k^{chr}-1)/2}\right)^T$$

[1]We use a special *padding character* for the characters with indices outside of the word boundaries.

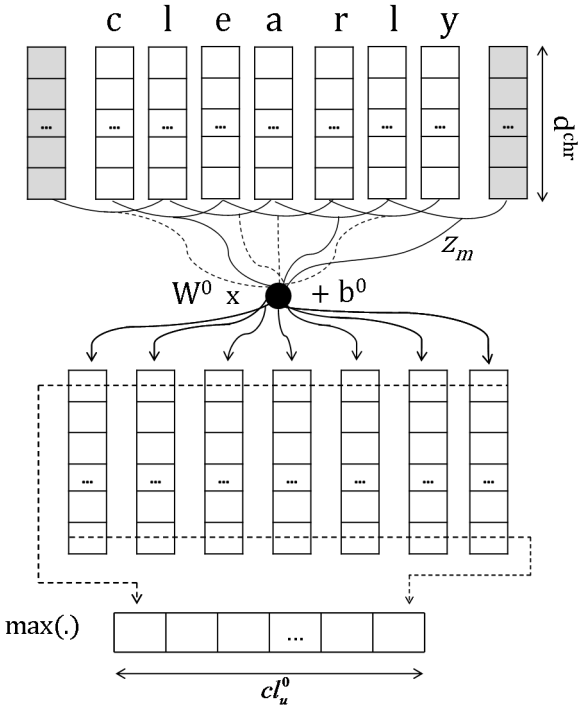


Figure 1: Convolutional approach to character-level feature extraction.

The convolutional layer computes the j-th element of the vector $r^{wch} \in \mathbb{R}^{cl_u^0}$, which is the character-level embedding of w, as follows:

$$[r^{wch}]_j = \max_{1<m<M} \left[W^0 z_m + b^0\right]_j \tag{3}$$

where $W^0 \in \mathbb{R}^{cl_u^0 \times d^{chr} k^{chr}}$ is the weight matrix of the convolutional layer. The same matrix is used to extract local features around each character window of the given word. Using the max over all character windows of the word, we extract a "global" fixed-sized feature vector for the word.

Matrices W^{chr} and W^0, and vector b^0 are parameters to be learned. The size of the character vector d^{chr}, the number of convolutional units cl_u^0 (which corresponds to the size of the character-level embedding of a word), and the size of the character context window k^{chr} are hyper-parameters.

2.2 Sentence-Level Representation and Scoring

Given a sentence x with N words $\{w_1, w_2, ..., w_N\}$, which have been converted to joint word-level and character-level embedding $\{u_1, u_2, ..., u_N\}$, the next step in CharSCNN consists in extracting a sentence-level representation r_x^{sent}. Methods to extract a sentence-wide feature set most deal with two main problems: sentences have different sizes; and important information can appear at any position in the sentence. We tackle these problems by using a convolutional layer to compute the sentence-wide feature vector r^{sent}. This second convolutional layer in our neural network architecture works in a very similar way to the one used to extract character-level features for words. This layer produces local features around each word in the sentence and then combines them using a max operation to create a fixed-sized feature vector for the sentence.

The second convolutional layer applies a matrix-vector operation to each window of size k^{wrd} of successive windows in the sequence $\{u_1, u_2, ..., u_N\}$. Let us define the vector $z_n \in \mathbb{R}^{(d^{wrd}+cl_u^0)k^{wrd}}$ as the concatenation of a sequence of k^{wrd} embeddings, centralized in the n-th word[2]:

$$z_n = \left(u_{n-(k^{wrd}-1)/2}, ..., u_{n+(k^{wrd}-1)/2}\right)^T$$

[2]We use a special *padding token* for the words with indices outside of the sentence boundaries.

The convolutional layer computes the j-th element of the vector $r^{sent} \in \mathbb{R}^{cl_u^1}$ as follows:

$$[r^{sent}]_j = \max_{1<n<N} \left[W^1 z_n + b^1\right]_j \tag{4}$$

where $W^1 \in \mathbb{R}^{cl_u^1 \times (d^{wrd} + cl_u^0)k^{wrd}}$ is the weight matrix of the convolutional layer. The same matrix is used to extract local features around each word window of the given sentence. Using the max over all word windows of the sentence, we extract a "global" fixed-sized feature vector for the sentence. Matrix W^1 and vector b^1 are parameters to be learned. The number of convolutional units cl_u^1 (which corresponds to the size of the sentence-level feature vector), and the size of the word context window k^{wrd} are hyper-parameters to be chosen by the user.

Finally, the vector r_x^{sent}, the "global' feature vector of sentence x, is processed by two usual neural network layers, which extract one more level of representation and compute a score for each sentiment label $\tau \in T$:

$$s(x) = W^3 h(W^2 r_x^{sent} + b^2) + b^3 \tag{5}$$

where matrices $W^2 \in \mathbb{R}^{hl_u \times cl_u^1}$ and $W^3 \in \mathbb{R}^{|T| \times hl_u}$, and vectors $b^2 \in \mathbb{R}^{hl_u}$ and $b^3 \in \mathbb{R}^{|T|}$ are parameters to be learned. The transfer function $h(.)$ is the hyperbolic tangent. The number of hidden units hl_u is a hyper-parameter to be chosen by the user.

2.3 Network Training

Our network is trained by minimizing a negative likelihood over the training set D. Given a sentence x, the network with parameter set θ computes a score $s_\theta(x)_\tau$ for each sentiment label $\tau \in T$. In order to transform these scores into a conditional probability distribution of labels given the sentence and the set of network parameters θ, we apply a softmax operation over the scores of all tags $\tau \in T$:

$$p\left(\tau | x, \theta\right) = \frac{e^{s_\theta(x)_\tau}}{\sum_{\forall i \in T} e^{s_\theta(x)_i}} \tag{6}$$

Taking the log, we arrive at the following conditional log-probability:

$$\log p\left(\tau | x, \theta\right) = s_\theta(x)_\tau - \log \left(\sum_{\forall i \in T} e^{s_\theta(x)_i} \right) \tag{7}$$

We use stochastic gradient descent (SGD) to minimize the negative log-likelihood with respect to θ:

$$\theta \mapsto \sum_{(x,y) \in D} -\log p(y | x, \theta) \tag{8}$$

where (x, y) corresponds to a sentence in the training corpus D and y represents its respective label.

The backpropagation algorithm is a natural choice to efficiently compute gradients of network architectures such as the one proposed in this work (Lecun et al., 1998; Collobert, 2011). In order to perform our experiments, we implement the proposed CharSCNN architecture using the *Theano* library (Bergstra et al., 2010). *Theano* is a versatile Python library that allows the efficient definition, optimization, and evaluation of mathematical expressions involving multi-dimensional arrays. We use *Theano*'s automatic differentiation capabilities in order to implement the backpropagation algorithm.

3 Related Work

There are a few works on neural network architectures for sentiment analysis. In (Socher et al., 2011), the authors proposed a semi-supervised approach based on recursive autoencoders for predicting sentiment distributions. The method learns vector space representation for multi-word phrases and exploits the recursive nature of sentences. In (Socher et al., 2012), it is proposed a matrix-vector recursive neural network model for semantic compositionality, which has the ability to learn compositional vector

representations for phrases and sentences of arbitrary length. The vector captures the inherent meaning of the constituent, while the matrix captures how the meaning of neighboring words and phrases are changed. In (Socher et al., 2013b) the authors propose the Recursive Neural Tensor Network (RNTN) architecture, which represents a phrase through word vectors and a parse tree and then compute vectors for higher nodes in the tree using the same tensor-based composition function. Our approach differ from these previous works because it uses a feed-forward neural network instead of a recursive one. Moreover, it does not need any input about the syntactic structure of the sentence.

Regarding convolutional networks for NLP tasks, in (Collobert et al., 2011), the authors use a convolutional network for the semantic role labeling task with the goal avoiding excessive task-specific feature engineering. In (Collobert, 2011), the authors use a similar network architecture for syntactic parsing. CharSCNN is related to these works because they also apply convolutional layers to extract sentence-level features. The main difference in our neural network architecture is the addition of one convolutional layer to extract character features.

In terms of using intra-word information in neural network architectures for NLP tasks, Alexandrescu et al. (2006) present a factored neural language model where each word is represented as a vector of features such as stems, morphological tags and cases and a single embedding matrix is used to look up all of these features. In (Luong et al., 2013), the authors use a recursive neural network (RNN) to explicitly model the morphological structures of words and learn morphologically-aware embeddings. Lazaridou et al. (Lazaridou et al., 2013) use compositional distributional semantic models, originally designed to learn meanings of phrases, to derive representations for complex words, in which the base unit is the morpheme. In (Chrupala, 2013), the author proposes a simple recurrent network (SRN) to learn continuous vector representations for sequences of characters, and use them as features in a conditional random field classifier to solve a character level text segmentation and labeling task. The main advantage of our approach to extract character-level features is it flexibility. The convolutional layer allows the extraction of relevant features from any part of the word and do not need handcrafted inputs like stems and morpheme lists (dos Santos and Zadrozny, 2014).

4 Experimental Setup and Results

4.1 Sentiment Analysis Datasets

We apply CharSCNN for two different corpora from two different domains: movie reviews and Twitter posts. The movie review dataset used is the recently proposed Stanford Sentiment Treebank (SSTb) (Socher et al., 2013b), which includes fine grained sentiment labels for 215,154 phrases in the parse trees of 11,855 sentences. In our experiments we focus in sentiment prediction of complete sentences. However, we show the impact of training with sentences and phrases instead of only sentences.

The second labeled corpus we use is the Stanford Twitter Sentiment corpus (STS) introduced by (2009). The original training set contains 1.6 million tweets that were automatically labeled as positive/negative using emoticons as noisy labels. The test set was manually annotated by Go et al. (2009). In our experiments, to speedup the training process we use only a sample of the training data consisting of 80K (5%) randomly selected tweets. We also construct a development set by randomly selecting 16K tweets from Go et al.'s training set. In Table 1, we present additional details about the two corpora.

Dataset	Set	# sentences / tweets	# classes
SSTb	Train	8544	5
	Dev	1101	5
	Test	2210	5
STS	Train	80K	2
	Dev	16K	2
	Test	498	3

Table 1: Sentiment Analysis datasets.

4.2 Unsupervised Learning of Word-Level Embeddings

Word-level embeddings play a very important role in the CharSCNN architecture. They are meant to capture syntactic and semantic information, which are very important to sentiment analysis. Recent work has shown that large improvements in terms of model accuracy can be obtained by performing unsupervised pre-training of word embeddings (Collobert et al., 2011; Luong et al., 2013; Zheng et al., 2013; Socher et al., 2013a). In our experiments, we perform unsupervised learning of word-level embeddings using the *word2vec* tool[3], which implements the *continuous bag-of-words* and *skip-gram* architectures for computing vector representations of words (Mikolov et al., 2013).

We use the December 2013 snapshot of the English Wikipedia corpus as a source of unlabeled data. The Wikipedia corpus has been processed using the following steps: (1) removal of paragraphs that are not in English; (2) substitution of non-western characters for a special character; (3) tokenization of the text using the tokenizer available with the Stanford POS Tagger (Manning, 2011); (4) and removal of sentences that are less than 20 characters long (including white spaces) or have less than 5 tokens. Like in (Collobert et al., 2011) and (Luong et al., 2013), we lowercase all words and substitute each numerical digit by a 0 (e.g., *1967* becomes *0000*). The resulting clean corpus contains about 1.75 billion tokens.

When running the *word2vec* tool, we set that a word must occur at least 10 times in order to be included in the vocabulary, which resulted in a vocabulary of 870,214 entries. To train our word-level embeddings we use *word2vec*'s skip-gram method with a context window of size 9. The training time for the English corpus is around 1h10min using 12 threads in a Intel® Xeon® E5-2643 3.30GHz machine.

In our experiments, we do not perform unsupervised pre-training of character-level embeddings, which are initialized by randomly sampling each value from an uniform distribution: $\mathcal{U}(-r, r)$, where $r = \sqrt{\frac{6}{|V^{chr}| + d^{chr}}}$. There are 94 different characters in the SSTb corpus and 453 different characters in the STS corpus. Since the two character vocabularies are relatively small, it has been possible to learn reliable character-level embeddings using the labeled training corpora. The raw (not lowercased) words are used to construct the character vocabularies, which allows the network to capture relevant information about capitalization.

4.3 Model Setup

We use the development sets to tune the neural network hyper-parameters. Many different combinations of hyper-parameters can give similarly good results. We spent more time tuning the learning rate than tuning other parameters, since it is the hyper-parameter that has the largest impact in the prediction performance. The only two parameters with different values for the two datasets are the learning rate and the number of units in the convolutional layer that extract sentence features. This provides some indication on the robustness of our approach to multiple domains. For both datasets, the number of training epochs varies between five and ten. In Table 2, we show the selected hyper-parameter values for the two labeled datasets.

Parameter	Parameter Name	SSTb	STS
d^{wrd}	Word-Level Embeddings dimension	30	30
k^{wrd}	Word Context window	5	5
d^{chr}	Char. Embeddings dimension	5	5
k^{chr}	Char. Context window	3	3
cl_u^0	Char. Convolution Units	10	50
cl_u^1	Word Convolution Units	300	300
hl_u	Hidden Units	300	300
λ	Learning Rate	0.02	0.01

Table 2: Neural Network Hyper-Parameters

[3] https://code.google.com/p/word2vec/

In order to assess the effectiveness of the proposed character-level representation of words, we compare the proposed architecture CharSCNN with an architecture that uses only word embeddings. In our experiments, SCNN represents a network which is fed with word representations only, i.e, for each word w_n its embedding is $u_n = r^{wrd}$. For SCNN, we use the same NN hyper-parameters values (when applicable) shown in Table 2.

4.4 Results for SSTb Corpus

In Table 3, we present the result of CharSCNN and SCNN for different versions of the SSTb corpus. Note that SSTb corpus is a sentiment treebank, hence it contains sentiment annotations for all phrases in all sentences in the corpus. In our experiments, we check whether using examples that are single phrases, in addition to complete sentences, can provide useful information for training the proposed NN. However, in our experiments the test set always includes only complete sentences. In Table 3, the column *Phrases* indicates whether all phrases (*yes*) or only complete sentences (*no*) in the corpus are used for training. The *Fine-Grained* column contains prediction results for the case where 5 sentiment classes (labels) are used (*very negative*, *negative*, *neutral*, *positive*, *very positive*). The *Positive/Negative* column presents prediction results for the case of binary classification of sentences, i.e, the neutral class is removed, the two negative classes are merged as well as the two positive classes.

Model	**Phrases**	**Fine-Grained**	**Positive/Negative**
CharSCNN	yes	**48.3**	**85.7**
SCNN	yes	**48.3**	85.5
CharSCNN	no	43.5	82.3
SCNN	no	43.5	82.0
RNTN (Socher et al., 2013b)	yes	45.7	85.4
MV-RNN (Socher et al., 2013b)	yes	44.4	82.9
RNN (Socher et al., 2013b)	yes	43.2	82.4
NB (Socher et al., 2013b)	yes	41.0	81.8
SVM (Socher et al., 2013b)	yes	40.7	79.4

Table 3: Accuracy of different models for fine grained (5-class) and binary predictions using SSTb.

In Table 3, we can note that CharSCN and SCNN have very similar results in both fine-grained and binary sentiment prediction. These results suggest that the character-level information is not much helpful for sentiment prediction in the SSTb corpus. Regarding the use of phrases in the training set, we can note that, even not explicitly using the syntactic tree information when performing prediction, CharSCNN and SCNN benefit from the presence of phrases as training examples. This result is aligned with Socher et al.'s (2013b) suggestion that information of sentiment labeled phrases improves the accuracy of other classification algorithms such as support vector machines (SVM) and naive Bayes (NB). We believe that using phrases as training examples allows the classifier to learn more complex phenomena, since sentiment labeled phrases give the information of how words (phrases) combine to form the sentiment of phrases (sentences). However, it is necessary to perform more detailed experiments to confirm this conjecture.

Regarding the fine-grained sentiment prediction, our approach provides an absolute accuracy improvement of 2.6 over the RNTN approach proposed by (Socher et al., 2013b), which is the previous best reported result for SSTb. CharSCN, SCNN and Socher et al.'s RNTN have similar accuracy performance for binary sentiment prediction. Compared to RNTN, our method has the advantage of not needing the output of a syntactic parser when performing sentiment prediction. For comparison reasons, in Table 3 we also report Socher et al.'s (2013b) results for sentiment classifiers trained with recursive neural networks (RNN), matrix-vector RNN (MV-RNN), NB, and SVM algorithms.

Initializing word-embeddings using unsupervised pre-training gives an absolute accuracy increase of around 1.5 when compared to randomly initializing the vectors. The *Theano* based implementation of CharSCNN takes around 10 min. to complete one training epoch for the SSTb corpus with all phrases

and five classes. In our experiments, we use 4 threads in a Intel® Xeon® E5-2643 3.30GHz machine.

4.5 Results for STS Corpus

In Table 4, we present the results of CharSCNN and SCNN for sentiment prediction using the STS corpus. As expected, character-level information has a greater impact for Twitter data. Using unsupervised pre-training, CharSCNN provides an absolute accuracy improvement of 1.2 over SCNN. Additionally, initializing word-embeddings using unsupervised pre-training gives an absolute accuracy increase of around 4.5 when compared to randomly initializing the word-embeddings.

In Table 4, we also compare CharSCNN performance with other approaches proposed in the literature. In (Speriosu et al., 2011), a label propagation (LProp) approach is proposed, while Go et al. (2009) use maximum entropy (MaxEnt), NB and SVM-based classifiers. CharSCNN outperforms the previous approaches in terms of prediction accuracy. As far as we know, 86.4 is the best prediction accuracy reported so far for the STS corpus.

Model	**Accuracy (unsup. pre-training)**	**Accuracy (random word embeddings)**
CharSCNN	**86.4**	81.9
SCNN	85.2	82.2
LProp (Speriosu et al., 2011)	84.7	
MaxEnt (Go et al., 2009)	83.0	
NB (Go et al., 2009)	82.7	
SVM (Go et al., 2009)	82.2	

Table 4: Accuracy of different models for binary predictions (positive/negative) using STS Corpus.

4.6 Sentence-level features

In figures 2 and 3 we present the behavior of CharSCNN regarding the sentence-level features extracted for two cases of negation, which are correctly predicted by CharSCNN. We choose these cases because negation is an important issue in sentiment analysis. Moreover, the same sentences are also used as illustrative examples in (Socher et al., 2013b). Note that in the convolutional layer, 300 features are first extracted for each word. Then the $\max$ operator selects the 300 features which have the largest values among the words to construct the sentence-level feature set r^{sent}. Figure 2 shows a positive sentence (left) and its negation. We can observe that in both versions of the sentence, the extracted features concentrate mainly around the main topic, *"film"*, and the part of the phrase that indicates sentiment (*"liked"* and *"did 'nt like"*). Note in the left chart that the word *"liked"* has a big impact in the set of extracted features. On the other hand, in the right chart, we can see that the impact of the word *"like"* is reduced because of the negation *"did 'nt"*, which is responsible for a large part of the extracted features.

In Figure 3 a similar behavior can be observed. While the very negative expression *"incredibly dull"* is responsible for 69% of the features extracted from the sentence in the left, its negation *"definitely not dull"*, which is somewhat more positive, is responsible for 77% of the features extracted from the sentence in the chart at right . These examples indicate CharSCNN's robustness to handle negation, as well as its ability to capture information that is important to sentiment prediction.

5 Conclusions

In this work we present a new deep neural network architecture that jointly uses character-level, word-level and sentence-level representations to perform sentiment analysis. The main contributions of the paper are: (1) the idea of using convolutional neural networks to extract from character- to sentence-level features; (2) the demonstration that a feed-forward neural network architecture can be as effective as RNTN (Socher et al., 2013a) for sentiment analysis of sentences; (3) the definition of new state-of-the-art results for SSTb and STS corpora.

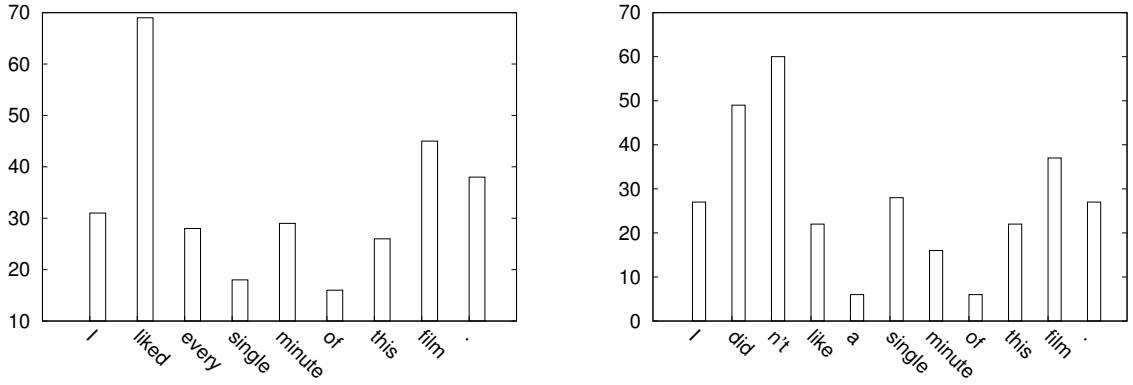


Figure 2: Number of local features selected at each word when forming the sentence-level representation. In this example, we have a positive sentence (left) and its negation (right).

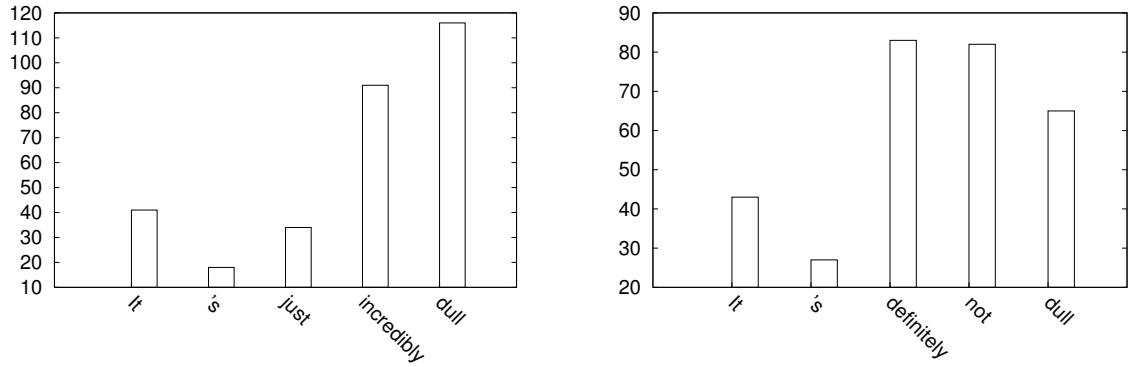


Figure 3: Number of local features selected at each word when forming the sentence-level representation. In this example, we have a negative sentence (left) and its negation (right).

As future work, we would like to analyze in more detail the role of character-level representations for sentiment analysis of tweets. Additionally, we would like to check the impact of performing the unsupervised pre-training step using texts from the specific domain at hand.

References

Andrei Alexandrescu and Katrin Kirchhoff. 2006. Factored neural language models. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 1–4, New York City, USA, June.

Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 36–44.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*.

Grzegorz Chrupala. 2013. Text segmentation with character-level text embeddings. In *Proceedings of the ICML workshop on Deep Learning for Audio, Speech and Language Processing*.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

R. Collobert. 2011. Deep learning for efficient discriminative parsing. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 224–232.

Cícero Nogueira dos Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning, JMLR: W&CP volume 32*, Beijing, China.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. Technical report, Stanford University.

Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. 2013. Compositional–ly derived representations of morphologically complex words in distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1517–1526.

Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324.

Minh-Thang Luong, Richard Socher, and Christopher D. Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Conference on Computational Natural Language Learning*, Sofia, Bulgaria.

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing'11, pages 171–189.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations*.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 312–320, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of theConference on Empirical Methods in Natural Language Processing*, pages 1201–1211.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013a. Parsing with compositional vector grammars. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Michael Speriosu, Nikita Sudan, Sid Upadhyay, and Jason Baldridge. 2011. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the First Workshop on Unsupervised Learning in NLP*, EMNLP, pages 53–63.

A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang. 1989. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(3):328–339.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *Proceedings of the Conference on Empirical Methods in NLP*, pages 647–657.

Joint Inference and Disambiguation of Implicit Sentiments via Implicature Constraints

Lingjia Deng[1], **Janyce Wiebe**[1,2], **Yoonjung Choi**[2]
[1]Intelligent Systems Program, University of Pittsburgh
[2]Department of Computer Science, University of Pittsburgh
lid29@pitt.edu, wiebe@cs.pitt.edu, yjchoi@cs.pitt.edu

Abstract

This paper addresses implicit opinions expressed via inference over explicit sentiments and events that positively/negatively affect entities (*goodFor/badFor*, *gfbf* events). We incorporate the inferences developed by implicature rules into an optimization framework, to jointly improve sentiment detection toward entities and disambiguate components of gfbf events. The framework simultaneously beats the baselines by more than 10 points in F-measure on sentiment detection and more than 7 points in accuracy on gfbf polarity disambiguation.

1 Introduction

Previous work in NLP on sentiment analysis has mainly focused on explicit sentiments. However, as noted in (Deng and Wiebe, 2014), many opinions are expressed implicitly, as shown by this example:

> Ex(1) The reform would lower health care costs, which would be a tremendous positive change across the entire health-care system.

There is an explicit positive sentiment toward the event of "reform lower costs". However, in expressing this sentiment, the writer also implies he is negative toward the "costs", since he's happy to see the costs being decreased. Moreover, the writer may be positive toward "reform" since it contributes to the "lower" event. Such inferences may be seen as opinion-oriented *implicatures* (i.e., defeasible inferences)[1].

We develop a set of rules for inferring and detecting implicit sentiments from explicit sentiments and events such as "lower" (Wiebe and Deng, 2014). In (Deng et al., 2013), we investigate such events, defining a ***badFor (bf)*** event to be an event that negatively affects the theme and a ***goodFor (gf)*** event to be an event that positively affects the theme of the event.[2] Here, "lower" is a bf event. According to their annotation scheme, ***goodFor/badFor (gfbf)*** events have NP agents and themes (though the agent may be implicit), and the polarity of a gf event may be changed to bf by a ***reverser*** (and vice versa).

The ultimate goal of this work is to utilize gfbf information to improve detection of the writer's sentiments toward entities mentioned in the text. However, this requires resolving several ambiguities: (Q1) Given a document, which spans are gfbf events? (Q2) Given a gfbf text span, what is its polarity, gf or bf? (Q3) Is the polarity of a gfbf event being reversed? (Q4) Which NP in the sentence is the agent and which is the theme? (Q5) What are the writer's sentiments toward the agent and theme, positive or negative? Fortunately, the implicature rules in (Deng and Wiebe, 2014) define dependencies among these ambiguities. As in Ex(1), the sentiments toward the agent and theme, the sentiment toward the gfbf event (positive or negative), and the polarity of the gfbf event (gf or bf) are all interdependent. Thus, rather than having to take a pipeline approach, we are able to develop an optimization framework which exploits these interdependencies to jointly resolve the ambiguities.

Specifically, we develop local detectors to analyze the four individual components of gfbf events, (Q2)-(Q5) above. Then, we propose an Integer Linear Programming (ILP) framework to conduct global

[1]Specifically, we focus on *generalized conversational implicature* (Grice, 1967; Grice, 1989).

[2]Compared to (Deng et al., 2013), we change the term "object" to "theme" as the later is more appropriate for this task.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 79–88, Dublin, Ireland, August 23-29 2014.

inference, where the gfbf events and their components are variables and the interdependencies defined by the implicature rules are encoded as constraints over relevant variables in the framework. The reason we do not address (Q1) is that the gold standard we use for evaluation contains sentiment annotations only toward the agents and themes of gfbf events. We are only able to evaluate true hits of gfbf events. Thus, the input to the system is the set of the text spans marked as gfbf events in the corpus. The results show that, compared to the local detectors, the ILP framework improves sentiment detection by more than 10 points in F-measure and disambiguating gfbf polarity by more than 7 points in the accuracy, without any loss in accuracy for other two components.

2 Related Work

Most work in sentiment analysis focuses on classifying explicit sentiments and extracting explicit opinion expressions, holders and targets (Wiebe et al., 2005; Johansson and Moschitti, 2013; Yang and Cardie, 2013). There is some work investigating features that directly indicate implicit sentiments (Zhang and Liu, 2011; Feng et al., 2013). In contrast, we focus on how we can bridge between explicit and implicit sentiments via inference. To infer the implicit sentiments related to gfbf events, some work mines various syntactic patterns (Choi and Cardie, 2008), proposes linguistic templates (Zhang and Liu, 2011; Anand and Reschke, 2010; Reschke and Anand, 2011), or generates a lexicon of patient polarity verbs (Goyal et al., 2013). Different from their work, which do not cover all cases relevant to gfbf events, (Deng and Wiebe, 2014) defines a generalized set of implicature rules and proposes a graph-based model to achieve sentiment propagation between the agents and themes of gfbf events. However, that system requires all of the gfbf information (Q1)-(Q4) to be input from the manual annotations; the only ambiguity it resolves is sentiments toward entities. In contrast, the method in this paper tackles four ambiguities simultaneously. Further, as we will see below in Section 6, the improvement over the local detectors by the current method is greater than that by the previous method, even though it operates over the noisy output of local components automatically.

Different from pipeline architectures, where each step is computed independently, joint inference has often achieved better results. Roth and Yih (2004) formulate the task of information extraction using Integer Linear Programming (ILP). Since then, ILP has been widely used in various tasks in NLP, including semantic role labeling (Punyakanok et al., 2004; Punyakanok et al., 2008; Das et al., 2012), joint extraction of opinion entities and relations (Choi et al., 2006; Yang and Cardie, 2013), co-reference resolution (Denis and Baldridge, 2007), and summarization (Martins and Smith, 2009). The most similar ILP model to ours is (Somasundaran and Wiebe, 2009), which improves opinion polarity classification using discourse constraints in an ILP model. However, their work addresses discourse relations among explicit opinions in different sentences.

3 GoodFor/BadFor Event and Implicature

This work addresses sentiments toward, in general, states and events which positively or negatively affect entities. Deng et al. (2013) (hereafter DCW) identify a clear case that occurs frequently in opinion sentences, namely the gfbf events mentioned above. As defined in DCW, a *gf* event is an event that positively affects the theme of the event and a *bf* event is an event that negatively affects the theme. According to the annotation schema, *gfbf* events have NP agents and themes (though the agent may be implicit). In the sentence "President Obama passed the bill", the agent of the gf "passed" is "President Obama" and the theme is "the bill". In the sentence "The bill was denied", the agent of the bf "was denied" is implicit. The polarity of a gf event may be changed to bf by a *reverser* (and vice versa). For example, in "The reform will not worsen the economy," "not" is a reverser and it reverses the polarity from bf to gf.[3]

The constraints we encode in the ILP framework described below are based on implicature rules in (Deng and Wiebe, 2014). Table 1 gives two rule schemas, each of which defines four specific rules. In

[3]DCW also introduce *retainers*. We don't analyze retainers in this work since they do not affect the polarity of gfbfs, and only 2.5% of gfbfs have retainers in the corpus.

	s(gfbf)	gfbf	→	s(agent)	s(theme)		s(gfbf)	gfbf	→	s(agent)	s(theme)
1	positive	gf	→	positive	positive	3	positive	bf	→	positive	negative
2	negative	gf	→	negative	negative	4	negative	bf	→	negative	positive

Table 1: Rule Schema 1 & Rule Schema 3 (Deng and Wiebe, 2014)

the table, *s(*α*)* = β means that the **writer's** sentiment toward α is β, where α is a gfbf event, or the agent or theme of a gfbf event, and β is either *positive* or *negative*. P → Q means to infer Q from P.

Applying the rules to Ex(1): the writer expresses a positive sentiment ("positive") toward a bf event ("lower"), thus matching Case 3 in Table 1. We infer that the writer is positive toward the agent ("reform") and negative toward the theme ("costs"). Two other rule schemas (not shown) make the same inferences as Rule Schemas 1 and 3 but in the opposite direction. As we can see, if two entities participate in a gf event, the writer has the same sentiment toward the agent and theme, while if two entities participate in a bf event, the writer has opposite sentiments toward them. Later we use this observation in our experiments.

4 Global Optimization Framework

Optimization is performed over two sets of variables. The first set is *GFBF*, containing a variable for each gfbf event in the document. The other set is *Entity*, containing a variable for each agent or theme candidate. Each variable k in *GFBF* has its corresponding agent and theme variables, i and j, in *Entity*. The three form a triple unit, $\langle i, k, j \rangle$. The set *Triple* consists of each $\langle i, k, j \rangle$, recording the correspondence between variables in *GFBF* and *Entity*. The goal of the framework is to assign optimal labels to variables in *Entity* and *GFBF*. We first introduce how we recognize candidates for agents and themes, then introduce the optimization framework, and then define local scores that are input to the framework.

4.1 Local Agents and Theme Candidates Detector

We extract two agent candidates and two theme candidates for each gfbf event (one each will ultimately be chosen by the ILP model).[4] We use syntax, and the output of the SENNA (Collobert et al., 2011) semantic role labeling tool. SENNA labels the A0 (subject), A1 (object), and A2 (indirect object) spans for each predicate, if possible. To extract the *semantic agent* candidate: If SENNA labels a span as A0 of the gfbf event, we consider it as the semantic agent; if there is no A0 but A1 is labeled, we consider A1; if there is no A0 or A1 but A2 is labeled, we consider A2. To extract the *syntactic agent* candidate, we find the nearest noun in front of the gfbf span, and then extract any other word that depends on the noun according to the dependency parse. Similarly, to extract the *semantic theme* candidate, we consider A1, A2, A0 in order. To extract the *syntactic theme* candidate, the same procedure is conducted as for the syntactic agent, but the nearest noun should be after the gfbf. If there is no A0, A1 or A2, then there is only one agent candidate, *implicit* and only one theme candidate, *null*. We treat a *null* theme as an incorrect span in the later evaluations. If the two agent (theme) candidate spans are the same, there is only one candidate.

4.2 Integer Linear Programming Framework

We use Integer Linear Programming (ILP) to assign labels to variables. Variables in *Entity* will be assigned *positive* or *negative*, representing the writer's sentiments toward them. We may have two candidate agents for a gfbf and that we will choose between them. Thus, only one agent is assigned a *positive* or *negative* label; the other is considered to be an incorrect agent of the gfbf (similarly for the theme candidates). Each variable in *GFBF* will be assigned the label *gf* or *bf*. Optionally, it may also be assigned the label *reversed*. Label *gf* or *bf* is the polarity of the gfbf event; *reversed* is assigned if the polarity is reversed (e.g., for "not harmed", the labels are *bf* and *reversed*).

The objective function of the ILP is:

[4]This framework is able to handle any number of candidates. The methods we tried using more candidates did not perform as well - the gain in recall was offset by larger losses in precision.

$$\min_{u_{1gf},u_{1bf}\ldots}\Big(-1*\sum_{i\in GFBF\cup Entity}\sum_{c\in L_i}p_{ic}u_{ic}\Big)+\sum_{\langle i,k,j\rangle\in Triple}\xi_{ikj}+\sum_{\langle i,k,j\rangle\in Triple}\delta_{ikj} \tag{1}$$

subject to

$$u_{ic}\in\{0,1\},\forall i,c\qquad \xi_{ikj},\delta_{ikj}\in\{0,1\},\forall\langle i,k,j\rangle\in Triple \tag{2}$$

where L_i is the set of labels given to $\forall i \in$ ***GFBF*** $\cup$ ***Entity***. If $i \in$ ***GFBF***, L_i is {gf, bf, reversed} ({*gf, bf, r*}, for short). If $i \in$ ***Entity***, L_i is {positive, negative} ({*pos, neg*}, for short). u_{ic} is a binary indicator representing whether the label c is assigned to the variable i. When an indicator variable is 1, the corresponding label is selected. p_{ic} is the score given by local detectors, introduced in the following sections. Variables ξ_{ikj} and δ_{ikj} are binary slack variables that correspond to the gfbf implicature constraints of $\langle i,k,j\rangle$. When a given slack variable is 1, the corresponding triple violates the implicature constraints. Minimizing the objective function could achieve two goals at the same time. The first part ($-1*\sum_i\sum_c p_{ic}u_{ic}$) tries to select a set of labels that maximize the scores given by the local detectors. The second part ($\sum_{ikj}\xi_{ikj}+\sum_{ikj}\delta_{ikj}$) aims at minimizing the cases where gfbf implicature constraints are violated. Here we do not force each triple to obey the implicature constraints, but to minimize the violating cases. For each variable, we have defined constraints:

$$\sum_{c\in L_{GFBF'}}u_{kc}=1,\forall k\in GFBF \tag{3}$$

$$\sum_{\substack{i\in Entity\\ \langle i,k,j\rangle\in Triple}}\sum_{c\in L_{Entity}}u_{ic}=1,\forall k\in GFBF \tag{4}$$

$$\sum_{\substack{j\in Entity,\\ \langle i,k,j\rangle\in Triple}}\sum_{c\in L_{Entity}}u_{jc}=1,\forall k\in GFBF \tag{5}$$

where $L_{GFBF'}$ in Equation (3) is a subset of L_{GFBF}, consisting of {*gf*, *bf*}. Equation (3) means a gfbf must be either gf or bf. But it is free to choose whether it is being reversed. Recall that we have two agent candidates ($a1$,$a2$) for a gfbf. Thus we have four agent indicators in Equation (4): $u_{a1,pos}$, $u_{a1,neg}$, $u_{a2,pos}$ and $u_{a2,neg}$. Equation (4) ensures that three of them are 0 and one of them is 1. For instance, $u_{a1,pos}$ assigned 1 means that candidate $a1$ is selected to be the agent span and pos is selected to be its polarity. In this way, the framework disambiguates the agent span and sentiment polarity simultaneously. (Similar comments apply for the theme candidates in Equation (5).)

According to the implicature rules in Table 1 in Section 3, the writer has the same sentiment toward entities in a gf relation. Thus, for each triple unit $\langle i,k,j\rangle$, the gf constraints are applied via the following:

$$|\sum_{i,\langle i,k,j\rangle}u_{i,pos}-\sum_{j,\langle i,k,j\rangle}u_{j,pos}|+|u_{k,gf}-u_{k,r}|<=1+\xi_{ikj},\forall k\in GFBF \tag{6}$$

$$|\sum_{i,\langle i,k,j\rangle}u_{i,neg}-\sum_{j,\langle i,k,j\rangle}u_{j,neg}|+|u_{k,gf}-u_{k,r}|<=1+\xi_{ikj},\forall k\in GFBF \tag{7}$$

We use $|u_{k,gf}-u_{k,r}|$ to represent whether this triple is gf. In Equation (6), if this value is 1, then the triple should follow the gf constraints. In that case, $\xi_{ikj}=0$ means that the triple doesn't violate the gf constraints, and $|\sum_i u_{i,pos}-\sum_j u_{j,pos}|$ must be 0. Further, in this case, $\sum_i u_{i,pos}$ and $\sum_j u_{j,pos}$ are constrained to be of the same value (both 1 or 0) – that is, entities i and j must be both positive or both not positive. However, if $\xi_{ikj}=1$, Equation (6) does not constrain the values of the variables at all. If $|u_{k,gf}-u_{k,r}|$ is 0, representing that the triple is not gf, then Equation (6) does not constrain the values of the variables. Similar comments apply to Equation (7).

In contrast, the writer has opposite sentiments toward entities in a bf relation.

$$|\sum_{i,\langle i,k,j\rangle}u_{i,pos}+\sum_{j,\langle i,k,j\rangle}u_{j,pos}-1|+|u_{k,bf}-u_{k,r}|<=1+\delta_{ikj},\forall k\in GFBF \tag{8}$$

$$|\sum_{i,\langle i,k,j\rangle}u_{i,neg}+\sum_{j,\langle i,k,j\rangle}u_{j,neg}-1|+|u_{k,bf}-u_{k,r}|<=1+\delta_{ikj},\forall k\in GFBF \tag{9}$$

We use $|u_{k,bf}-u_{k,r}|$ to represent whether this triple is bf. In Equation (8), if a triple is bf and the constraints are not violated, then $|\sum_i u_{i,pos}+\sum_j u_{j,pos}-1|$ must be 0. Further, in this case, $\sum_i u_{i,pos}$

	u_{gf}	u_{bf}	u_r	$\lvert u_{gf} - u_r \rvert$	$\lvert u_{bf} - u_r \rvert$		u_{gf}	u_{bf}	u_r	$\lvert u_{gf} - u_r \rvert$	$\lvert u_{bf} - u_r \rvert$
A	1	0	0	1	0	C	0	1	0	0	1
B	0	1	1	1	0	D	1	0	1	0	1

Table 2: Truth table of being reversed or not (k is omitted)

and $\sum_j u_{j,pos}$ are constrained to be of the opposite value – that is, if entity i is positive then entity j must not be positive. Similar comments apply to Equation (9).

Note that above we use $|u_{k,gf} - u_{k,r}|$ and $|u_{k,bf} - u_{k,r}|$ to represent whether a triple is gf or bf. In Table 2, we show that they always take opposite values and that they are consistent with the actual polarities. In Table 2, Case A means the triple is gf and Case B means the triple is bf but it is reversed. In both cases, $|u_{gf} - u_r| = 1$, indicating that the triple should follow the gf constraints. Similarly for Case C and Case D to follow the bf constraints.

4.3 Local GoodFor/BadFor Score: $p_{k,gf}, p_{k,bf}$

We utilize a sense-level gfbf lexicon by (Choi et al., 2014). In total there are 6,622 gf senses and 3,290 bf senses. The gf lexicon covers 64% of the gf words in the corpus and the bf lexicon covers 42% of the bf words. We then look up the gfbf span k in the gfbf lexicon. If k only appears in the gf lexicon, then $p_{k,gf} = 1 - \epsilon$ and $p_{k,bf} = \epsilon$. Here $\epsilon = 0.0001$, to prevent there being any 0 scores in our computation. If k only appears in the bf lexicon, then $p_{k,bf} = 1 - \epsilon$ and $p_{i,gf} = \epsilon$. If k appears in both the gf and bf lexicon, and there are a senses in the gf lexicon and b senses in the bf lexicon, then $p_{k,gf} = a/(a+b)$ and $p_{k,bf} = b/(a+b)$. If k is not in either lexicon, then $p_{k,gf} = p_{k,bf} = \epsilon$. If there is more than one word in the gfbf span, we take the maximum score.

4.4 Local Reversed Score: $p_{k,r}$

As introduced in Section 3, a reverser changes the polarity of a gfbf. First, we build reverser lexicons from Wilson's shifter lexicon (2008), namely the entries labeled as *genshifter*, *negation*, and *shiftneg*. We create two lexicons: one with the verbs and the other with the non-verb entries, excluding nouns, adjectives, and adverbs, since most non-verb reversers are prepositions or subordinating conjunctions. There are 219 reversers in the entire corpus; 134 (61.19%) are instances of words in one of the two lexicons. Based on the lexicon, we categorize reversers into three classes. Examples are shown below.

Ex(2) They will **not** be able to water down your coverage.
Ex(3) ... how a massive new bureaucracy will cut costs **without** hurting the old and the helpless.
Ex(4) The new law includes new rules to **prevent** insurance companies from overcharging patients.

Negation: An instance in this category is "not" in Ex(2). If any word in the gfbf span has a *neg* dependency relation according to the Stanford dependency parser, then we consider the gfbf to be negated (i.e., reversed). In this case the path between the negator and the gfbf is labeled *neg* and the length of the path is one.

Other Non-Verb: This category consists of words such as "without" in Ex(3) (others are "never" and "few", etc). These words lower the extent of the gfbf event. We look in the sentence for instances of words in the non-verb reverser lexicon, which are not tagged as noun, verb, adj, or adv. For any found, we examine the path in the dependency parse between the potential reverser and the gfbf span. If the path has at least one of *advmod*, *pcomp*, *cc*, *xcomp*, *nsubj*, *neg* and the length of the path is less than four (learnt from development set), the event is considered to be reversed.

Verb: In Ex(4), the verb "prevent" stops the gfbf event "overcharging" from happening. We call such words *Verb* reverser (others are "prohibit" and "ban", etc). We look in the sentence for instances of words in the verb reverser lexicon. For any that appear before the gfbf span in the sentence, if the path has at least one of *xcomp*, *pcomp*, *obj* and the length of the path is less than four, then the event is reversed. For the triple ⟨companies, overcharging, patients⟩ in Ex(4), though it is reversed by "prevent", the agent of the reverser, which is "law", is different from the agent of the gfbf, which is "companies", so the bf

within the "overcharging" event is not reversed.[5] Though we extract the *Verb* reversers to evaluate the performance of recognizing a reverser, in the optimization framework, gfbf events with *Verb* reversers are not considered to be reversed, since almost all *Verb* reversers introduce new agents.

Different from other scores, $p_{k,r}$ could be negative. According to the heuristics above, the probability of a gfbf event being reversed decreases as the length of the path increases. We define $p_{k,r}$ so it is inversely proportional to the length of the path. Further, to make sense of a gfbf triple ⟨agent, gfbf, theme⟩, where, e.g., the local detectors label it ⟨pos, bf, pos⟩, the framework is choosing the smaller one from (a) $-1 * p_{k,r} * u_{k,r}$ (it has a reverser) versus (b) $1 * \xi_{ikj}$ (it is an exception to the rules). The framework assigns $u_{k,r} = 0$ and $\xi_{ikj} = 1$ if $-1 * p_{k,r} > 1$. It assigns $u_{k,r} = 1$ and $\xi_{ikj} = 0$ if $-1 * p_{k,r} <= 1$. For gfbf events which have *Negation* or *Other Non-verb* reversers, since we use the length four as a threshold in the heuristics above, we define $p_{k,r} = \frac{1}{d} - \frac{5}{4}$, so that $-1 * p_{k,r} = \frac{5}{4} - \frac{1}{d} > 1$ if $d > 4$. For gfbf events for which no reverser word appears in the sentence, or those which only have *Verb* reversers, $p_{k,r} = -1 * \frac{5}{4}$ (so $-1 * p_{k,r} > 1$), so that the framework chooses case (b) (choosing the gfbf event to be not reversed).

4.5 Local Sentiment Score: $p_{i,pos}, p_{i,neg}$

In the corpus of DCW, only the writer's sentiments toward the agents and the themes of gfbf events are annotated. Thus, since there are many false negatives of sentiments toward entities, the corpus does not support training a classifier. Therefore, we adopt the same local sentiment detector from (Deng and Wiebe, 2014), using available resources to detect writer's sentiments toward all agent and theme candidates.[6] The sentiment scores range from 0.5 to 1.

5 Co-reference In the Framework

So far the constraints in the framework are within a gfbf triple. Consider the following example:

Ex(5) **The reform** will decrease the healthcare costs and improve the medical qualify as expected.

The two gfbfs, "decrease" and "improve" have the same agent, "reform". Thus, if there is more than one gfbf in a sentence, and the path between the two gfbfs in dependency parse contains only *conj* or *xcomp*, and there is no other noun between the latter gfbf and the conjunction, we assume the two agents are the same and the sentiments toward them should be the same. Thus, for any $i, j \in$ *Entity*, if i, j co-refer[7], or they are the same agent as described above, $Coref(i, j) = 1$ (otherwise 0). We add two more constraints, similar to the gf constraints in Equations (6) and (7), as shown in Equation (10) and (11). where ν_{ij} is a slack variable, $e(i)$ is the set of agent/theme candidates linked to the same gfbf as i is. If $Coref(i, j) = 0$, Equations (10) and (11) do not constrain the variables. The objective function in Equation (12) is updated to incorporate these new constraints.

$$|\sum_{e(i)} u_{i,pos} - \sum_{e(j)} u_{j,pos}| + Coref(i,j) <= 1 + \nu_{ij}, \forall i, j \in Entity \tag{10}$$

$$|\sum_{e(i)} u_{i,neg} - \sum_{e(j)} u_{j,neg}| + Coref(i,j) <= 1 + \nu_{ij}, \forall i, j \in Entity \tag{11}$$

$$\min_{u_{1gf}, u_{1bf} \cdots} \left(-1 * \sum_{i \in GFBF \cup Entity} \sum_{c \in L_i} p_{ic} u_{ic} \right) + \sum_{\langle i,k,j \rangle \in Triple} \xi_{ikj} + \sum_{\langle i,k,j \rangle \in Triple} \delta_{ikj} + \sum_{i,j \in Entity} \nu_{ij} \tag{12}$$

6 Experiment and Performance

In this section we introduce the data we use, the baseline methods, the evaluations and the results. In addition, we give examples illustrating how opinion inference may improve performances.

[5] DCW defines here is a *triple chain*: ⟨law, prevent ⟨companies, overcharging, patients⟩⟩. The reverser is changing the polarity between "law" and "patients", but it does not change the polarity between "companies" and "patients".

[6] We use Opinion Extractor (Johansson and Moschitti, 2013) , opinionFinder (Wilson et al., 2005), MPQA subjectivity lexicon (Wilson et al., 2005), General Inquirer (Stone et al., 1966) and a connotation lexicon (Feng et al., 2013), to detect writer's sentiments toward all agent and theme candidates, and all gfbf events. We adopt Rule 1 and Rule 3 to infer from the sentiment toward event to the sentiment toward theme. Then we conduct a majority voting based on the results.

[7] We use the co-reference resolution system from (Stoyanov et al., 2010).

6.1 Experiment Data

We use the "Affordable Care Act" corpus of DCW, consisting of 134 online editorials and blogs. In total, there are 1,762 annotated triples, out of which 692 are gf or retainers and 1,070 are bf or reversers. From the writer's perspective, 1,495 noun phrases are annotated positive, 1,114 noun phrases are negative and the remaining 8 are neutral. This indicates that there are many opinions in the corpus. Out of 134 documents in the corpus, 3 do not have any annotation. 6 are used as a development set to develop the heuristics in Sections 4 and 5. We use the remaining 125 for the experiments.

6.2 Baseline Methods and Evaluation Metrics

We compare the output of the global optimization framework with the outputs of baseline systems built from the local detectors in Section 4. For the gfbf polarity and reverser ambiguities, the local detectors directly provide a disambiguation result. For the agent/theme span and sentiment ambiguities, the local sentiment detector assigns positive and negative scores to each candidate. The framework chooses among the combined options. Thus, for comparison, we build a baseline system that combines the outputs of the local agent/theme candidate detector and the local sentiment detector.

Recall from Section 4, a variable $k \in$ *GFBF* has two agent candidates, $a1$ and $a2 \in$ *Entity*. Together there are four binary indicator variables: $u_{a1,pos}$, $u_{a1,neg}$, $u_{a2,pos}$ and $u_{a2,neg}$. Among these indicator variables whose corresponding local scores (e.g., $p_{a1,pos}$ is the score of $u_{a1,pos}$) are larger than 0.5, the baseline system (denoted *Local*) chooses the one with the largest local sentiment score. If there is a tie, it prefers the variable representing the semantic candidate. If there is still a tie, it chooses the variable representing the majority polarity (positive). If all the local scores of the four variables are 0.5 (neutral), *Local* fails to recognize any sentiment for that entity, so it assigns 0 to all the indicator variables. *Local+coref* takes the maximum local score of the entities if they co-ref, and assigns each entity the maximum score before disambiguation.

Another baseline, *Majority*, always chooses the semantic candidate and the majority polarity.

To evaluate the performance in detecting sentiment, we use precision, recall, and F-measure. We do not take into account any agent or theme manually annotated as neutral (there are only 8).

$$\text{P} = \frac{\#(\text{auto=gold \& gold!=neutral})}{\#\text{auto!=neutral}} \qquad \text{Accuracy} = \text{R} = \frac{\#(\text{auto=gold \& gold!=neutral})}{\#\text{gold!=neutral}} \qquad \text{F} = \frac{2*\text{P}*\text{R}}{\text{P+R}} \tag{13}$$

In the equations, *auto* is the system's output and *gold* is the gold-standard label from annotations. Since we don't take into account any *neutral* agent or theme, *#gold!=neutral* equals to all nodes in the experiment set. Thus accuracy is equal to recall. We only report recall here. Here we have two definitions of *auto=gold*: (1) **Strict** evaluation means that, by saying *auto=gold*, the agent/theme must have the same polarity and must be the same NP as the gold standard, and (2) **Relaxed** evaluation means the agent/theme has the same polarity as the gold standard, regardless whether the span is correct or not.

Note that according to DCW, an implicit agent isn't annotated with any sentiment. Thus, for an implicit agent in *gold*, if *auto* outputs the span "implicit", we treat it as a correct span with correct polarity, regardless what sentiment *auto* gives to it. If *auto* outputs any span other than "implicit", we treat it as a wrong span with wrong polarity, regardless of its sentiment as well. For the theme span, if *auto* outputs a "null" theme candidate, we treat it as a wrong span but we evaluate its sentiment according to *gold*.

To evaluate extracting candidate span, we use accuracy. The baseline for this task always chooses the semantic candidate. To evaluate gfbf polarity and reverser, we also use accuracy.

Note that although we evaluate the performance in different tasks separately, the framework resolves all the ambiguities at the same time.

6.3 Results

We report the performance results for **(A) sentiment detection** in Table 3, on two sets. One is the subset containing the agents and themes where *auto* has the correct spans with *gold*. The other is the set of all agents and themes. As shown in Table 3, *ILP* significantly improves performance, approximately 10-20 points on F-measure over different baselines. Though *Local* has a competitive precision with

		correct span subset			whole set, strict eval			whole set, relaxed eval		
		P	R	F	P	R	F	P	R	F
1	ILP	0.6421	0.6421	0.6421	0.4401	0.4401	0.4401	0.5939	0.5939	0.5939
2	Local	0.6409	0.3332	0.4384	0.4956	0.2891	0.3652	0.5983	0.3490	0.4408
3	ILP+coref	0.6945	0.6945	0.6945	0.4660	0.4660	0.4660	0.6471	0.6471	0.6471
4	Local+coref	0.6575	0.3631	0.4678	0.5025	0.3103	0.3836	0.6210	0.3834	0.4741
5	Majority	0.5792	0.5792	0.5792	0.3862	0.3862	0.3862	0.5462	0.5462	0.5462

Table 3: Performances of sentiment detection

ILP, it has a much lower recall. That means the local sentiment detector cannot recognize implicit sentiments toward most entities. But *ILP* is able to recognize more entities correctly. By adding *coref*, performance improves for both *ILP* and *Local*. In comparison to (Deng and Wiebe, 2014), our current method improves more in F-measure (2.43 points more) over local sentiment detector than the earlier work, even though the earlier work takes the manual annotations of all the gfbf information as input.

In terms of the other tasks: For **(B) agent/theme span**, the baseline achieves 66.67% in accuracy, compared to 68.54% and 67.10% for *ILP* and *ILP+coref*, respectively. For **(C) gfbf polarity**, the baseline has an accuracy of 70.68%, whereas *ILP* achieves 77.25% and *ILP+coref* achieves 77.47%, respectively, both 7 points higher. This improvement is interesting because it represents cases in which the optimization framework is able to ***infer*** the correct polarity even though the gfbf span is not recognized by the local detector (i.e., the span isn't in the gfbf lexicon). For **(D) reverser**, the baseline is 88.07% in accuracy. *ILP* and *ILP+coref* are competitive with the baseline: 89% and 88.07% respectively. Note that both our local detector and *ILP* surpass the majority class (not reversed) which has an accuracy of 86.60%.

Following (Akkaya et al., 2009), since ILP is unsupervised without multiple runs, we adopt McNemar's test to measure statistical significance of our improvements (Dietterich, 1998). In Table 3, the improvements in recalls of Line 1 over 2, Line 3 over 4, and Lines 1&3 over 5 are statistically significant at the $p < .001$ level. The improvements of Line 3 over 1 are statistically significant at the $p < .005$ level. For accuracy of gfbf polarity, the improvement is significant at the $p < .001$ level.

6.4 Examples

This sections gives simplified examples to illustrate how the framework can improve over the local detectors. The explicit sentiment clues referred to in this section are from MPQA lexicon.

Ex(6) The reform would curb skyrocketing costs in the long run.

The local sentiment detector assigns "costs" *negative* due to the single sentiment clue, "skyrocketing". Since the agent and theme are in a bf triple, and the writer is *negative* toward that theme, we can infer the writer is *positive* toward the agent. This illustrates how we improve recall on sentiments.

Ex(7) The supposedly costly reform will curb skyrocketing costs in the long run.

In Ex(7), agent "reform" is labeled *negative* because "costly" is a negative clue in the lexicon. ("supposedly" is not in it.) However, in Ex(7), it is actually positive. The agent's negative score is 0.6, and its positive score is 0.5 due to the absence of a positive clue. Since the theme is *negative* too, by the bf constraints, we **expect** to see a positive agent. If we were to assign *negative* to the agent, the objective function would have -0.6 subjectivity score and +1 in violation penalty, together giving +0.4. If we assign *positive*, the subjectivity score is -0.5, and there is no violation, resulting in a total score of -0.5. Thus, the framework correctly chooses the positive label. This shows how we can improve precision on sentiments.

Ex(8) The great reform will curb skyrocketing costs in the long run.

In this case, the agent is positive and the theme is negative. If the gfbf word "curb" is not in the lexicon, we could still infer its polarity. Given that the entities in the triple have different sentiments, to not violate

the implicature rules, the framework will assign it *bf*, or assign it *gf* along with *reversed*. However, there is no reverser word in the sentence, so the reversed score $p_r = -\frac{5}{4}$. The framework will assign the reverser indicator $u_r = 0$, in order to avoid a gain in the objective function by $-1 * p_r * u_r$. Thus the framework assigns the label *bf* to "curb". This is how the framework can improve the accuracy of recognizing gfbf polarity.

7 Conclusion

The ultimate goal of this work is to utilize gfbf information to improve detection of the writer's sentiments toward entities mentioned in the text. Using an unsupervised optimization framework that incorporates gfbf implicature rules as constraints, our method improves over local sentiment recognition by almost 20 points in F-measure and over all sentiment baselines by over 10 points in F-measure. The global optimization framework jointly infers the polarity of gfbf events, whether or not they are reversed, which candidate NPs are the agent and theme, and the writer's sentiments toward them. In addition to beating the baselines for sentiment detection, the framework significantly improves the accuracy of gfbf polarity disambiguation. This work not only automatically utilizes gfbf information to improve sentiment detection, it also proposes a framework for jointly solving various ambiguities related to gfbf events.

Acknowledgement This work was supported in part by DARPA-BAA-12-47 DEFT grant. We would like to thank the anonymous reviewers for their helpful feedback.

References

Cem Akkaya, Janyce Wiebe, and Rada Mihalcea. 2009. Subjectivity word sense disambiguation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 190–199, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pranav Anand and Kevin Reschke. 2010. Verb classes as evaluativity functor classes. In *Interdisciplinary Workshop on Verbs. The Identification and Representation of Verb Features*.

Yejin Choi and Claire Cardie. 2008. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 793–801, Honolulu, Hawaii, October. Association for Computational Linguistics.

Yejin Choi, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 431–439, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yoonjung Choi, Janyce Wiebe, and Lingjia Deng. 2014. Lexical acquisition for opinion inference: A sense-level lexicon of benefactive and malefactive events. In *5th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November.

Dipanjan Das, André FT Martins, and Noah A Smith. 2012. An exact dual decomposition algorithm for shallow semantic parsing with constraints. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 209–217. Association for Computational Linguistics.

Lingjia Deng and Janyce Wiebe. 2014. Sentiment propagation via implicature constraints. In *Meeting of the European Chapter of the Association for Computational Linguistics (EACL-2014)*.

Lingjia Deng, Yoonjung Choi, and Janyce Wiebe. 2013. Benefactive/malefactive event and writer attitude annotation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 120–125, Sofia, Bulgaria, August. Association for Computational Linguistics.

Pascal Denis and Jason Baldridge. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 236–243, Rochester, New York, April. Association for Computational Linguistics.

Thomas G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10:1895–1923.

Song Feng, Jun Sak Kang, Polina Kuznetsova, and Yejin Choi. 2013. Connotation lexicon: A dash of sentiment beneath the surface meaning. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Sofia, Bulgaria, Angust. Association for Computational Linguistics.

Amit Goyal, Ellen Riloff, and Hal Daum III. 2013. A computational model for plot units. *Computational Intelligence*, 29(3):466–488.

Herbert Paul Grice. 1967. Logic and conversation. The William James lectures.

Herbert Paul Grice. 1989. *Studies in the Way of Words*. Harvard University Press.

Richard Johansson and Alessandro Moschitti. 2013. Relational features in fine-grained opinion analysis. *Computational Linguistics*, 39(3).

André F. T. Martins and Noah a. Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing - ILP '09*, pages 1–9, Morristown, NJ, USA. Association for Computational Linguistics.

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1346. Association for Computational Linguistics.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Kevin Reschke and Pranav Anand. 2011. Extracting contextual evaluativity. In *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, pages 370–374, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *CONLL*.

Swapna Somasundaran and Janyce Wiebe. 2009. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 226–234, Suntec, Singapore, August. Association for Computational Linguistics.

P.J. Stone, D.C. Dunphy, M.S. Smith, and D.M. Ogilvie. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, Cambridge.

Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010. Coreference resolution with reconcile. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 156–161, Stroudsburg, PA, USA. Association for Computational Linguistics.

Janyce Wiebe and Lingjia Deng. 2014. An account of opinion implicatures. arXiv:1404.6491v1 [cs.CL].

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language ann. *Language Resources and Evaluation*, 39(2/3):164–210.

Theresa Wilson, Janyce Wiebe, , and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLP/EMNLP*, pages 347–354.

Theresa Wilson. 2008. *Fine-grained subjectivity analysis*. Ph.D. thesis, Doctoral Dissertation, University of Pittsburgh.

Bishan Yang and Claire Cardie. 2013. Joint Inference for Fine-grained Opinion Extraction. In *Proceedings of ACL*, pages 1640–1649.

Lei Zhang and Bing Liu. 2011. Identifying noun product features that imply opinions. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 575–580, Portland, Oregon, USA, June. Association for Computational Linguistics.

Group Non-negative Matrix Factorization with Natural Categories for Question Retrieval in Community Question Answer Archives

Guangyou Zhou, Yubo Chen, Daojian Zeng, and Jun Zhao
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing 100190, China
{gyzhou,yubo.chen,djzeng,jzhao}@nlpr.ia.ac.cn

Abstract

Community question answering (CQA) has become an important service due to the popularity of CQA archives on the web. A distinctive feature is that CQA services usually organize questions into a hierarchy of natural categories. In this paper, we focus on the problem of question retrieval and propose a novel approach, called group non-negative matrix factorization with natural categories (GNMFNC). This is achieved by learning the category-specific topics for each category as well as shared topics across all categories via a group non-negative matrix factorization framework. We derive an efficient algorithm for learning the factorization, analyze its complexity, and provide proof of convergence. Experiments are carried out on a real world CQA data set from Yahoo! Answers. The results show that our proposed approach significantly outperforms various baseline methods and achieves the state-of-the-art performance for question retrieval.

1 Introduction

Community question answering (CQA) such as Yahoo! Answers[1] and Quora[2], has become an important service due to the popularity of CQA archives on the web. To make use of the large-scale questions and their answers, it is critical to have functionality of helping users to retrieve previous answers (Duan et al., 2008). Typically, such functionality is achieved by first retrieving the historical questions that best match a user's queried question, and then using answers of these returned questions to answer the queried question. This is what we called *question retrieval* in this paper.

The major challenge for question retrieval, as for most information retrieval tasks, is the *lexical gap* between the queried questions and the historical questions in the archives. For example, if a queried question contains the word "company" but a relevant historical question instead contains the word "firm", then there is a mismatch and the historical question may not be easily distinguished from an irrelevant one. To solve the *lexical gap* problem, most researchers focused on translation-based approaches since the relationships between words (or phrases) can be explicitly modeled through word-to-word (or phrases) translation probabilities (Jeon et al., 2005; Riezler et al., 2007; Xue et al., 2008; Lee et al., 2008; Bernhard and Gurevych, 2009; Zhou et al., 2011; Singh, 2012). However, these existing methods model the relevance ranking without considering the category-specific and shared topics with natural categories, it is not clear whether this information is useful for question retrieval.

A distinctive feature of question-answer pairs in CQA is that CQA services usually organize questions into a hierarchy of natural categories. For example, Yahoo! Answers contains a hierarchy of 26 categories at the first level and more than 1262 subcategories at the leaf level. When a user asks a question, the user is typically required to choose a category label for the question from a predefined hierarchy. Questions in the predefined hierarchy usually share certain generic topics while questions in different categories have their specific topics. For example, questions in categories "Arts & Humanities" and "Beauty & Style" may share the generic topic of "dance" but they also have the category-specific topics of "poem" and "wearing", respectively.

[1] http://answers.yahoo.com/

[2] http://www.quora.com/

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 89–98, Dublin, Ireland, August 23-29 2014.

Inspired by the above observation, we propose a novel approach, called group non-negative matrix factorization with natural categories (GNMFNC). GNMFNC assumes that there exists a set of category-specific topics for each of the category, and there also exists a set of shared topics for all of the categories. Each question in CQA is specified by its category label, category-specific topics, as well as shared topics. In this way, the large-scale question retrieval problem can be decomposed into small-scale subproblems.

In GNMFNC, questions in each category are represented as a term-question matrix. The term-question matrix is then approximated as the product of two matrices: one matrix represents the category-specific topics as well as the shared topics, and the other matrix denotes the question representation based on topics. An objective function is defined to measure the goodness of prediction of the data with the model. Optimization of the objective function leads to the automatic discovery of topics as well as the topic representation of questions. Finally, we calculate the relevance ranking between the queried questions and the historical questions in the latent topic space.

Past studies by (Cao et al., 2009; Cao et al., 2010; Ming et al., 2010; Cai et al., 2011; Ji et al., 2012; Zhou et al., 2013) confirmed a significant retrieval improvement by adding the natural categories into various existing retrieval models. However, all these previous work regarded natural categories individually without considering the relationships among them. On the contrary, this paper can effectively capture the relationships between the shared aspects and the category-specific individual aspects with natural categories via a group non-negative matrix factorization framework. Also, our work models the relevance ranking in the latent topic space rather than using the existing retrieval models. To date, no attempts have been made regarding group non-negative matrix factorization in studies of question retrieval, which remains an under-explored area.

The remainder of this paper is organized as follows. Section 2 describes our proposed group non-negative matrix factorization with natural categories for question retrieval. Section 3 presents the experimental results. In Section 4, we conclude with ideas for future research.

2 Group Non-negative Matrix Factorization with Natural Categories

2.1 Problem Formulation

In CQA, all questions are usually organized into a hierarchy of categories. When a user asks a question, the user is typically required to choose a category label for the question from a predefined hierarchy of categories. Hence, each question in CQA has a category label. Suppose that we are given a question collection $\mathcal{D}$ in CQA archive with size N, containing terms from a vocabulary $\mathcal{V}$ with size M. A question d is represented as a vector $\mathbf{d} \in \mathbb{R}^M$ where each entry denotes the weight of the corresponding term, for example tf-idf is used in this paper. Let $C = \{c_1, c_2, \cdots, c_P\}$ denote the set of categories (subcategories) of question collection $\mathcal{D}$, where P is the number of categories (subcategories). The question collection $\mathcal{D}$ is organized into P groups according to their category labels and can be represented as $\mathcal{D} = \{\mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_P\}$. $\mathbf{D}_p = \{\mathbf{d}_1^{(p)}, \cdots, \mathbf{d}_{N_p}^{(p)}\} \in \mathbb{R}^{M \times N_p}$ is the term-question matrix corresponding to category c_p, in which each row stands for a term and each column stands for a question. N_p is the number of questions in category c_p such that $\sum_{p=1}^{P} N_p = N$.

Let $\mathbf{U}_p' = [\mathbf{U}_s, \mathbf{U}_p] \in \mathbb{R}^{M \times (K_s + K_p)}$ be the term-topic matrix corresponding to category c_p, where K_s is the number of shared topics, K_p is the number of category-specific topics corresponding to category c_p, and $p \in [1, P]$. Term-topic matrix $\mathbf{U}_s$ can be represented as $\mathbf{U}_s = [\mathbf{u}_1^{(s)}, \cdots, \mathbf{u}_{K_s}^{(s)}] \in \mathbb{R}^{M \times K_s}$, in which each column corresponds to a shared topic. While the term-topic matrix $\mathbf{U}_p$ can be represented as $\mathbf{U}_p = [\mathbf{u}_1^{(p)}, \cdots, \mathbf{u}_{K_p}^{(p)}] \in \mathbb{R}^{M \times K_p}$. The total number of topics in the question collection $\mathcal{D}$ is $K = K_s + PK_p$. Let $\mathbf{V}_p = [\mathbf{v}_1^{(p)}, \cdots, \mathbf{v}_{N_p}^{(p)}] \in \mathbb{R}^{(K_s + K_p) \times N_p}$ be the topic-question matrix corresponding to category c_p, in which each column denotes the question representation in the topic space. We also denote $\mathbf{V}_p^T = [\mathbf{H}_p^T, \mathbf{W}_p^T]$, where $\mathbf{H}_p \in \mathbb{R}^{K_s \times N_p}$ and $\mathbf{W}_p \in \mathbb{R}^{K_p \times N_p}$ correspond to the coefficients of shared topics $\mathbf{U}_s$ and category-specific topics $\mathbf{U}_p$, respectively.

Thus, given a question collection $\mathcal{D} = \{\mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_P\}$ together with the category labels $C = \{c_1, c_2, \cdots, c_P\}$, our proposed GNMFNC amounts to modeling the question collection $\mathcal{D}$ with P group

simultaneously, arriving at the following objective function:

$$\mathcal{O} = \sum_{p=1}^{P} \left\{ \lambda_p \left\| \mathbf{D}_p - [\mathbf{U}_s, \mathbf{U}_p]\mathbf{V}_p \right\|_F^2 + R(\mathbf{U}_s, \mathbf{U}_p) \right\} \tag{1}$$

where $\lambda_p \triangleq \|\mathbf{D}_p\|_F^{-2}$. $R(\mathbf{U}_s, \mathbf{U}_p)$ is a regularization term used to penalize the "similarity" between the shared topics and category-specific topics through $\mathbf{U}_s$ and $\mathbf{U}_p$.

In this paper, we aim to ensure that matrix $\mathbf{U}_s$ captures only shared topics and matrix $\mathbf{U}_p$ captures only the category-specific topics. For example, if matrices $\mathbf{U}_s$ and $\mathbf{U}_p$ are mutually orthogonal, we have $\mathbf{U}_s^T\mathbf{U}_p = \mathbf{0}$. To impose this constraint, we attempt to minimize the sum-of-squares of entries of the matrix $\mathbf{U}_s^T\mathbf{U}_p$ (e.g., $\|\mathbf{U}_s^T\mathbf{U}_p\|_F^2$ which uniformly optimizes each entry of $\mathbf{U}_s^T\mathbf{U}_p$). With this choice, the regularization term of $R(\mathbf{U}_s, \mathbf{U}_p)$ is given by

$$R(\mathbf{U}_s, \mathbf{U}_p) = \sum_{p=1}^{P} \alpha_p \left\| \mathbf{U}_s^T\mathbf{U}_p \right\|_F^2 + \sum_{l=1, l\neq p}^{P} \beta_l \left\| \mathbf{U}_p^T\mathbf{U}_l \right\|_F^2 \tag{2}$$

where α_p and β_l are the regularization parameters, $\forall p \in [1, P]$, $\forall l \in [1, P]$.

Learning the objective function in equation (1) involves the following optimization problem:

$$\min_{\mathbf{U}_s, \mathbf{U}_p, \mathbf{V}_p \geq 0} \mathcal{L} = \mathcal{O} + \sigma_1 \left\| \mathbf{U}_s^T\mathbf{1}_M - \mathbf{1}_{K_s} \right\|_F^2 + \sigma_2 \left\| \mathbf{U}_p^T\mathbf{1}_M - \mathbf{1}_{K_p} \right\|_F^2 + \sigma_3 \left\| \mathbf{V}_p\mathbf{1}_{N_p} - \mathbf{1}_{K_s+K_p} \right\|_F^2 \tag{3}$$

where σ_1, σ_2 and σ_3 are the shrinkage regularization parameters. Based on the shrinkage methodology, we can approximately satisfy the normalization constraints for each column of $[\mathbf{U}_s, \mathbf{U}_p]$ and $\mathbf{V}_p^T$ by guaranteeing the optimization converges to a stationary point.

2.2 Learning Algorithm

We present the solution to the GNMFNC optimization problem in equation (3) as the following theorem. The theoretical aspects of the optimization are presented in the next subsection.

Theorem 2.1. *Updating $\mathbf{U}_s$, $\mathbf{U}_p$ and $\mathbf{V}_p$ using equations (4)∼(6) corresponds to category c_p will monotonically decrease the objective function in equation (3) until convergence.*

$$\mathbf{U}_s \leftarrow \mathbf{U}_s \circ \frac{\left[\sum_{p=1}^{P} \lambda_p \mathbf{D}_p\mathbf{H}_p^T \right]}{\left[\sum_{p=1}^{P} \lambda_p [\mathbf{U}_s, \mathbf{U}_p]\mathbf{V}_p\mathbf{H}_p^T + \alpha_p \mathbf{U}_p\mathbf{U}_p^T\mathbf{U}_s \right]} \tag{4}$$

$$\mathbf{U}_p \leftarrow \mathbf{U}_p \circ \frac{\left[\lambda_p \mathbf{D}_p\mathbf{W}_p^T \right]}{\left[\lambda_p [\mathbf{U}_s, \mathbf{U}_p]\mathbf{V}_p\mathbf{W}_p^T + \alpha_p \mathbf{U}_s\mathbf{U}_s^T\mathbf{U}_p + \sum_{l=1, l\neq p}^{P} \beta_l \mathbf{U}_l\mathbf{U}_l^T\mathbf{U}_p \right]} \tag{5}$$

$$\mathbf{V}_p \leftarrow \mathbf{V}_p \circ \frac{\left[\lambda_p \mathbf{D}_p^T [\mathbf{U}_s, \mathbf{U}_p] \right]}{\left[\lambda_p \mathbf{V}_p^T [\mathbf{U}_s, \mathbf{U}_p]^T [\mathbf{U}_s, \mathbf{U}_p] \right]} \tag{6}$$

where operator $\circ$ is element-wise product and $\frac{[\cdot]}{[\cdot]}$ is element-wise division.

Based on Theorem 2.1, we note that multiplicative update rules given by equations (4)∼(6) are obtained by extending the updates of standard NMF (Lee and Seung, 2001). A number of techniques can be used here to optimize the objective function in equation (3), such as alternating least squares (Kim and Park, 2008), the active set method (Kim and Park, 2008), and the projected gradients approach (Lin, 2007). Nonetheless, the multiplicative updates derived in this paper have reasonably fast convergence behavior as shown empirically in the experiments.

2.3 Theoretical Analysis

In this subsection, we give the theoretical analysis of the optimization, convergence and computational complexity.

Without loss of generality, we only show the optimization of $\mathbf{U}_s$ and formulate the Lagrange function with constraints as follows:

$$\mathcal{L}(\mathbf{U}_s) = \mathcal{O} + \sigma_1 \left\| \mathbf{U}_s^T \mathbf{1}_M - \mathbf{1}_{K_s} \right\|_F^2 + \mathrm{Tr}(\Psi_s \mathbf{U}_s^T) \tag{7}$$

where $\mathrm{Tr}(\cdot)$ denotes the trace of a matrix, $\Psi_s \in \mathbb{R}^{K_s \times K_s}$ is the Lagrange multiplier for the nonnegative constraint $\mathbf{U}_s \geq \mathbf{0}$.

The partial derivative of $\mathcal{L}(\mathbf{U}_s)$ w.r.t. $\mathbf{U}_s$ is

$$\begin{aligned} \bigtriangledown_{\mathbf{U}_s} \mathcal{L}(\mathbf{U}_s) = & -2 \sum_{p=1}^{P} \lambda_p \mathbf{D}_p \mathbf{H}_p^T + 2 \sum_{p=1}^{P} \lambda_p [\mathbf{U}_s, \mathbf{U}_p] \mathbf{V}_p \mathbf{H}_p^T \\ & + 2 \sum_{p=1}^{P} \alpha_p \mathbf{U}_p \mathbf{U}_p^T \mathbf{U}_s + 2\sigma_1 \mathbf{U}_s - 2\sigma_1 + \Psi_s \end{aligned} \tag{8}$$

Using the Karush-Kuhn-Tucker (KKT) (Boyd and Vandenberghe, 2004) condition $\Psi_s \circ \mathbf{U}_s = \mathbf{0}$, we obtain

$$\bigtriangledown_{\mathbf{U}_s} \mathcal{L}(\mathbf{U}_s) \circ \mathbf{U}_s = \left\{ \begin{array}{l} -\sum_{p=1}^{P} \lambda_p \mathbf{D}_p \mathbf{H}_p^T + \sum_{p=1}^{P} \lambda_p [\mathbf{U}_s, \mathbf{U}_p] \mathbf{V}_p \mathbf{H}_p^T \\ +\sum_{p=1}^{P} \alpha_p \mathbf{U}_p \mathbf{U}_p^T \mathbf{U}_s + \sigma_1 \mathbf{U}_s - \sigma_1 \end{array} \right\} \circ \mathbf{U}_s = \mathbf{0} \tag{9}$$

After normalization of $\mathbf{U}_s$, the terms $\sigma_1 \mathbf{U}_s$ and σ_1 are in fact equal. They can be safely ignored from the above formula without influencing convergence. This leads to the updating rule for $\mathbf{U}_s$ in equation (4). Following the similar derivations as shown above, we can obtain the updating rules for the rest variables $\mathbf{U}_p$ and $\mathbf{V}_p$ in GNMFNC optimization, as shown in equations (5) and (6).

2.3.1 Convergence Analysis

In this subsection, we prove the convergence of multiplicative updates given by equations (4)$\sim$(6). We first introduce the definition of auxiliary function as follows.

Definition 2.1. *$\mathcal{F}(\mathbf{X}, \mathbf{X}')$ is an auxiliary function for $\mathcal{L}(\mathbf{X})$ if $\mathcal{L}(\mathbf{X}) \leq \mathcal{F}(\mathbf{X}, \mathbf{X}')$ and equality holds if and only if $\mathcal{L}(\mathbf{X}) = \mathcal{F}(\mathbf{X}, \mathbf{X})$.*

Lemma 2.1. *(Lee and Seung, 2001) If $\mathcal{F}$ is an auxiliary function for $\mathcal{L}$, $\mathcal{L}$ is non-increasing under the update*

$$\mathbf{X}^{(t+1)} = \arg\min_{\mathbf{X}} \mathcal{F}(\mathbf{X}, \mathbf{X}^{(t)})$$

Proof. By Definition 2.1, $\mathcal{L}(\mathbf{X}^{(t+1)}) \leq \mathcal{F}(\mathbf{X}^{(t+1)}, \mathbf{X}^{(t)}) \leq \mathcal{F}(\mathbf{X}^{(t)}, \mathbf{X}^{(t)}) = \mathcal{L}(\mathbf{X}^{(t)})$ □

Theorem 2.2. *Let $\mathcal{L}(\mathbf{U}_s^{(t+1)})$ denote the sum of all terms in $\mathcal{L}$ that contain $\mathbf{U}_s^{(t+1)}$, the following function is an auxiliary function for $\mathcal{L}(\mathbf{U}_s^{(t+1)})$*

$$\mathcal{F}(\mathbf{U}_s^{(t+1)}, \mathbf{U}_s^{(t)}) = \mathcal{L}(\mathbf{U}_s^{(t)}) + (\mathbf{U}_s^{(t+1)} - \mathbf{U}_s^{(t)}) \bigtriangledown_{\mathbf{U}_s^{(t)}} \mathcal{L}(\mathbf{U}_s^{(t)}) + \frac{1}{2} (\mathbf{U}_s^{(t+1)} - \mathbf{U}_s^{(t)})^2 \mathcal{P}(\mathbf{U}_s^{(t)}) \tag{10}$$

$$\mathcal{P}(\mathbf{U}_s^{(t)}) = \frac{\sum_{ij} \left[\sum_{p=1}^{P} \lambda_p [\mathbf{U}_s^{(t)}, \mathbf{U}_p] \mathbf{V}_p \mathbf{W}_p^T + \alpha_p \mathbf{U}_p \mathbf{U}_p^T \mathbf{U}_s^{(t)} + \sigma_1 \mathbf{U}_s^{(t)} \right]_{ij}}{\sum_{ij} [\mathbf{U}_s^{(t)}]_{ij}}$$

where $\bigtriangledown_{\mathbf{U}_s^{(t)}} \mathcal{L}(\mathbf{U}_s^{(t)})$ is the first-order derivative of $\mathcal{L}(\mathbf{U}_s^{(t)})$ with respect to $\mathbf{U}_s^{(t)}$. Theorem 2.2 can be proved similarly to (Lee and Seung, 2001) by validating $\mathcal{L}(\mathbf{U}_s^{(t+1)}) \leq \mathcal{F}(\mathbf{U}_s^{(t+1)}, \mathbf{U}_s^{(t)})$, $\mathcal{L}(\mathbf{U}_s^{(t+1)}) = \mathcal{F}(\mathbf{U}_s^{(t+1)}, \mathbf{U}_s^{(t+1)})$, and the Hessian matrix $\bigtriangledown \bigtriangledown_{\mathbf{U}_s^{(t+1)}} \mathcal{F}(\mathbf{U}_s^{(t+1)}, \mathbf{U}_s^{(t)}) \succeq 0$. Due to limited space, we omit the details of the validation.

	addition	multiplication	division	overall
GNMFNC: $\mathbf{U}_s$	$P(3MN_pK_s + MN_pK_p + MK_s^2)$	$P(3MN_pK_s + MN_pK_p + MK_s^2)$	MK_s	$O(PMN_pK_{max})$
GNMFNC: $\mathbf{U}_p$	$3MN_pK_p + MN_pK_s + PM^2K'$	$3MN_pK_p + MN_pK_s + PM^2K'$	MK_p	$O(PMRK')$
GNMFNC: $\mathbf{V}_p$	$3MN_pK'$	$3MN_pK'$	N_pK'	$O(MN_pK')$

Table 1: Computational operation counts for each iteration in GNMFNC.

Based on Theorem 2.2, we can fix $\mathbf{U}_s^{(t)}$ and minimize $\mathcal{F}(\mathbf{U}_s^{(t+1)}, \mathbf{U}_s^{(t)})$ with respect to $\mathbf{U}_s^{(t+1)}$. When setting $\bigtriangledown_{\mathbf{U}_s^{(t+1)}} \mathcal{F}(\mathbf{U}_s^{(t+1)}, \mathbf{U}_s^{(t)}) = \mathbf{0}$, we get the following updating rule

$$\mathbf{U}_s^{(t+1)} \leftarrow \mathbf{U}_s^{(t)} \circ \frac{\left[\sum_{p=1}^{P} \lambda_p \mathbf{D}_p \mathbf{H}_p^T + \sigma_1\right]}{\left[\sum_{p=1}^{P} \lambda_p [\mathbf{U}_s^{(t)}, \mathbf{U}_p] \mathbf{V}_p \mathbf{W}_p^T + \alpha_p \mathbf{U}_p \mathbf{U}_p^T \mathbf{U}_s^{(t)} + \sigma_1 \mathbf{U}_s^{(t)}\right]} \tag{11}$$

which is consistent with the updating rule derived from the KKT conditions aforementioned.

By Lemma 2.1 and Theorem 2.2, we have $\mathcal{L}(\mathbf{U}_s^{(0)}) = \mathcal{F}(\mathbf{U}_s^{(0)}, \mathbf{U}_s^{(0)}) \geq \mathcal{F}(\mathbf{U}_s^{(1)}, \mathbf{U}_s^{(0)}) \geq \mathcal{F}(\mathbf{U}_s^{(1)}, \mathbf{U}_s^{(1)}) = \mathcal{L}(\mathbf{U}_s^{(1)}) \geq \cdots \geq \mathcal{L}(\mathbf{U}_s^{(Iter)})$, where $Iter$ is the number of iterations. Therefore, $\mathbf{U}_s$ is monotonically decreasing. Since the objective function $\mathcal{L}$ is lower bounded by 0, the correctness and convergence of Theorem 2.1 is validated.

2.3.2 Computational Complexity

In this subsection, we discuss the time computational complexity of the proposed algorithm GNMFNC. Besides expressing the complexity of the algorithm using big O notation, we also count the number of arithmetic operations to provide more details about running time. We show the results in Table 1, where $K_{max} = \max\{K_s, K_p\}$, $K' = K_s + K_p$ and $R = \max\{M, N_p\}$.

Suppose the multiplicative updates stop after $Iter$ iterations, the time cost of multiplicative updates then becomes $O(Iter \times PMRK')$. We set $Iter = 100$ empirically in rest of the paper. Therefore, the overall running time of GNMFNC is linear with respect to the size of word vocabulary, the number of questions and categories.

2.4 Relevance Ranking

The motivation of incorporating matrix factorization into relevance ranking is to learn the word relationships and reduce the "lexical gap" (Zhou et al., 2013a). To do so, given a queried question q with category label c_p from Yahoo! Answers, we first represent it in the latent topic space as $\mathbf{v}_q$,

$$\mathbf{v}_q = \arg\min_{\mathbf{v} \geq 0} \|\mathbf{q} - [\mathbf{U}_s, \mathbf{U}_p]\mathbf{v}\|_2^2 \tag{12}$$

where vector $\mathbf{q}$ is the tf-idf representation of queried question q in the term space.

For each historical question d (indexed by r) in question collection $\mathcal{D}$, with representation $\mathbf{v}_d = r$-th column of $\mathbf{V}$, we compute its similarity with queried question $\mathbf{v}_q$ as following

$$s_{topic}(q, d) = \frac{< \mathbf{v}_q, \mathbf{v}_d >}{\|\mathbf{v}_q\|_2 \cdot \|\mathbf{v}_d\|_2} \tag{13}$$

The latent topic space score $s_{topic}(q, d)$ is combined with the conventional term matching score $s_{term}(q, d)$ for final relevance ranking. There are several ways to conduct the combination. Linear combination is a simple and effective way. The final relevance ranking score $s(q, d)$ is:

$$s(q, d) = \gamma s_{topic}(q, d) + (1 - \gamma) s_{term}(q, d) \tag{14}$$

where $\gamma \in [0, 1]$ is the parameter which controls the relative importance of the latent topic space score and term matching score. $s_{term}(q, d)$ can be calculated with any of the conventional relevance models such as BM25 (Robertson et al., 1994) and LM (Zhai and Lafferty, 2001).

3 Experiments

3.1 Data Set and Evaluation Metrics

We collect the data set from Yahoo! Answers and use the *getByCategory* function provided in Yahoo! Answers API[3] to obtain CQA threads from the Yahoo! site. More specifically, we utilize the *resolved* questions and the resulting question repository that we use for question retrieval contains 2,288,607 questions. Each resolved question consists of four parts: "question title", "question description", "question answers" and "question category". We only use the "question title" and "question category" parts, which have been widely used in the literature for question retrieval (Cao et al., 2009; Cao et al., 2010). There are 26 first-level categories in the predefined natural hierarchy, i.e., each historical question is categorized into one of the 26 categories. The categories include "Arts & Humanities", "Beauty & Style", "Business & Finance", etc.

In order to evaluate our approach, we randomly select 2,000 questions as queried questions from the above data collection to construct the validation/test sets, and the remaining data collection as training set. Note that we select the queried questions in proportion to the number of questions and categories against the whole distribution to have a better control over a possible imbalance. To obtain the ground-truth, we employ the Vector Space Model (VSM) (Salton et al., 1975) to retrieve the top 10 results and obtain manual judgements. The top 10 results don't include the queried question itself. Given a returned result by VSM, an annotator is asked to label it with "relevant" or "irrelevant". If a returned result is considered semantically equivalent to the queried question, the annotator will label it as "relevant"; otherwise, the annotator will label it as "irrelevant". Two annotators are involved in the annotation process. If a conflict happens, a third person will make judgement for the final result. In the process of manually judging questions, the annotators are presented only the questions. As a result, there are in total 20,000 judged question pairs. We randomly split the 2,000 queried questions into validation/test sets, each has 1,000/1,000 queried questions. We use the validation set for parameter tuning and the test set for evaluation.

Evaluation Metrics: We evaluate the performance of question retrieval using the following metrics: Mean Average Precision (MAP) and Precision@N (P@N). MAP rewards methods that return relevant questions early and also rewards correct ranking of the results. P@N reports the fraction of the top-N questions retrieved that are relevant. We perform a significant test, i.e., a t-test with a default significant level of 0.05.

There are several parameters used in the paper, we tune these parameters on the validation set. Specifically, we set the number of category-specific topics per category and the number of shared topics in GNMFNC as $(K_s, K_p) = \{(5,2),(10,4),(20,8),(40,16),(80,32)\}$, resulting in $K = \{57, 114, 228, 456, 912\}$ total number of topics. (Note that the total number of topics in GNMFNC is $K_s + 26 \times K_p$, where 26 is the number of categories in the first-level predefined natural hierarchy[4]). Finally, we set $(K_s, K_p) = (20, 8)$ and $K = 228$ empirically as this setting yields the best performance.

For regularization parameters α_p and β_l, it is difficult to directly tune on the validation set, we present an alternative way by adding a common factor a to look at the objective function of optimization problem in equation (3) on the training data. In other words, we set $\alpha_p = \frac{a}{K_s \times K_p}$ and $\beta_l = \frac{a}{K_p \times K_l}$. Therefore, we tune the parameters α_p and β_l by alternatively adjusting the common factor a via grid search. As a result, we set $a = 100$, resulting in $\alpha_p = \beta_l = 0.625$ in the following experiments. The trade-off parameter γ in the linear combination is set from 0 to 1 in steps of 0.1 for all methods. We set $\gamma = 0.6$ empirically. For shrinkage regularization parameters, we empirically set $\sigma_1 = \sigma_2 = \sigma_3 = 1$.

3.2 Question Retrieval Results

In this experiment, we present the experimental results for question retrieval on the test data set. Specifically, for our proposed GNMFNC, we combine the latent topic matching scores with the term matching scores given by BM25 and LM, denoted as "BM25+GNMFNC" and "LM+GNMFNC". Table 2 shows

[3] http://developer.yahoo.com/answers

[4] Here we do not use the leaf categories because we find that it is not possible to run GNMFNC with such large number of topics on the current machines, and we will leave it for future work.

Table 2: Comparison with different methods for question retrieval.

#	Methods	MAP	P@10
1	BM25	0.243	0.225
2	LM	0.286	0.232
3	(Jeon et al., 2005)	0.327	0.235
4	(Xue et al., 2008)	0.341	0.238
5	(Zhou et al., 2011)	0.365	0.243
6	(Singh, 2012)	0.354	0.240
7	(Cao et al., 2010)	0.358	0.242
8	(Cai et al., 2011)	0.331	0.236
9	**BM25+GNMFNC**	**0.369**	**0.248**
10	**LM+GNMFNC**	**0.374**	**0.251**

Table 3: Comparison of matrix factorizations for question retrieval.

#	Methods	MAP	P@10
1	BM25	0.243	0.225
2	BM25+NMF	0.325	0.235
3	BM25+CNMF	0.344	0.239
4	BM25+GNMF	0.361	0.242
5	BM25+GNMFNC	0.369	0.248
6	LM	0.286	0.232
7	LM+NMF	0.337	0.237
8	LM+CNMF	0.352	0.240
9	LM+GNMF	0.365	0.243
10	LM+GNMFNC	0.374	0.251

the main retrieval performances under the evaluation metrics MAP, P@1 and P@10. Row 1 and row 2 are the baseline systems, which model the relevance ranking using BM25 (Robertson et al., 1994) and language model (LM) (Zhai and Lafferty, 2001) in the term space. Row 3 is word-based translation model (Jeon et al., 2005), and row 4 is word-based translation language model (TRLM) (Xue et al., 2008). Row 5 is phrase-based translation model (Zhou et al., 2011), and row 6 is the entity-based translation model (Singh, 2012). Row 7 to row 11 explore the natural categories for question retrieval. In row 7, Cao et al. (2010) employed the natural categories to compute the local and global relevance with different model combination, here we use the combination VSM + TRLM for comparison because this combination obtains the superior performance than others. In row 8, Cai et al. (2011) proposed a category-enhanced TRLM for question retrieval. There are some clear trends in the results of Table 2:

(1) BM25+GNMFNC and LM+GNMFNC perform *significantly* better than BM25 and LM respectively (t-test, p-value < 0.05, row 1 vs. row 9; row 2 vs. row 10), indicating the effective of GNMFNC.

(2) BM25+GNMFNC and LM+GNMFNC perform better than translation methods, some improvements are statistical significant (t-test, p-value < 0.05, row 3 and row 4 vs. row 9 and row 10). The reason may be that GNMFNC models the relevance ranking in the latent topic space, which can also effectively solve the the lexical gap problem.

(3) Capturing the shared aspects and the category-specific individual aspects with natural categories in the group modeling framework can *significantly* improve the performance of question retrieval (t-test, p-value < 0.05, row 7 and row 8 vs. row 9 and row 10).

(4) Natural categories are useful and effectiveness for question retrieval, no matter in the group modeling framework or existing retrieval models (row 3$\sim$ row 6 vs. row 7$\sim$row 10).

3.3 Comparison of Matrix Factorizations

We note that our proposed GNMFNC is related to non-negative matrix factorization (NMF) (Lee and Seung, 2001) and its variants, we introduce three baselines. The first baseline is NMF, which is trained on the whole training data. The second baseline is CNMF, which is trained on each category without considering the shared topics. The third baseline is GNMF (Lee and Choi, 2009; Wang et al., 2012), which is similar to our GNMFNC but there are no constraints on the category-specific topics to prevent them from capturing the information from the shared topics.

NMF and GNMF are trained on the training data with the same parameter settings in section 4.1 for fair comparison. For CNMF, we also train the model on the training data with the same parameter settings in section 4.1, except parameter K_s, as there exists no shared topics in CNMF.

Table 3 shows the question retrieval performance of NMF families on the test set, obtained with the best parameter settings determined by the validation set. From the results, we draw the following observations:

(1) All of these methods can *significantly* improve the performance in comparison to the baseline BM25 and LM (t-test, p-value < 0.05).

(2) GNMF and GNMFNC perform *significantly* better than NMF and CNMF respectively (t-test, p-value < 0.05), indicating the effectiveness of group matrix factorization framework, especially the use of shared topics.

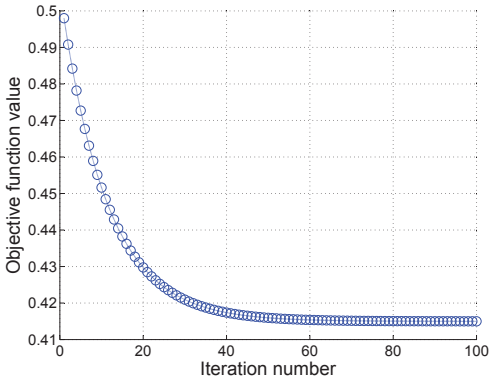


Figure 1: Convergence curve of GNMFNC.

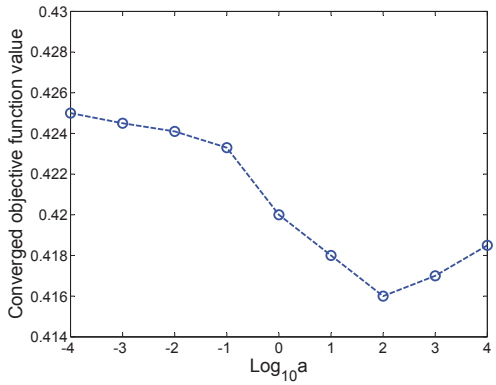


Figure 2: Objective function value vs. factor a.

(3) GNMFNC performs *significantly* better than GNMF (t-test, p-value < 0.05, row 4 vs. row 5; row 9 vs. row 10), indicating the effectiveness of the regularization term on the category-specific topics to prevent them from capturing the information from the shared topics.

From the experimental results reported above, we can conclude that our proposed GNMFNC is useful for question retrieval with high accuracies. To the best of our knowledge, it is the first time to investigate the group matrix factorization for question retrieval.

3.4 Convergence Behavior

In subsection 2.3.1, we have shown that the multiplicative updates given by equations (4)$\sim$(6) are convergent. Here, we empirically show the convergence behavior of GNMFNC.

Figure 1 shows the convergence curve of GNMFNC on the training data set. From the figure, y-axis is the value of objective function and x-axis denotes the iteration number. We can see that the multiplicative updates for GNMFNC converge very fast, usually within 80 iterations.

3.5 Regularization Parameters Selection

One success of this paper is to use regularized constrains on the category-specific topics to prevent them from capturing the information from the shared topics. It is necessary to give an in-depth analysis of the regularization parameters used in the paper. Consider the regularization term used in equation (2), each element in $\mathbf{U}_s^T\mathbf{U}_p$ and $\mathbf{U}_p^T\mathbf{U}_l$ has a value between 0 and 1 as each column of $\mathbf{U}_s$, $\mathbf{U}_p$ and $\mathbf{U}_l$ is normalized. Therefore, it is appropriate to normalize the term having $\|\mathbf{U}_s^T\mathbf{U}_p\|_F^2$ by K_sK_p since there are $K_s \times K_p$ elements in $\mathbf{U}_s^T\mathbf{U}_p$. Similarly, $\|\mathbf{U}_p^T\mathbf{U}_l\|_F^2$ is normalized by K_lK_p. Note that $K_l = K_p$ and $l \neq p$. As discussed in subsection 4.1, we present an alternative way by adding a common factor a and set $\alpha_p = \frac{a}{K_s \times K_p}$ and $\beta_l = \frac{a}{K_p \times K_l}$. The common factor a is used to adjust a trade-off between the matrix factorization errors and the mutual orthogonality, which cannot directly tune on the validation set. Thus, we look at the objective function of optimization problem in equation (3) on the training data and find the optimum value for a.

Figure 2 shows the objective function value vs. common factor a, where y-axis denotes the converged objective function value, and x-axis denotes $\text{Log}_{10}a$. We can see that the optimum value of a is 100. Therefore, the common factor a can be fixed at 100 for our data set used in the paper, resulting in $\alpha_p = \beta_l = 0.625$. Note that the optimum value of (K_s, K_p) are set as $(20, 8)$ in subsection 4.1. Due to limited space, we do not give an in-depth analysis for other parameters.

4 Conclusion and Future Work

In this paper, we propose a novel approach, called group non-negative matrix factorization with natural categories (GNMFNC). The proposed method is achieved by learning the category-specific topics for each category as well as shared topics across all categories via a group non-negative matrix factorization framework. We derive an efficient algorithm for learning the factorization, analyze its complexity, and

provide proof of convergence. Experiments show that our proposed approach significantly outperforms various baseline methods and achieves state-of-the-art performance for question retrieval.

There are some ways in which this research could be continued. First, the optimization of GNMFNC can be decomposed into many sub-optimization problems, a natural avenue for future research is to reduce the running time by executing the optimization in a distributed computing environment (e.g., MapReduce (Dean et al., 2004)). Second, another combination approach will be used to incorporate the latent topic match score as a feature in a learning to rank model, e.g., LambdaRank (Burges et al., 2007). Third, we will try to investigate the use of the proposed approach for other kinds of data sets with larger categories, such as categorized documents from ODP project.[5]

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61333018 and No. 61303180), the Beijing Natural Science Foundation (No. 4144087), CCF Opening Project of Chinese Information Processing, and also Sponsored by CCF-Tencent Open Research Fund. We thank the anonymous reviewers for their insightful comments.

References

D. Bernhard and I. Gurevych. 2009. Combining lexical semantic resources with question & answer archives for translation-based answer finding. In *Proceedings of ACL*, pages 728-736.

S. Boyd and L. Vandenberghe. 2004. *Convex Optimization*. Cambridge university press.

C. Boutsidis and E. Gallopoulos. 2008. SVD based initialization: a head start for nonnegative matrix factorization. *Pattern Recognition*, 41(4):1350-1362.

C. Burges, R. Ragno, and Q. Le. 2007. Learning to rank with nonsmooth cost function. In *Proceedings of NIPS*.

L. Cai, G. Zhou, K. Liu, and J. Zhao. 2011. Learning the latent topics for question retrieval in community QA. In *Proceedings of IJCNLP*.

X. Cao, G. Cong, B. Cui, C. Jensen, and C. Zhang. 2009. The use of categorization information in language models for question retrieval. In *Proceedings of CIKM*, pages 265-274.

X. Cao, G. Cong, B. Cui, and C. Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of WWW*.

J. Dean, S. Ghemanwat, and G. Inc. 2004. Mapreduce: simplified data processing on large clusters. In *Proceedings of OSDI*.

H. Duan, Y. Cao, C. Lin, and Y. Yu. 2008. Searching questions by identifying questions topics and question focus. In *Proceedings of ACL*, pages 156-164.

J. Jeon, W. Croft, and J. Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of CIKM*, pages 84-90.

Z. Ji, F. Xu, and B. Wang. 2012. A category-integrated language model for question retrieval in community question answering. In *Proceedings of AIRS*, pages 14-25.

H. Kim and H. Park. 2008. Non-negative matrix factorization based on alternating non-negativity constrained least squares and active set method. *SIAM J Matrix Anal Appl*, 30(2):713-730.

A. Langville, C. Meyer, R. Albright, J. Cox, and D. Duling. 2006. Initializations for the nonnegative matrix factorization. In *Proceedings of KDD*.

J. Lee, S. Kim, Y. Song, and H. Rim. 2008. Bridging lexical gaps between queries and questions on large online Q&A collections with compact translation models. In *Proceedings of EMNLP*, pages 410-418.

D. Lee and H. Seung. 2001. Algorithms for non-negative matrix factorization. In *Proceedings of NIPS*.

[5]http://www.dmoz.org/

H. Lee and S. Choi. 2009. Group nonnegative matrix factorization for eeg classification. In *Proceedings of AISTATS*, pages 320-327.

C. Lin. 2007. Projected gradient methods for nonnegative matrix factorization. *Neural Comput*, 19(10):2756-2779.

Z. Ming, T. Chua, and G. Cong. 2010. Exploring domain-specific term weight in archived question search. In *Proceedings of CIKM*, pages 1605-1608.

S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of ACL*, pages 464-471.

S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. 1994. Okapi at trec-3. In *Proceedings of TREC*, pages 109-126.

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

A. Singh. 2012. Entity based q&a retrieval. In *Proceedings of EMNLP-CoNLL*, pages 1266-1277.

Q. Wang, Z. Cao, J. Xun, and H. Li. 2012. Group matrix factorizaiton for scalable topic modeling. In *Proceedings of SIGIR*.

X. Xue, J. Jeon, and W. Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of SIGIR*, pages 475-482.

C. Zhai and J. Lafferty. 2001. A study of smooth methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR*, pages 334-342.

G. Zhou, L. Cai, J. Zhao, and K. Liu. 2011. Phrase-based translation model for question retrieval in community question answer archives. In *Proceedings of ACL*, pages 653-662.

G. Zhou, F. Liu, Y. Liu, S. He, and J. Zhao. 2013. Statistical machine translation improves question retrieval in community question answering via matrix factorization. In *Proceedings of ACL*, pages 852-861.

G. Zhou, Y. Chen, D. Zeng, and J. Zhao. 2013. Toward faster and better retrieval models for question search. In *Proceedings of CIKM*, pages 2139-2148.

Multi-Objective Search Results Clustering

Sudipta Acharya **Sriparna Saha** **Jose G. Moreno** **Gaël Dias**
Indian Institute of Technology Patna Normandie University - CNRS GREYC
Kurji, Patna, Bihar, India Caen, France
{sudipta.pcs13,sriparna}@iitp.ac.in first.last@unicaen.fr

Abstract

Most web search results clustering (SRC) strategies have predominantly studied the definition of adapted representation spaces to the detriment of new clustering techniques to improve performance. In this paper, we define SRC as a multi-objective optimization (MOO) problem to take advantage of most recent works in clustering. In particular, we define two objective functions (compactness and separability), which are simultaneously optimized using a MOO-based simulated annealing technique called AMOSA. The proposed algorithm is able to automatically detect the number of clusters for any query and outperforms all state-of-the-art text-based solutions in terms of F_β-measure and F_{b^3}-measure over two gold standard data sets.

1 Introduction

Web search results clustering (SRC), also known as post-retrieval clustering or ephemeral clustering has received much attention for the past twenty years for easing up user's effort in web browsing. The key idea behind SRC systems is to return some meaningful labeled clusters from a set of web documents (or web snippets) retrieved from a search engine for a given query.

Recently, SRC strategies have been focusing on the introduction of external (exogenous) knowledge to better capture semantics between documents (Scaiella et al., 2012; Marco and Navigli, 2013). Although this research direction has evidenced competitive results, the proposed clustering techniques are based on a single cluster quality measure, which must reflect alone the goodness of a given partitioning. These techniques are usually referred to as single objective optimizations (SOO).

In this paper, we hypothesize that improved clustering can be achieved by defining different objective functions over well-known data representations. As such, our study aims to focus on new clustering issues for SRC instead of defining new representation spaces.

Recent studies (Maulik et al., 2011) have shown that clustering can be defined as a multi-objective optimization (MOO) problem. Within the context of SRC, we propose to define two objective functions (compactness and separability), which are simultaneously optimized using a MOO-based simulated annealing technique called AMOSA (Bandyopadhyay et al., 2008).

In order to draw conclusive remarks, we present an exhaustive evaluation where our MOO algorithm (*MOO-clus*) is compared to the most competitive text-based (endogenous) SRC algorithms: STC (Zamir and Etzioni, 1998), LINGO (Osinski and Weiss, 2005), OPTIMSRC (Carpineto and Romano, 2010) and GK-means (Moreno et al., 2013). Experiments are run over two different gold standard data sets (ODP-239 and MORESQUE) for two clustering evaluation metrics (F_β-measure and F_{b^3}-measure). Results show that *MOO-clus* outperforms all text-based solutions and approaches performances of knowledge driven strategies (Scaiella et al., 2012). In this paper, our main contributions are:

- The first[1] attempt to solve SRC by defining multiple objective functions,
- A new MOO clustering algorithm for SRC, which automatically determines the number of clusters,

[1] As far as we know.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 99–108, Dublin, Ireland, August 23-29 2014.

- An exhaustive evaluation of SRC algorithms with recent data sets and evaluation metrics over the most competitive state-of-the-art text-based SRC algorithms.

2 Related Work

2.1 SRC Algorithms

One of the most cited SRC solutions is the Suffix Tree Clustering (STC) algorithm proposed by (Zamir and Etzioni, 1998). They propose a monothetic clustering technique, which merges base clusters with high string overlap based on web snippets represented as compact tries. Their evaluation shows improvements over agglomerative hierarchical clustering, K-Means, Buckshot, Fractionation and Single-Pass algorithms, and is still a hard baseline to beat (Moreno and Dias, 2014).

Later, (Osinski and Weiss, 2005) proposed a polythetic solution called LINGO based on the same string representation as of (Zamir and Etzioni, 1998). They first extract frequent phrases based on suffix-arrays and match group descriptions with topics obtained with latent semantic analysis. Documents are then assigned straightforwardly to their corresponding groups. Their evaluation does not allow conclusive remarks but they propose an open source implementation, which is an important contribution.

More recently, (Carpineto and Romano, 2010) showed that the characteristics of the outputs returned by SRC algorithms suggest the adoption of a meta clustering approach. The underlying idea is that different SOO solutions lead to complementary results that must be combined. So, they introduce a novel criterion to measure the concordance of two partitions of objects into different clusters based on the information content associated to the series of decisions made by the partitions on single pairs of objects. The results of OPTIMSRC demonstrate that meta clustering is superior over individual clustering techniques.

The latest work, exclusively based on endogenous information (i.e. web snippets returned by the search engine), is proposed by (Moreno et al., 2013). They adapt the K-means algorithm to a third-order similarity measure and propose a stopping criterion to automatically determine the "optimal" number of clusters. Experiments are run over two gold standard data sets, ODP-239 (Carpineto and Romano, 2010) and MORESQUE (Navigli and Crisafulli, 2010), and show improved results over all state-of-the-art text-based SRC techniques so far.

A great deal of works have also proposed to include exogenous information to solve the SRC problem. One important work is proposed by (Scaiella et al., 2012) who use Wikipedia articles to build a bipartite graph and apply spectral clustering over it to discover relevant clusters. More recently, (Marco and Navigli, 2013) proposed to include word sense induction based on the Web1T corpus (Brants and Franz, 2006) to improve SRC. In this paper, we exclusively focus on endogenous solutions.

2.2 MOO-based Clustering

Many works have been proposed where the problem of clustering is posed as one of multi-objective optimization (Deb, 2009; Maulik et al., 2011). One important work is proposed by (Handl and Knowles, 2007) who define a multi-objective clustering technique with automatic K-determination called MOCK. Their algorithm outperforms several standard single-objective clustering algorithms (K-means, agglomerative hierarchical clustering and ensemble clustering) on artificial data sets.

In parallel, a multi-objective evolutionary algorithm for fuzzy clustering is proposed by (Bandyopadhyay et al., 2007) for clustering gene expressions. Here, two objectives are simultaneously optimized. The first one is the objective function optimized in the fuzzy C-means algorithm (Bezdek, 1981) and the other one is the Xie-Beni index (Xie and Beni, 1991).

Later, (Mukhopadhyay and Maulik, 2009) proposed a novel approach that combines the multi-objective fuzzy clustering method of (Bandyopadhyay et al., 2007) with a Support Vector Machines (SVM) classifier. Performance results are provided for remote sensing data.

As far as we know, within text applications, (Morik et al., 2012) is the first work, which formulates text clustering a multi-objective optimization problem. In particular, they express desired properties of frequent termset clustering in terms of multiple conflicting objective functions. The optimization is solved by a genetic algorithm and the result is a set of Pareto-optimal solutions. Note that this effort is

defined for large text colllections with high dimensional data, which is contradictory to the specific task of SRC (Carpineto et al., 2009)[2].

2.3 Our Motivation

Recent works have focused on the introduction of external (exogenous) knowledge to solve the SRC task. However, this research direction higly depends on existing resources, which are not available for a great deal of languages. Moreover, (Carpineto and Romano, 2010) has suggested an interesting research direction, which has still remained unexplored. Indeed, (Carpineto and Romano, 2010) showed that meta clustering leads to improved results in the context of text-based (endogenous) SRC. This suggests that better clustering can be obtained by combining different SOO solutions. However, their algorithm is casted to a SOO problem of the concordance between the clustering combination and a meta partition.

As a consequence, we hypothesize that improved performances can be obtained by defining the SRC task as a MOO clustering problem. For that purpose, we (1) take advantage of the recent advances in the field of multi-objective clustering (Saha and Bandyopadhyay, 2010), (2) define new objective functions in a non euclidean space and (3) adapt a MOO-based simulated annealing technique called AMOSA (Bandyopadhyay et al., 2008) to take into account third-order similarity metrics (Moreno et al., 2013).

3 Clustering as a MOO Problem

3.1 Formal Definition of MOO Clustering

Multi-objective optimization can be formally stated as finding the vector $\overline{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ of decision variables that simultaneously optimize *M* objective function values $\{f_1(\overline{x}), f_2(\overline{x}), \ldots, f_M(\overline{x})\}$ while satisfying user-defined constraints, if any.

An important concept in MOO is that of domination. Within the context of a maximization problem, a solution $\overline{x_i}$ is said to dominate $\overline{x_j}$ if $\forall k \in 1, 2, \ldots, M,\ f_k(\overline{x_i}) \geq f_k(\overline{x_j})$ and $\exists k \in 1, 2, \ldots, M,$ such that $f_k(\overline{x_i}) > f_k(\overline{x_j})$.

Among a set of solutions R, the non-dominated set of solutions $R^{'}$ are those that are not dominated by any member of the set R and is called the globally Pareto-optimal set or Pareto front. In general, a MOO algorithm outputs a set of solutions not dominated by any solution encountered by it. These notions can be illustrated by considering an optimization problem with two objective functions (f_1 and f_2) with six different solutions, as shown in Figure 1. Here target is to maximize both objective functions f_1 and f_2.

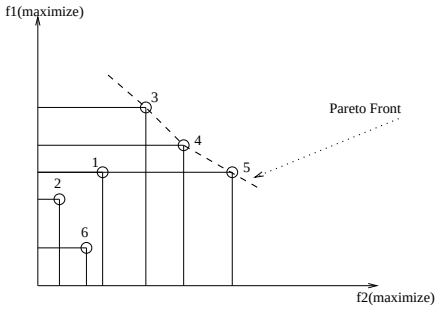


Figure 1: Example of dominance and Pareto optimal front.

In this example, solutions 3, 4 and 5 dominate all the other three solutions 1, 2 and 6. Solutions 3, 4 and 5 are nondominating to each other. Because 3 is better than 4 w.r.t. function f_1, but 4 is better than 3 w.r.t. f_2. Similarly 4 is better than 5 w.r.t. f_1 but 5 is better than 4 w.r.t. f_2. The same happens for solutions 3 and 5. So, the Pareto front is made of solutions 3, 4 and 5.

Within the specific context of clustering, two objective functions are usually defined, which must be optimized simultaneously. These functions are based on two intrinsic properties of the data space and are defined as follows.

[2]SRC is usually referred to as text clustering in the "small": i.e. small list of short text documents.

Compactness: This objective function measures the proximity among the various elements of a given cluster and must be maximized.

Separability: This objective function measures the similarity between two cluster centroids and must be minimized.

3.2 AMOSA Optimization Strategy

Clustering is viewed as a search problem, where optimal partitions satisfying the given set of objective functions must be discovered. As such, an optimization strategy must be defined. Here, we propose to use archived multi-objective simulated annealing (AMOSA) proposed by (Bandyopadhyay et al., 2008). AMOSA incorporates the concept of an archive where the non-dominated solutions seen so far are stored.

Two limits are kept on the size of the archive: a hard limit denoted by *HL* and a soft limit denoted by *SL*. Given $\gamma > 1$, the algorithm begins with the initialization of a number $(\gamma \times SL)$ of solutions each of which representing a state in the search space. Thereafter, the non-dominated solutions are determined and stored in the archive.

Then, one point is randomly selected from the archive. This is taken as the current point, or the initial solution, at temperature $T = T_{max}$. The current point is perturbed/mutated to generate a new solution named new-pt and its objective functions are computed. The domination status of the new-pt is checked w.r.t. the current point and the solutions in the archive. Based on domination status, different cases may arise: (i) accept the new-pt, (ii) accept the current-pt or (iii) accept a solution from the archive. In case of overflow of the archive, its size is reduced to *HL*.

The process is repeated *iter* times for each temperature that is annealed with a cooling rate of α ($<$1) till the minimum temperature T_{min} is attained. The process thereafter stops and the archive contains the final non-dominated solutions i.e. the Pareto front.

4 SRC as MOO Problem: MOO-clus

4.1 Archive Initialization

As we follow an endogenous approach, only the information returned by a search engine is used. In particular, we only deal with web snippets and each one is represented as a word feature vector. So, our solution called *MOO-clus* starts its execution after initializing the archive with some random solutions as archive members. Here, a particular solution refers to a complete assignment of web snippets (or data points) in several clusters. So, the first step is to represent a solution compatible with AMOSA, which represents each individual solution as a string. In order to encode the clustering problem in the form of a string, a center-based representation is used. Note that the use of a string representation facilitates the definition of individuals and mutation functions (Bandyopadhyay et al., 2008).

Let us assume that the archive member i represents the centroids of K_i clusters and the number of tokens in a centroid is p[3], then the archive member (or string) has length l_i where $l_i = p \times K_i$. To initialize the number of centroids K_i encoded in the string i, a random value between 2 and K_{max} is chosen and each K_i cluster centroid is initialized by randomly generated tokens from the global vocabulary.

4.2 Assignment of Web Snippets

As for any classical clustering algorithms, web snippets (or data points) must be assigned to their respective clusters. In *MOO-clus*, this assignment is computed as in (Moreno et al., 2013), to take advantage of recent advances in similarity measures. For two word feature vectors d_i and d_j, their similarity is evaluated by the similarity of their constituents as defined in Equation 1.

$$S(d_i, d_j) = \frac{1}{\|d_i\|\|d_j\|} \sum_{r=1}^{\|d_i\|} \sum_{b=1}^{\|d_j\|} SCP(w_i^r, w_j^b), \quad \text{with} \quad SCP(w_1, w_2) = \frac{P(w_1, w_2)^2}{P(w_1) \times P(w_2)} \tag{1}$$

[3] A centroid is represented by a p word feature vector $(w_k^1, w_k^2, w_k^3, \ldots, w_k^p)$.

Here, w_i^r (resp. w_j^b) corresponds to the token at the r^{th} (resp. b^{th}) position of the word feature vector d_i (resp. d_j). $\|d_i\|$ and $\|d_j\|$ respectively denote the total number of tokens in word feature vectors d_i and d_j. $SCP(w_i^r, w_j^b)$ is the Symmetric Conditional Probability (da Silva et al., 1999) where $P(.,.)$ is the joint probability of two tokens (w_1 and w_2) appearing in the same word feature vector and $P(.)$ is the marginal probability of any token appearing in a word feature vector.

Note that each cluster centroid is a word feature vector of varying number of tokens. Thus, Equation 2 is used to assign any data point (web snippet) d_j to a cluster t whose centroid has the maximum similarity value to d_j.

$$t = argmax_{k=1,...K} S(d_j, m_{\pi_k}) \tag{2}$$

K denotes the total number of clusters, d_j is the j^{th} web snippet, m_{π_k} is the centroid of the k^{th} cluster π_k and $S(d_j, m_{\pi_k})$ denotes similarity measurement between the point d_j and cluster centroid m_{π_k} defined in Equation 1.

4.3 Definition of Objective Functions

A string i represents a set of centroids to which web snippets can be assigned as seen in Section 4.2. As a consequence, each string i corresponds to a candidate partition of the data space. Now, in order to verify the domination of different solutions over other ones, objective functions must be defined. Compactness and separability are usually used in MOO clustering solutions. Here, compactness can be defined as the informational density of each cluster. This can be straightforwardly formulated as in Equation 3.

$$Compactness = \sum_{k=1}^{K} \sum_{d_i \in \pi_k} S(d_i, m_{\pi_k}) \tag{3}$$

Note that if tokens in a particular cluster are very similar to the cluster centroid then the corresponding $Compactness$ value would be maximized. Here our target is to form good clusters whose compactness in terms of similarity should be maximum.

The second objective function is cluster separability, which measures the dissimilarity between two cluster centroids. Indeed, the purpose of any clustering algorithm is to obtain compact similar typed clusters, which are dissimilar to each other. Here, we define separability as the minimization of the summation of similarities between each pair of cluster centroids. This is defined in Equation 4, where m_{π_k} and m_{π_o} are the centroids of clusters π_k and π_o, respectively.

$$Separability = \sum_{k=1}^{K} \sum_{o=k+1}^{K} S(m_{\pi_k}, m_{\pi_o}) \tag{4}$$

Finally, for a particular string, the following objectives $\{Compactness, \frac{1}{Separability}\}$ are maximized using the search capability of AMOSA.

4.4 Search Operators

In *MOO-clus*, AMOSA is used as the optimization strategy. For that purpose, three different types of mutation operations have been defined to suit the framework.

Mutation 1: This mutation operation is used to update the cluster center representation. Each token of cluster centroid is replaced by one token from the global vocabulary according to highest SCP similarity. This is applied individually to all tokens of a particular centroid if it is selected for mutation.

Mutation 2: This mutation operation is used to reduce the size of the string by 1. We randomly select a cluster centroid and thereafter all the tokens of this centroid are deleted from the string.

Mutation 3: This mutation is for increasing the size of string by 1 i.e. one new centroid is inserted in the string. For that purpose, we randomly choose p number of tokens from the global vocabulary and add it to the string.

Let be a string $< w_1\ w_2\ w_3\ w_4\ w_5\ w_6 >$ representing three cluster centroids (w_1, w_2), (w_3, w_4) and (w_5, w_6)[4]. For mutation 1, let position 2 be selected randomly. Each token of the word vector (w_3, w_4) will be changed by some token from the global vocabulary using SCP. Then, after change, the string will look like $< w_1\ w_2\ w_3^{new}\ w_4^{new}\ w_5\ w_6 >$. If mutation 2 is selected, a centroid will be removed from the string. Let centroid 3 be selected for deletion. The new string will look like $< w_1\ w_2\ w_3\ w_4 >$. In case of mutation 3, a new centroid will be added to the string. A new cluster centroid is generated choosing p=2 number of tokens from the global vocabulary. Let the randomly generated new cluster centroid to be added to the string be (w_7, w_8). After inclusion of this centroid, the string will be $< w_1\ w_2\ w_3\ w_4\ w_5\ w_6\ w_7\ w_8 >$. In our experiments, we have associated equal probability to each of these mutation operations. Thus, each mutation is applied in 33% cases of the cases.

5 Experimental Setup

5.1 Datasets

The main gold standards used for the evaluation of SRC algorithms are ODP-239 and MORESQUE[5]. In ODP-239 (Carpineto and Romano, 2010), each document is represented by a title and a web snippet and the subtopics are chosen from the top levels of DMOZ[6]. On the other hand, the subtopics in MORESQUE (Navigli and Crisafulli, 2010) follow a more natural distribution as they are defined based on the disambiguation pages of Wikipedia. As such, the subtopics cover most of the query-related senses. However, not all queries are Wikipedia related or ambiguous (e.g. "Olympic Games", which Wikipedia entry is not ambiguous, although there are many events related to this topic). As a consequence, it is clear that different results can be obtained from one data set to another. A quick summary of both data sets is presented in Table 1.

Dataset	# of queries	# of Subtopics Avg / Min / Max	# of Snippets
ODP-239	239	10 / 10 / 10	25580
MORESQUE	114	6.7 / 2 / 38	11402

Table 1: SRC gold standard data sets.

5.2 Evaluation Metrics

A successful SRC system must evidence high quality level clustering. Each query subtopic should ideally be represented by a unique cluster containing all the relevant web pages inside. However, determining a unique and complete metric to evaluate the performance of a clustering algorithm is still an open problem (Amigó et al., 2013).

In this paper, we propose to use the F_{b^3}-measure (Amigó et al., 2009) to explore the Pareto front. In particular, F_{b^3} has been defined to evaluate cluster homogeneity, completeness, rag-bag and size-vs-quantity constraints. F_{b^3} is a function of $Precision_{b^3}$ (P_{b^3}) and $Recall_{b^3}$ (R_{b^3}). All metrics are defined in Equation 5

$$F_{b^3} = \frac{2 * P_{b^3} * R_{b^3}}{P_{b^3} + R_{b^3}}, \quad P_{b^3} = \frac{1}{N}\sum_{i=1}^{K}\sum_{d_j \in \pi_i}\frac{1}{|\pi_i|}\sum_{d_l \in \pi_i} g^*(d_j, d_l), \quad R_{b^3} = \frac{1}{N}\sum_{i=1}^{K}\sum_{d_j \in \pi_i^*}\frac{1}{|\pi_i^*|}\sum_{d_l \in \pi_i^*} g(d_j, d_l) \qquad (5)$$

where π_i is i^{th} cluster, π_i^* is the gold standard of the category i, and $g^*(.,.)$ and $g(.,.)$ are defined as follows:

$$g^*(d_i, d_j) = \begin{cases} 1 \Leftrightarrow \exists l: & d_i \in \pi_l^* \wedge d_j \in \pi_l^* \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad g(d_i, d_j) = \begin{cases} 1 \Leftrightarrow \exists l: & d_i \in \pi_l \wedge d_j \in \pi_l \\ 0 & \text{otherwise} \end{cases}.$$

[4] with p=2.

[5] AMBIENT has received less attention since the creation of ODP-239.

[6] http://www.dmoz.org [Last access: 14/03/2014].

Most SRC studies have also used the F_β-measure (F_β), which is defined in Equation 6.

$$F_\beta = \frac{(\beta^2+1)*P*R}{\beta^2*P+R}, \quad P = \frac{TP}{TP+FP}, \quad R = \frac{TP}{TP+FN} \tag{6}$$

where

$$TP = \sum_{i=1}^{K} \sum_{d_j \in \pi_i^*} \sum_{\substack{d_l \in \pi_i^* \\ l \neq j}} g(d_i, d_j), FP = \sum_{i=1}^{K} \sum_{d_j \in \pi_i} \sum_{\substack{d_l \in \pi_i \\ l \neq j}} (1 - g^*(d_i, d_j)), FN = \sum_{i=1}^{K} \sum_{d_j \in \pi_i^*} \sum_{\substack{d_l \in \pi_i^* \\ l \neq j}} (1 - g(d_i, d_j)).$$

6 Results and Discussion

In this evaluation, we used the open source framework GATE (Cunningham et al., 2013) without stopword removal for web snippet tokenization[7]. We executed *MOO-clus* over ODP-239 and MORESQUE. The parameters of *MOO-clus* are: $T_{min} = 0.01$, $T_{max} = 100$, $\alpha = 0.85$, $HL = 10$, $SL = 20$ and $iter = 15$. Note that, they have been determined after conducting a thorough sensitivity study. A first set of experiments have been conducted for different p values of tokens present in the centroid, namely in the range 2 to 5 in order to understand the behavior of *MOO-clus* w.r.t. centroid size[8]. Note that the partition with maximum F_{b^3} is choosen for each size of p[9]. Overall results are shown in Table 2.

	MORESQUE				ODP-239			
	MOO-clus				*MOO-clus*			
	2	3	4	5	2	3	4	5
$F_b{}^3$	0.477	0.491	0.497	**0.502**	0.478	0.481	**0.484**	0.481
F_1	0.661	0.666	**0.675**	0.658	0.379	0.379	**0.384**	0.381
F_2	0.750	**0.768**	0.764	0.742	0.534	0.536	**0.537**	0.535
F_5	0.831	**0.862**	0.846	0.820	0.717	**0.720**	0.716	0.715

Table 2: Evaluation results of *MOO-clus* over MORESQUE and ODP239 data sets.

Results show that for MORESQUE, *MOO-clus* obtains the highest F_{b^3} value for p=5. In particular, performance increases for higher values of p. For ODP-239, best results are reported for p=4, but evidence less sensitivity to the number of words in the centroids. Indeed, a marginal difference is obtained between all runs. In terms of F_β, the same behaviour is obtained for ODP-239. But, for MORESQUE, best results are provided for smaller values of p, namely p=3.

Two important comments must be pointed at. In the first place, F_{b^3} shows a steady behaviour compared to F_β when the data set changes. The conclusions drawn in (Amigó et al., 2009) reporting the superiority of F_{b^3} over F_β seem to be verified for the specific case of SRC. In the second place, *MOO-clus* evidences a marginal sensitivity to different p values. Indeed, for ODP-239, changing p between 2 and 5 words has a negligible impact on F_{b^3}. The figures show a different behaviour for MORESQUE but this can easily be explained. In MORESQUE, less queries are provided for test and the number of reference clusters varies between 2 and 38, with a majority of queries containing very few clusters (the average cluster size is 6.7). As such, small clustering errors may result in high deviations in the evaluation metrics. So, p can be seen as a non influent parameter for clustering purposes. In fact, increasing the value of p may exclusively allow a more descriptive power for cluster labeling.

We also compared *MOO-clus* to the current state-of-the-art text-based (endogenous) SRC algorithms: STC (Zamir and Etzioni, 1998), LINGO (Osinski and Weiss, 2005), OPTIMSRC (Carpineto and Romano, 2010), Bisecting Incremental K-means (BIK), GK-means (Moreno et al., 2013) and the combination STC-LINGO (Moreno and Dias, 2014). The results are illustrated in Table 3 where we provide values for all the metrics for open source implementations and reported values in the literature for the

[7]Note that keeping stop words is a challenging task as most methodologies withdraw these elements as they are hard to handle. This decision is supported by the fact that we aim to produce as much as possible language-independent solutions.

[8]Note that to ease the user effort in searching for information, the cluster label must be small and expressive. Typical configurations range between 3 to 5 to include multiword expressions.

[9]F_β metrics are calculated over the partition with highest F_{b^3} value.

other experiments i.e. OPTIMSRC, GK-means and STC-LINGO. In particular, the Min (resp. Max) column refers to the worst (resp. best) performance when varying p, the size of the centroid.

The results of Table 3 clearly show the performance improvements of our proposed methodology over existing text-based techniques for both data sets and most evaluation metrics. For ODP-239, *MOO-clus* attains the highest values with respect to F_1, F_2, F_5 and F_{b^3} metrics against all existing endogenous algorithms. For MORESQUE, our algorithm reaches highest performance over all state-of-the-art algorithms for F_1 and F_{b^3} metrics but marginally fails for F_2 and F_5 against GK-means.

		MOO-clus		*SOO SRC*				*Combination of SOO SRC*	
		Min	*Max*	GK-means	STC	LINGO	BIK	OPTIMSRC	STC-LINGO
MORESQUE	F_1	0.658	**0.675**	0.665	0.455	0.326	0.317	N/A	0.561
	F_2	0.742	0.768	**0.770**	0.392	0.260	0.269	N/A	N/A
	F_5	0.820	0.862	**0.872**	0.370	0.237	0.255	N/A	N/A
	${F_b}^3$	0.477	**0.502**	0.482	0.460	0.399	0.315	N/A	0.498
ODP-239	F_1	0.379	**0.384**	0.366	0.324	0.273	0.200	0.313	0.362
	F_2	0.534	**0.537**	0.416	0.319	0.167	0.173	0.341	N/A
	F_5	0.715	**0.720**	0.462	0.322	0.153	0.165	0.380	N/A
	${F_b}^3$	0.478	**0.484**	0.452	0.403	0.346	0.307	N/A	0.425

Table 3: Comparative results with respect to F_β and F_{b^3} metrics over the ODP-239 and MORESQUE datasets obtained by different SRC techniques.

It is important to notice that OPTIMSRC and STC-LINGO can be viewed as a combination of different SRC SOO solutions but still casted to a SOO solution. These previous results report interesting issues for SRC and confort the idea that the combination of different objective functions may lead to enhanced SRC algorithms. But, *MOO-clus* is capable to find better partitions than OPTIMSRC and STC-LINGO for all data sets and all evaluation metrics as reported in Table 3.

It is important to notice that the *MOO-clus* provides a set of partitions with automatic definition of the number of clusters. So, defining one unique solution is an important issue for SRC. So far, we have provided results for the best partition evaluated by F_{b^3}. However, deeper analysis of all the partitions on the Pareto front must be endeavoured. Results are reported for F_{b^3} only as all other metrics behave correspondingly and are reported in Table 4.

	MORESQUE				ODP-239			
	2	3	4	5	2	3	4	5
Min	0.428	0.464	0.464	0.462	0.396	0.401	0.403	0.408
Max	0.477	0.491	0.497	0.502	0.478	0.481	0.484	0.481
Avg.	0.454	0.479	0.482	0.486	0.443	0.447	0.448	0.449

Table 4: F_{b^3} evaluation results of the Pareto front.

Figures show the validity of each individual solution of the Pareto front. In the worst case, *MOO-clus* produces similar results compared to the hard baseline STC. On average, it reaches the results of GK-means and the highest performance values can be found on the Pareto front. The correct identification of the best partition is still an open issue and can be compared to the automatic selection of K clusters, which is a hard task as shown in recent studies (Scaiella et al., 2012; Marco and Navigli, 2013).

7 Conclusions

In this paper, we proposed the first attempt[10] to define the SRC task as a multi-objective problem. For that purpose, we defined two objective functions, which are simultaneously optimized through the archived multi-objective simulated annealing framework called AMOSA. A correct definition of the task allowed to take advantage of the most recent advances in terms of endogenous SRC algorithms as well as the most powerful techniques for multi-objective clustering. The performance of *MOO-clus* has been evaluated over two gold standard data sets, ODP-239 and MORESQUE for different evaluation metrics, F_1 and F_{b^3}.

[10] As far as we know.

Results showed that our proposal steadily outperforms all existing state-of-the-art text-based endogenous SRC algorithms and approaches recent knowledge-driven exogenous strategies (Scaiella et al., 2012), which reach F_1=0.413 for ODP-239[11].

As future works, we propose to use MOO clustering in a strict meta learning way, where any labeled-based SOO solution is defined by specific $Compactness$ and $Separability$ functions. Another research direction is the definition of the Dual representation proposed by (Moreno et al., 2014) as a MOO problem. Finally, new objective functions can be defined to measure the quality of the labels, which may integrate meaningful multiword expressions or named entities.

Acknowledgement

We would like to thank the CNRS to provide Sriparna Saha with a 6 months internship at the GREYC Laboratory of the Normandie University.

References

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4):461–486.

Enrique Amigó, Julio Gonzalo, and Felisa Verdejo. 2013. A general evaluation measure for document organization tasks. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 643–652.

Sanghamitra Bandyopadhyay, Anirban Mukhopadhyay, and Ujjwal Maulik. 2007. An improved algorithm for clustering gene expression data. *Bioinformatics*, 23(21):2859–2865.

Sanghamitra Bandyopadhyay, Sriparna Saha, Ujjwal Maulik, and Kalyanmoy Deb. 2008. A simulated annealing-based multiobjective optimization algorithm: Amosa. In *IEEE transactions on evolutionary computation*, pages 269–283.

James C. Bezdek. 1981. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum, New York.

Thorsten Brants and Alex Franz. 2006. Web 1t 5-gram.

Claudio Carpineto and Giovanni Romano. 2010. Optimal meta search results clustering. In *33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 170–177.

Claudio Carpineto, Stanislaw Osinski, Giovanni Romano, and Dawid Weiss. 2009. A survey of web clustering engines. *ACM Computing Surveys*, 41(3):1–38.

Hamish Cunningham, Valentin Tablan, Angus Roberts, and Kalina Bontcheva. 2013. Getting more out of biomedical documents with gate's full lifecycle open source text analytics. *PLoS Computational Biology*, 9(2):e1002854.

Joaquim Ferreira da Silva, Gaël Dias, Sylvie Guilloré, and José Gabriel Pereira Lopes. 1999. Using localmaxs algorithm for the extraction of contiguous and non-contiguous multiword lexical units. In *Proceedings of 9th Portuguese Conference in Artificial Intelligence (EPIA)*, pages 113–132.

Kalyanmoy Deb. 2009. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley.

Julia Handl and Joshua Knowles. 2007. An evolutionary approach to multiobjective clustering. *IEEE Transactions on Evolutionary Computation*, 11:56–76.

Antonio D. Marco and Roberto Navigli. 2013. Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, 39(4):1–43.

Ujjwal Maulik, Sanghamitra Bandyopadhyay, and Anirban Mukhopadhyay. 2011. *Multiobjective Genetic Algorithms for Clustering - Applications in Data Mining and Bioinformatics*. Springer.

José G. Moreno and Gaël Dias. 2014. Easy web search results clustering: When baselines can reach state-of-the-art algorithms. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 1–5.

[11]Note that results of (Marco and Navigli, 2013) are not reported in this paper as the authors do not use the standard versions of MORESQUE and do not provide experiments for ODP-239.

José G. Moreno, Gaël Dias, and Guillaume Cleuziou. 2013. Post-retrieval clustering using third-order similarity measures. In *51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 153–158.

José G. Moreno, Gaël Dias, and Guillaume Cleuziou. 2014. Query log driven web search results clustering. In *Proceedings of the 37th Annual ACM SIGIR Conference (SIGIR)*.

Katharina Morik, Andreas Kaspari, Michael Wurst, and Marcin Skirzynsk. 2012. Multi-objective frequent termset clustering. *Knowledge Information Systems*, 30(3):715–738.

Anirban Mukhopadhyay and Ujjwal Maulik. 2009. Unsupervised pixel classification in satellite imagery using multiobjective fuzzy clustering combined with SVM classifier. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1132–1138.

Roberto Navigli and Giuseppe Crisafulli. 2010. Inducing word senses to improve web search result clustering. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 116–126.

Stanislaw Osinski and Dawid Weiss. 2005. A concept-driven algorithm for clustering search results. *IEEE Intelligent Systems*, 20(3):48–54.

Sriparna Saha and Sanghamitra Bandyopadhyay. 2010. A symmetry based multiobjective clustering technique for automatic evolution of clusters. *Pattern Recognition*, 43(3):738–751.

Ugo Scaiella, Paolo Ferragina, Andrea Marino, and Massimiliano Ciaramita. 2012. Topical clustering of search results. In *5th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 223–232.

Xuanli L. Xie and Gerardo Beni. 1991. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:841–847.

Oren Zamir and Oren Etzioni. 1998. Web document clustering: A feasibility demonstration. In *21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 46–54.

Query-by-Example Image Retrieval using Visual Dependency Representations

Desmond Elliott, Victor Lavrenko and **Frank Keller**
Institute of Language, Communication, and Computation
School of Informatics
University of Edinburgh
d.elliott@ed.ac.uk {vlavrenk,keller}@inf.ed.ac.uk

Abstract

Image retrieval models typically represent images as bags-of-terms, a representation that is well-suited to matching images based on the presence or absence of terms. For some information needs, such as searching for images of people performing actions, it may be useful to retain data about how parts of an image relate to each other. If the underlying representation of an image can distinguish between images where objects only co-occur from images where people are interacting with objects, then it should be possible to improve retrieval performance. In this paper we model the spatial relationships between image regions using Visual Dependency Representations, a structured image representation that makes it possible to distinguish between object co-occurrence and interaction. In a query-by-example image retrieval experiment on data set of people performing actions, we find an 8.8% relative increase in MAP and an 8.6% relative increase in Precision@10 when images are represented using the Visual Dependency Representation compared to a bag-of-terms baseline.

1 Introduction

Every day millions of people search for images on the web, both professionally and for personal amusement. The majority of image searches are aimed at finding a particular named entity, such as *Justin Bieber* or *supernova*, and a typical image retrieval system is well-suited to this type of information need because it represents an image as a bag-of-terms drawn from data surrounding the image, such as text, manual tags, and anchor text (Datta et al., 2008). It is not always possible to find useful terms in the surrounding data; the last decade has seen advances in automatic methods for assigning terms to images that have neither user-assigned tags, nor a textual description (Duygulu et al., 2002; Lavrenko et al., 2003; Guillaumin and Mensink, 2009). These automatic methods learn to associate the presence and absence of labels with the visual characteristics of an image, such as colour and texture distributions, shape, and points of interest, and can automatically generate a bag of terms for an unlabelled image.

It is important to remember that not all information needs are entity-based: people also search for images reflecting a mood, such as *people having fun at a party*, or an action, such as *using a computer*. The bag-of-terms representation is limited to matching images based on the *presence or absence* of terms, and not the *relation* of the terms to each other. Figures 1(a) and (b) highlight the problem with using unstructured representations for image retrieval: there is a person and a computer in both images but only (a) depicts a person actually using the computer. To address this problem with unstructured representations we propose to represent the structure of an image using the Visual Dependency Representation (Elliott and Keller, 2013). The Visual Dependency Representation is a directed labelled graph over the regions of an image that captures the spatial relationships between regions. The representation is inspired by evidence from the psychology literature that people are better at recognising and searching for objects when the spatial relationships between the objects in the image are consistent with our expectations of the world.(Biederman, 1972; Bar and Ullman, 1996). In an automatic image description task, Elliott

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 109–120, Dublin, Ireland, August 23-29 2014.

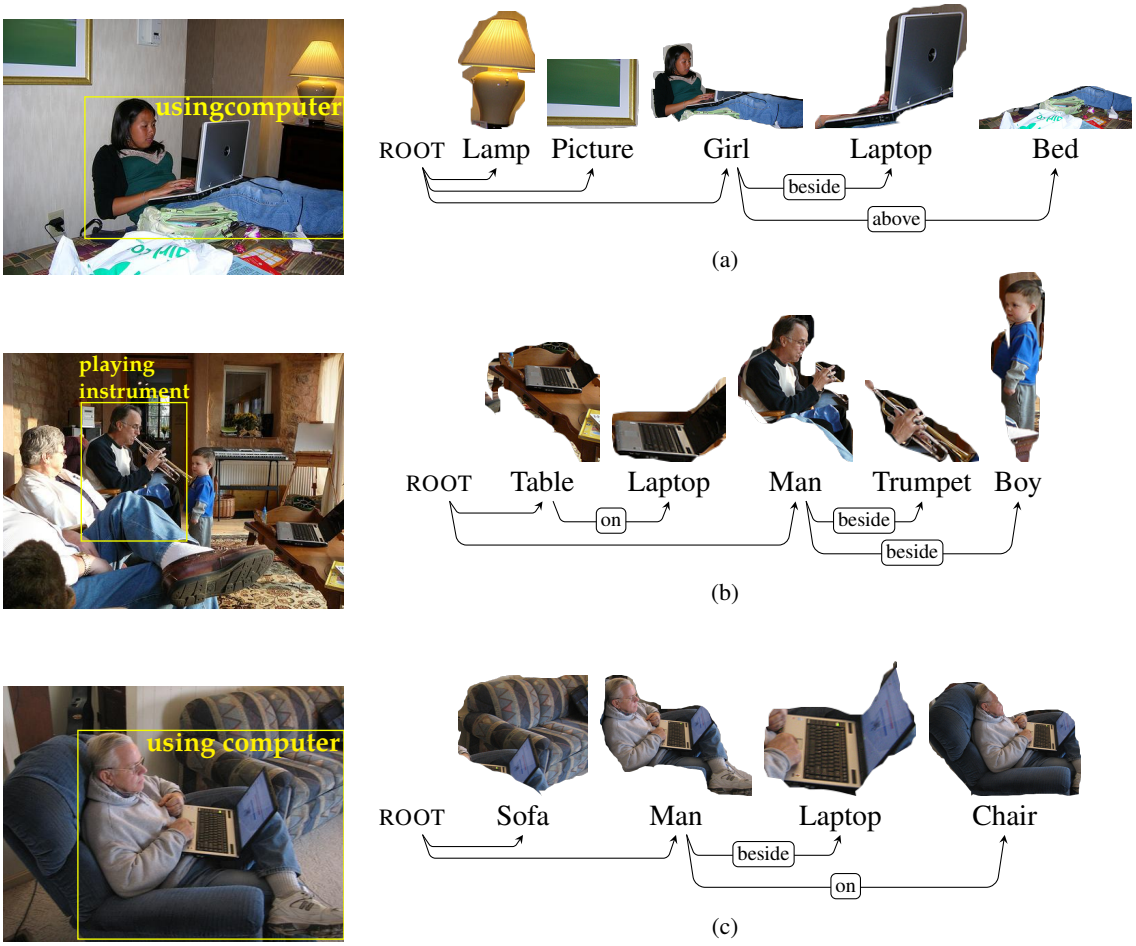


Figure 1: Three examples of images depicting a person and a computer, alongside a respective Visual Dependency Representation for each image. The bag-of-terms representation can be observed in the annotated regions of the Visual Dependency Representations. In (a) and (c) there is a person using a laptop, whereas in (b) the man is actually using the trumpet. The gold-standard action annotation is shown in the yellow bounding box.

and Keller (2013) showed that encoding the spatial relationships between objects in the Visual Dependency Representation helped to generate significantly better descriptions than approaches based on the spatial proximity of objects (Farhadi et al., 2010) or corpus-based models (Yang et al., 2011). In this paper we study whether the Visual Dependency Representation of images can improve the performance of query-by-example image retrieval models. The main finding is that encoding images using the Visual Dependency Representation leads to significantly better retrieval accuracy compared to a bag-of-terms baseline, and that the improvements are most pronounced for transitive verbs.

2 Related Work

2.1 Representing Images

A central problem in image retrieval is how to abstractly represent images (Datta et al., 2008). A bag-of-terms representation of an image is created by grouping visual features, such as color, shape (Shi and Malik, 2000), texture, and interest points (Lowe, 1999), in a vector or as a probability distribution over the features. Image retrieval can then be performed by trying to find the best matchings of terms across an image collection. Spatial Pyramid Matching is an approach to constructing low-level image representations that capture the relationships between features at differently sized partitions of the image (Lazebnik et al., 2006). This approach has proven successful for scene categorisation tasks. An alternative approach to representing images is to learn a mapping (Duygulu et al., 2002; Lavrenko et al.,

2003; Guillaumin and Mensink, 2009) between the bags-of-terms and object tags. An image can then be represented as a bag-of-terms and image retrieval is similar to text retrieval (Wu et al., 2012).

In this work, we represent an image as a directed acyclic graph over a set of labeled object region annotations. This representation captures the important spatial relationships between the image regions and makes it possible to distinguish between co-occurring regions and interacting regions.

2.2 Still-Image Action Recognition

One approach to recognizing actions is to learn appearance models for *visual phrases* and use these models to predict actions (Sadeghi and Farhadi, 2011). A visual phrase is defined as the people and the objects they interact with in an action. In this approach, a fixed number of visual phrase models are trained using the deformable parts object detector (Felzenszwalb et al., 2010) and used to perform action recognition.

An alternative approach is to model the relationships between objects in an image, and hence the visible actions, as a Conditional Random Field (CRF), where each node in the field is an object and the factors between nodes correspond to features that capture the relationships between the objects (Zitnick et al., 2013). The factors between object nodes in the CRF include object occurrence, absolute position, person attributes, and the relative location of pairs of objects. This model has been used to generate novel images of people performing actions and to retrieve images of people performing actions.

Most recently, actions have been predicted in images by selecting the most likely verb and object pair given a set of candidate objects detected in an image (Le et al., 2013a). The verb and object is selected amongst those that maximize the distributional similarity of the pair in a large and diverse collection of documents. This approach is most similar to ours but it relies on an external corpus and, depending on the text collections used to train the distributional model, will compound the problem of co-occurrence of objects instead of the relationships between the objects.

The work presented in this paper uses ground-truth annotation for region labels, an assumption similar to (Zitnick et al., 2013), but requires no external data to make predictions of the relationships between objects, unlike the approach of (Le et al., 2013a). The directed acyclic graph representation we propose for images can be seen as a latent representation of the depicted action in the image, where the spatial relationships between the regions capture the different types of actions.

3 Task and Baseline

In this paper we study the task of query-by-example image retrieval within the restricted domain of images depicting actions. More specifically, given an image that depicts a given action, such as *using a computer*, the aim of the retrieval model is to find all other images in the image collection that depict the same action. We define an action as an event involving one or more entities in an image, e.g., *a woman running* or *boy using a computer*, and assume all images have been manually annotated for objects. This assumption means we can explore the utility of the Visual Dependency Representation without the noise introduced by automatic computer vision methods. The data available to the retrieval models can be seen in Figure 1, and Section 5 provides further details about the different sources of data The action label - which is only used for evaluation - is shown in the labelled bounding box, and the Visual Dependency Representation - not used by the baseline model - is shown as a tree at the bottom of the figure.

The main hypothesis explored in this paper is that the accuracy of an image retrieval model will increase if the representation encodes information about the relationships between the objects in images. This hypothesis is tested by encoding images as either an unstructured bag-of-terms representation or as the structured Visual Dependency Representation. The Bag-of-Terms baseline represents the query image and the image collection as an unstructured bags-of-terms vector. All of the models used to test the main hypothesis use the cosine similarity function is to determine the similarity of the query image to other images in the collection, and thus to generate a ranked list from the similarity values.

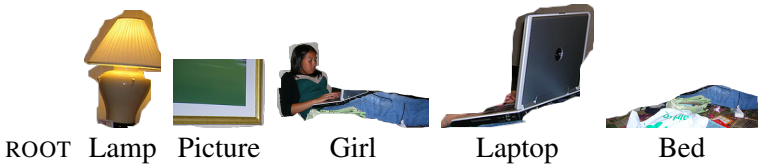
4 Visual Dependency Representation

The Visual Dependency Representation (VDR) is a structured representation of an image that captures the spatial relationships between pairs of image regions in a directed labelled graph. The Visual Dependency Grammar defines eight possible spatial relationships between pairs of regions, as shown in Table 1. The relationships in the grammar were designed to provide *sufficient* coverage of the types of spatial relationships required to describe the data, and are mathematically defined in terms of pixel overlap, distance between regions, and the angle between regions. The frame of reference for annotating spatial relationships is the image itself and not the object in the image, and angles and distance measurements are taken or estimated from the centroids of the regions. The VDR of an image is created by a trained human annotator in a two-stage process:

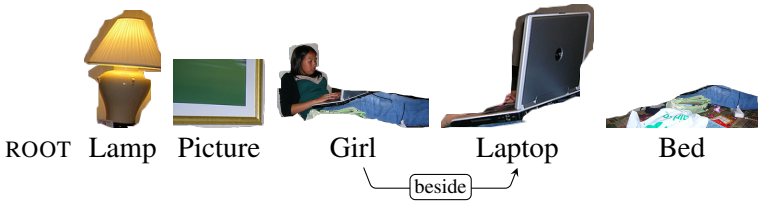
1. The annotator draws and labels boundaries around the parts of the image they think contribute to defining the action depicted in the image, and the context within which the action occurs;

2. The annotator draws labelled directed edges between the annotated regions that captures how the relationships between the image convey the action. In Section 4.1, we will explain how to automate the second stage of the process from a collection of labelled region annotations.

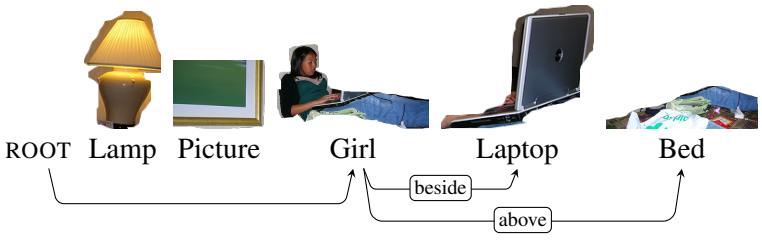
In addition to the annotated image regions, a VDR also contains a ROOT node, which acts as a placeholder for the image. In the remainder of this section we describe how a gold-standard VDR is created by a human annotator. The starting point for the VDR in Figure 1(a) is the following set of regions and the ROOT node:



First, the regions are attached to each other based on how the relationship between the objects contributes to the depicted action. In Figure 1(a), the Girl is *using* the Laptop, therefore a labelled directed edge is created from the Girl region to the Laptop region. The spatial relationship is labelled as BESIDE.



The Girl is also attached to the Bed because the bed supports her body. The spatial relation label is ABOVE because it expresses the spatial relationship between the regions, not the semantic relationship ON. ROOT is attached to the Girl without an edge label to symbolize that she is an actor in the image.



Now the regions that are not concerned with the depicted action are first attached to each other if there is a clear spatial relationship between them (for an example, see Figure 1(b), where the laptop is attached to the table because it is sitting on the table), and then to the ROOT node to signify that they do not play a part in the depicted action. In this example, neither the Lamp nor the Picture are related to the action of using the computer, so they are attached to the ROOT node.

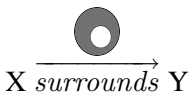
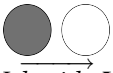
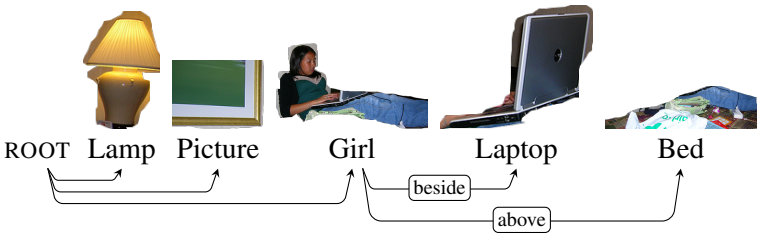
 X $\overrightarrow{on}$ Y	More than 50% of the pixels of region X overlap with region Y.	 X $\overrightarrow{surrounds}$ Y	The entirety of region X overlaps with region Y.
 X $\overrightarrow{beside}$ Y	The angle between the centroid of X and the centroid of Y lies between 315° and 45° or 135° and 225°.	 X $\overrightarrow{opposite}$ Y	Similar to *beside*, but used when there X and Y are at opposite sides of the image.
 X $\overrightarrow{above}$ Y	The angle between X and Y lies between 225° and 315°.	 X $\overrightarrow{below}$ Y	The angle between X and Y lies between 45° and 135°.
 X $\overrightarrow{infront}$ Y	The Z-plane relationship between the regions is dominant.	 X $\overrightarrow{behind}$ Y	Identical to *infront* except X is behind Y in the Z-plane.

Table 1: Visual Dependency Grammar defines eight relations between pairs of annotated regions. To simplify explanation, all regions are circles, where X is the grey region and Y is the white region. All relations are considered with respect to the centroid of a region and the angle between those centroids.



This now forms a completed VDR for the image in Figure 1(a). This structured representation of an image captures the prominent relationship between the girl, the laptop, and the bed. There is no prominent relationship defined between the girl and either the lamp of the picture, in effect these regions have been relegated to background objects. The central hypothesis underpinning the Visual Dependency Representation is that images that contain similar VDR substructures are more likely to depict the same action than images that only contain the same set of objects. For example, the VDR for Figure 1(a) correctly captures the relationship between the people and the laptops, whereas this relationship is not present in Figure 1(b), where the person is playing a trumpet.

4.1 Predicting Visual Dependency Representations

We follow the approach of Elliott and Keller (2013) and predict the VDR y of an image over a collection of labelled region annotations $\mathbf{x}$. This task is framed as a supervised learning problem, where the aim is to construct a Maximum Spanning Tree from a fully-connected directed weighted graph over the labelled regions (McDonald et al., 2005). Reducing the fully-connected graph to the Maximum Spanning Tree removes the region–region edges that are not important in defining the prominent relationships between the regions in an image. The score of the VDR y over the image regions is calculated as the sum of the scores of the directed labelled edges:

$$score(\mathbf{x}, y) = \sum_{(a,b)\in y} \mathbf{w} \cdot \mathbf{f}(a, b) \quad (1)$$

where the score of an edge between image regions a and b is calculated using a vector of weighted feature functions $\mathbf{f}$. The feature functions characterize the image regions and the edge between pairs of regions, and include: the labels of the regions and the spatial relation annotated on the edge; the (normalized) distance between the centroids of the regions; the angle formed between the annotated regions, which is

mapped onto the set of spatial relations; the relative size of the region compared to the image; and the distance of the region centroid from the center of the image.

The model is trained over i instances of region-annotated images $\mathbf{x_i}$ associated with human-created VDR structures y_i, $I_{train} = \{\mathbf{x_i}, y_i\}$. The score of each edge a, b is calculated by applying the feature functions to the data associated with that edge, and this is performed over each edge in a VDR to obtain a score for a complete gold-standard structure. The parameters of the weight vector w are iteratively adjusted to maximise the score of the gold-standard structures in the training data using the Margin Infused Relaxation Algorithm (Crammer and Singer, 2002).

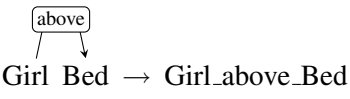
The test data contains i instances of region-annotated images with image regions $\mathbf{x_i}$, $I_{test} = \{\mathbf{x_i}\}$. The parsing model computes the highest scoring structure $\hat{y}_i$ for each instance in the test data by scoring each possible directed edge between pairs of regions in $\mathbf{x_i}$. This process forms a fully-connected graph over the image regions, from which the Maximum Spanning Tree is taken and returned as the predicted VDR.

We evaluate the performance of this VDR prediction model by comparing how well it can recover the manually created trees in the data set. This evaluation is performed on the development data in a 10-fold cross validation setting where each fold of the data is split 80%/10%/10%. Unlabelled directed accuracy means the model correctly proposes an edge between a pair of regions in the correct direction; Labelled directed accuracy means it additionally proposes the correct edge label. The baseline approach is to assume no latent image structure and attach all image regions to the ROOT node of the VDR; this achieves 51.6% labelled and unlabelled directed attachment accuracy. The accuracy of our automatic approach to VDR prediction is 61.3% labelled and 68.8% unlabelled attachment accuracy.

4.2 Comparing Visual Dependency Representations

It remains to define how to compare the Visual Dependency Representation of a pair of images. The most obvious approach is to use the labelled directed accuracy measurement used for the VDR prediction evaluation in the previous section, but we did not find significant improvements in retrieval accuracy using this method. We hypothesise that the lack of weight given to the edges between nodes in the Visual Dependency Representation results in this comparison function not distinguishing between object–object relationships that matter, such as PERSON $\overrightarrow{beside}$ BIKE, compared to ROOT $\rightarrow$ TREES. The former is a potential person–object relationship that explains the depicted event, whereas the latter is only a background object.

The approach we adopted in this paper is to compare Visual Dependency Representations of images by decomposing the structure into a set of labelled and a unlabelled parent–child subtrees in a depth-first traversal of the VDR. The decomposition process allows use to use the same similarity function as the Bag-of-Terms baseline model, removing the confound of choosing different similarity functions. The subtrees can be transformed into tokens and these tokens can be used as weighted terms in a vector representation. An example of a labelled transformation is shown below:



We now demonstrate the outcome of comparing images represented using either a vector that concatenates the decomposed transformed VDR and bag-of-terms, or a vector that contains only the bag-of-terms. In this demonstration, each term has a *tf-idf* weight of 1. The first illustration (*Similar*) compares images that depict the same underlying action: Figure 1 (a) and (c). The second illustration (*Dissimilar*) compares images that depict different actions: Figure 1 (a) and (b).

$$Similar : \cos(\text{VDR}_a, \text{VDR}_c) = 0.56 > \cos(\text{Bag}_a, \text{Bag}_c) = 0.52$$

$$Dissimilar : \cos(\text{VDR}_b, \text{VDR}_a) = 0.201 \ll \cos(\text{Bag}_b, \text{Bag}_a) = 0.4$$

It can be seen that when the images represent the same action, the decomposed VDR increases the similarity of the pair of images compared to the bag-of-terms representation; and when images do not

represent the same action, the decomposed VDR yields a lower similarity than the bag-of-terms representation. These illustrations confirm that Visual Dependency Representations can be used to distinguish the difference between presence or absence of objects, and the prominent relationships between objects.

5 Data

We use an existing dataset of VDR-annotated images to study whether modelling the structure of an image can improve image retrieval in the domain of action depictions. The data set of Elliott and Keller (2013) contains 341 images annotated with region annotations, three visual dependency representations per image (making a total of 1,023 instances), and a ground-truth action label for each image. An example of the annotations can be seen in Figure 1. The image collection is drawn from the PASCAL Visual Object Classification Challenge 2011 action recognition taster and covers a set of 10 actions (Everingham et al., 2011): riding a bike, riding a horse, reading, running, jumping, walking, playing an instrument, using a computer, taking a photo, and talking on the phone.

Image Descriptions

Each image is associated with three human-written descriptions collected from untrained annotators on Amazon Mechanical Turk. The descriptions do not form any part of the models presented in the current paper; they were used in the automatic image description task of Elliott and Keller (2013). Each description contains two sentences: the first sentence describes the action depicted in the image, and the second sentence describes other objects not involved in the action. A two sentence description of an image helps distinguish objects that are central to depicting the action from objects that may be distractors.

Region Annotations

The images contain human-drawn labelled region annotations. The annotations were drawn using the LabelMe toolkit, which allows for arbitrary labelled polygons to be created over an image (Russell et al., 2008). The annotated regions were restricted to those present in at least one of three human-written descriptions. To reduce the effects of label sparsity, frequently occurring equivalent labels were conflated, i.e., man, child, and boy $\rightarrow$ person; bike, bicycle, motorbike $\rightarrow$ bike; this reduced the object label vocabulary from 496 labels to 362 labels. The data set contains a total of 5,034 region annotations, with a mean of 4.19 ± 1.94 annotations per image.

Visual Dependency Representations

Recall that each image is associated with three descriptions, and that people were free to decide how to describe the action and background of the image. The differences between how people describe images leads to the creation of one Visual Dependency Representation per image–description pair in the data set, resulting in a total of 1,023 instances. The process for creating a visual dependency representation of an image is described in Section 4. The annotated dataset comprises a total of 5,748 spatial relations, corresponding to a mean of 4.79 ± 3.51 relations per image. Elliott and Keller (2013) report inter-annotator agreement on a subset of the data at 84% agreement for labelled directed attachments and 95.1% for unlabelled directed attachments.

Action Labels

The original PASCAL action recognition dataset contains ground truth action class annotations for each image. These annotations are in the form of labelled bounding boxes around the person performing the action in the image. The action labels are only used as the gold-standard relevance judgements for the query-by-example image retrieval experiments.

6 Experiments

In this section we present the results of a query-by-example image retrieval experiment to determine the utility of the Visual Dependency Representation compared to a bag-of-terms representation. In this

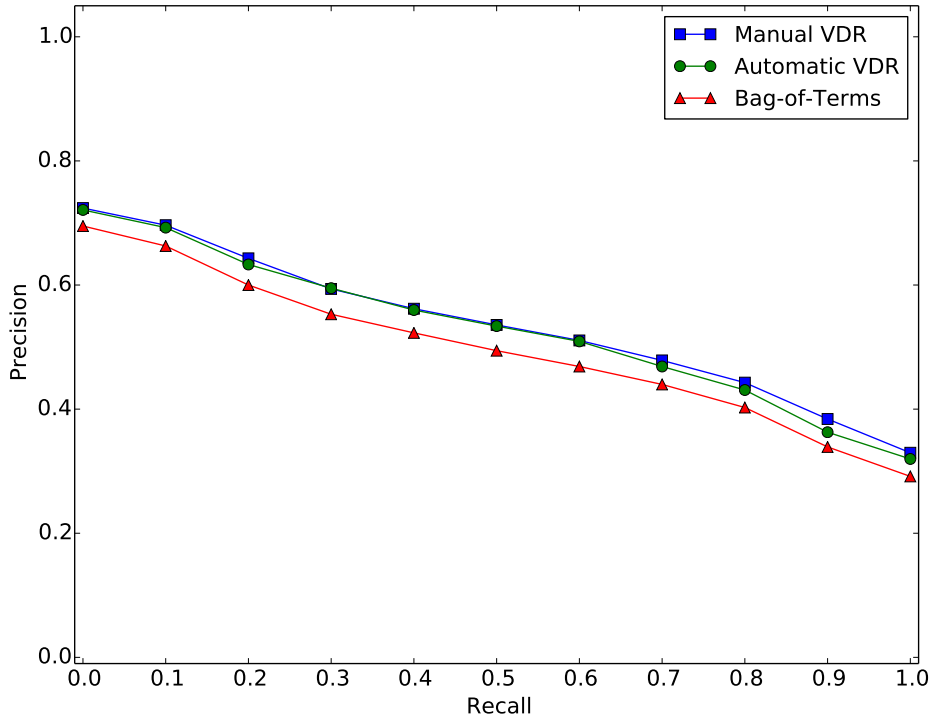


Figure 2: Average 11-point precision/recall curves show that the VDR-based retrieval models are consistently better than the Bag-of-Terms model.

experiment, a single image (the query image) is used to rank the images in the test collection, where the goal is to construct a ranking where the top images depict the same action as the query image.

6.1 Protocol

The image retrieval experiment is performed using 10-fold cross-validation in the following manner. The 341 images in the dataset are randomly partitioned into 80%/10%/10% splits, resulting in 1011 test queries[1]. For each query we compute average precision and Precision@10 of the ranked list, and use the resulting values to test the statistical significance of the results.

The *training set* is used to train the VDR prediction model and to estimate inverse document frequency statistics. During the training phase, the VDR-based models have access to region boundaries, region labels and three manually-created VDRs for each training image. In the *test set*, all models have access to the region boundaries and labels for each image. Each image in the test set forms a query and the models produce a ranked list of the remaining images in the test collection. Images are marked for relevance as follows: a image at rank r is considered *relevant* if it has the same action label as the query image; otherwise it is *non-relevant*. The *dev set* was used to experiment with different matching functions and to optimise the feature functions used in the VDR prediction model.

6.2 Models

We compare the retrieval accuracy of three approaches: Bag-of-Terms uses an unstructured representation for each image. A *tf-idf* weight is assigned to each region label in an image, and the cosine measure is used to calculate the similarity of images. This model allows us to compare the usefulness of a structured vs. unstructured image representation. Automatic VDR is a model using the VDR prediction method from Section 4.1, and Manual VDR uses the gold-standard data described in Section 5. Both

[1]Recall there are three Visual Dependency Representations for each image. The partitions are the same as those used in the VDR prediction experiment in Section 4.1

	MAP	P@10
Manual VDR	$0.514^{\star\dagger}$	$0.454^{\star}$
Automatic VDR	$0.508^{\star}$	$0.451^{\star}$
Bag-of-Terms	0.467	0.415

Table 2: Overall Mean Average Precision and Precision@10 images. The VDR-based models are significantly better than the Bag-of-Terms model, supporting the hypothesis that modelling the structure of an image using the Visual Dependency Representation is useful for image retrieval. $\star$: significantly different than Bag-of-Terms at $p < 0.01$; †: significantly different than Automatic VDR at $p < 0.01$.

of the VDR-based models have a tf-idf weight assigned to the transformed decomposed terms and the cosine similarity measure is used to calculate the similarity of images.

6.3 Results

Figure 2(a) shows the interpolated precision/recall curve and Table 2 shows the Mean Average Precision (MAP) and Precision at 10 retrieved images (P@10). The MAP of the Automatic VDR model increases by 8.8% relative to the Bag-of-Terms model, and a relative improvement up to 10.1% would possible if we had a better structure prediction model, as evidenced by Manual VDR. Furthermore, if we assume a user will only view the top results returned by the retrieval model, then P@10 increases by 8.6% when we model the structure of an image, relative to using an unstructured representation; a relative improvement of up to 9.4% would be possible if we had a better image parser.

To determine whether the differences are statistically significant, we perform the Wilcoxon Signed Ranks Test on the average precision and P@10 values over the 1011 queries in our cross-validation data set. The results support the main hypothesis of this paper: structured image representations allow us to find images depicting actions more accurately than the standard bag-of-terms representation. We find significant differences in average precision and P@10 between the Bag-of-Terms baseline and both Automatic VDR ($p < 0.01$) and Manual VDR ($p < 0.01$). This suggests that structure is very useful in the query-by-example scenario. We find a significant difference in average precision between Automatic VDR and Manual VDR ($p < 0.01$), but no difference in P@10 between Automatic VDR and Manual VDR ($p = 0.442$).

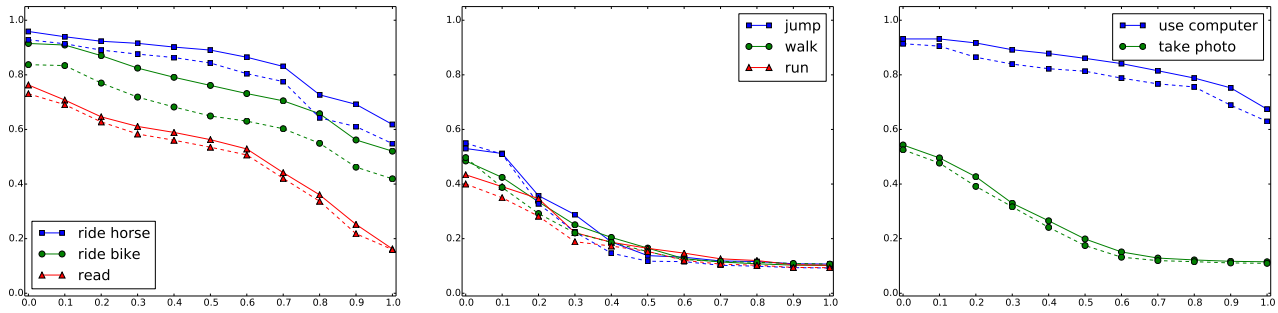
6.4 Retrieval Performance by Type of Action and Verb

We now analyse whether image structure is useful when the action does not require a direct object. The analysis presented here compares the Bag-of-Terms model against the Automatic VDR model because there was no significant difference in P@10 between the Automatic and Manual VDR models. Table 3 shows the MAP and Precision@10 per type of action. Figure 3 shows the precision/recall curves for (a) transitive verbs, (b) intransitive verbs, and (c) light verbs.

In Figure 3(a), it can be seen that the actions that can be classified as transitive verbs benefit from exploiting the structure encoded in the Visual Dependency Representation. The only exception is for the action *to read*, which frequently behaves as an intransitive verb: *the man reads on a train*. The consistent improvement in both the entirety of the ranked list and at the top of the ranked list can be seen in the MAP and P@10 results in Table 3.

Figure 3(b) shows that there is a small increase in retrieval performance for intransitive verbs compared to the transitive verbs. We conjecture this is because there are fewer objects to annotate in an image when the verb does not require a direct object. The summary results for the intransitive verbs in Table 3 confirm the small but insignificant increase in MAP and P@10.

Finally, the light verbs, shown in Figure 3(c), exhibit variable behaviour in retrieval performance. One reason for this could be that if the light verb encodes information about the object, as in *using a computer*, then the computer can be annotated in the image, and thus it acts as a transitive verb. Conversely, when



(a) (b) (c)

Figure 3: Precision/recall curves grouped by the type of verb. The solid lines represent the Automatic VDR model; the dashed lines represent the Bag-of-Terms model; y-axis is Precision, and the x-axis is Recall. (a) Images depicting transitive verbs benefit the most from the Visual Dependency Representation and are easiest to retrieve. (b) Intransitive verbs are difficult to retrieve and there is is a negligible improvement in performance when using Visual Dependency Representation. (c) Light verbs benefit from the Visual Dependency Representation depending on the type of the object involved in the action.

	MAP		P@10	
	VDR	Bag	VDR	Bag
Ride bike	0.721⋆	0.601	0.596⋆	0.513
Ride horse	0.833⋆	0.768	0.787⋆	0.726
Talk on phone	0.762⋆	0.679	0.666⋆	0.582
Play instrument	0.774⋆	0.705	0.634⋆	0.586
Read	0.483	0.454	0.498	0.475
Walk	0.198	0.186	0.184	0.174
Run	0.193	0.165	0.151	0.132
Jump	0.211	0.189	0.142	0.136
Use computer	0.814⋆	0.761	0.694⋆	0.648
Take photo	0.241	0.223	0.212	0.198

Table 3: Mean Average Precision and Precision@10 for each action in the data set, grouped into transitive (top), intransitive (middle), and light (bottom) verbs. VDR is the Automatic VDR model and Bag is the Bag-of-Terms model. It can be seen that the Automatic VDR retrieval model is consistently better than the Bag-of-Terms model on both MAP and Precision@10. ⋆: the Automatic VDR model is significantly different than Bag-of-Terms at $p < 0.01$.

the light verb conveys information about the outcome of the event, as in the action *take a photograph*, the outcome is rarely possible to annotate in an image, and so no improvements can be gained from structured image representations.

6.5 Discussion

In our experiments we observed that all models can achieve high precision at very low levels of recall. We found that this happens for testing images that are almost identical to the query image. For such images, objects that are unrelated to the target action form an effective context, which allows this image to be placed at the top of the ranking. However, near-identical images are relatively rare, and performance degrades for higher levels of recall.

It is surprising that image retrieval using automatically predicted VDR model is statistically indistinguishable from the manually crafted VDR model, given the relatively low accuracy of our VDR prediction model: 61.3% by the labelled dependency attachment accuracy measure. One possible explanation could be that not all parts of the VDR structure are useful for retrieval purposes, and our VDR prediction model does well on the useful ones. This observation also suggests that we are unlikely to achieve better retrieval performance by continuing to improve the accuracy of VDR prediction. We believe a more promising direction is refining the current formulation of the VDR, and exploring more sophisticated ways to measure the similarity of two structured representations.

7 Conclusion

In this paper we argued that a limiting factor of retrieving images depicting actions is the unstructured bag-of-terms representation typically used for images. In a bag-of-terms representation, images that share similar sets of regions are deemed to be related even when the depicted actions are different. We proposed that representing an image using the Visual Dependency Representation (VDR) can prevent this type of misclassification in image retrieval. The VDR of an image captures the region–region relationships that explain what is happening in an image, and it can be automatically predicted from a region-annotated image.

In a query-by-example image retrieval task, we found that representing images as automatically predicted VDRs resulted in statistically significant 8.8% relative improvement in MAP and 8.6% relative improvement in Precision@10 compared to a Bag-of-Terms model. There was a significant difference in MAP when using manually or automatically predicted image structures, but no difference in the Precision@10, suggesting that the proposed automatic prediction model is accurate enough for retrieval purposes. Future work will focus on using automatically generated visual input, such as the output of the image tagger (Guillaumin and Mensink, 2009), or an automatic object detector (Felzenszwalb et al., 2010), which will make it possible to tackle image ranking tasks (Hodosh et al., 2013). It would also be interesting to explore alternative structure prediction methods, such as predicting the relationships using a conditional random field (Zitnick et al., 2013), or by leveraging distributional lexical semantics (Le et al., 2013b).

Acknowledgments

The anonymous reviewers provided valuable feedback on this paper. The research is funded by ERC Starting Grant SYNPROC No. 203427.

References

Moshe Bar and Shimon Ullman. 1996. Spatial Context in Recognition. *Perception*, 25(3):343–52, January.

I Biederman. 1972. Perceiving real-world scenes. *Science*, 177(4043):77–80.

Koby Crammer and Yoram Singer. 2002. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292.

Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. 2008. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):1–60.

P Duygulu, Kobus Barnard, J F G de Freitas, and David A Forsyth. 2002. Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary. In *Proceedings of the 7th European Conference on Computer Vision*, pages 97–112, Copenhagen, Denmark.

Desmond Elliott and Frank Keller. 2013. Image Description using Visual Dependency Representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, Seattle, Washington, U.S.A.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. 2011. The PASCAL Visual Object Classes Challenge 2011.

Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: generating sentences from images. In *Proceedings of the 15th European Conference on Computer Vision*, pages 15–29, Heraklion, Crete, Greece.

P F Felzenszwalb, R B Girshick, D McAllester, and D Ramanan. 2010. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645.

Matthieu Guillaumin and Thomas Mensink. 2009. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *IEEE 12th International Conference on Computer Vision*, pages 309–316, Kyoto, Japan.

Micah Hodosh, Peter Young, and Julia Hockenmaier. 2013. Framing Image Description as a Ranking Task: Data, Models and Evaluation Metrics. *Journal of Artificial Intelligence Research*, 47:853–899.

Victor Lavrenko, R Manmatha, and Jiwoon Jeon. 2003. A Model for Learning the Semantics of Pictures. In *Advances in Neural Information Processing Systems 16*, Vancouver and Whistler, British Columbia, Canada.

S. Lazebnik, C. Schmid, and J. Ponce. 2006. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2169–2178, New York, NY, USA.

DT Le, R Bernardi, and Jasper Uijlings. 2013a. Exploiting language models to recognize unseen actions. In *Proceedings of the International Conference on Multimedia Retrieval*, pages 231–238, Dallas, Texas, U.S.A.

DT Le, Jasper Uijlings, and Raffaella Bernardi. 2013b. Exploiting language models for visual recognition. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 769–779, Seattle, Washington, U.S.A.

D G Lowe. 1999. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, pages 1150–1157, Washington, D.C., USA.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada.

Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. 2008. LabelMe: A Database and Web-Based Tool for Image Annotation. *International Journal of Computer Vision*, 77(1-3):157–173.

Mohammad A Sadeghi and Ali Farhadi. 2011. Recognition Using Visual Phrases. In *2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1745–1752, Colorado Springs, Colorado, U.S.A.

Jianbo Shi and Jitendra Malik. 2000. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, August.

Lei Wu, Rong Jin, and Anil K Jain. 2012. Tag Completion for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):716–727.

Yezhou Yang, Ching Lik Teo, Hal Daumé III, and Yiannis Aloimonos. 2011. Corpus-Guided Sentence Generation of Natural Images. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 444–454, Edinburgh, Scotland, UK.

CL Zitnick, Devi Parikh, and Lucy Vanderwende. 2013. Learning the Visual Interpretation of Sentences. In *IEEE International Conference on Computer Vision*, pages 1681–1688, Sydney, Australia.

Augmenting Business Entities with Salient Terms from Twitter

Riham Mansour
Microsoft Research ATL
Cairo, Egypt
rihamma@microsoft.com

Nesma Refaei
Cairo University
Cairo, Egypt
nesma.a.refaei@eng.cu.edu.eg

Vanessa Murdock
Microsoft
Seattle, WA
vanessa.murdock@yahoo.com

Abstract

A significant portion of search engine queries mention business entities such as restaurants, cinemas, banks, and other places of interest. These queries are commonly known as "local search" queries, because they represent an information need about a place, often a place local to the user. A portion of these queries is not well served by the search engine because there is a mismatch between the query terms, and the terms representing the local business entity in the index. Business entities are frequently represented by their name, the category of entity (whether it is a restaurant, an airport, a grocery store, etc.) and other meta-data such as opening hours and price ranges. In this paper, we propose a method for representing business entities with a term distribution generated from web data and from social media that more closely aligns with user search query terms. We evaluate our system with the local search task of ranking businesses given a query, in both the U.S. and in Brazil. We show that augmenting entities with salient terms from social media and the Web improves precision at rank one for the U.S. by 18%, and for Brazil by 9% over a competitive baseline. For precision at rank three, the improvement for the U.S. is 19%, and for Brazil 15%.

1 Introduction

Search engine queries, particularly queries issued from mobile devices, often mention business entities such as restaurants, cinemas, banks, and other places of interest. These "local search" queries represent an information need about a place. Often there is a mismatch between the query terms, and the terms representing the local business entity in the index, making it difficult for the search engine to find results that satisfy the user. Local data consists largely of listings of businesses, annotated with metadata. This metadata includes the name of the location, category information (is the business a clothing retailer, or a Thai restaurant, for example), address and phone number, opening hours, and indicators such as price range, popularity, star ratings, etc. Figure 1 shows an example of the type of information available to local search systems.

Some local search queries are known item searches, where the user knows the name of a business and they seek other information about the place, such as the opening hours. Other local search queries are category searches where the user does not know the name of a specific business but is using the Internet in much the same way they might have used the Yellow Pages in pre-Internet days. An example of a category search is "Thai restaurants in Denver". There are also descriptive local queries such as "pizza delivery" or "romantic brunch in Seattle" where the user does not mention a category or a business name directly, but for which there is a closed class of businesses that will satisfy the user's need.

Descriptive queries such as "roasted chiles in Santa Fe" or "kid-friendly Caribbean resorts" pose a significant challenge to local search systems, as the information in the local index does not typically include terms that match the user's query. That is, the system may know businesses in Santa Fe, but not whether they sell roasted chiles.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 121–129, Dublin, Ireland, August 23-29 2014.

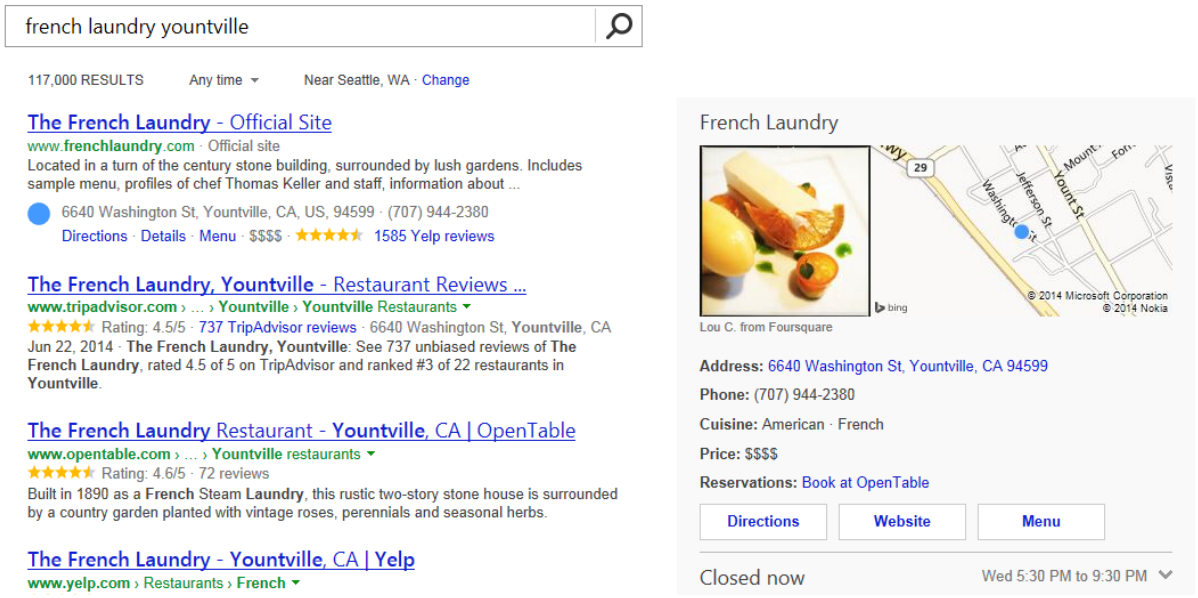


Figure 1: Example of the type of meta-data associated with a business entity, in this case a restaurant.

However, collectively people themselves know this type of information, and they frequently mention it in social media. The discussion of a local business in social media, such as Twitter[1], Flickr[2], Facebook[3] and Foursquare[4] may take the form of a simple check-in ("Drinking a Smog Rocket at @byronhamburgers") or a Facebook status caption to a photo ("Sea stars at the Seattle Aquarium"), or a Tweet ("the quad & the blonde both were good! The choc flavored one wasn't so much to my tastes..."), among others.

A growing number of users of social media attach geographic coordinates to their status updates, allowing the text of the updates to be associated to a location. Further, businesses use social networks as a publicity platform to widen their customer base. Today, Twitter has more than 500 million users.[5]

In this paper we augment business entities with salient terms describing the business. We extract the terms from Twitter, and from the Web. To determine which terms are salient, we compute the co-occurrence of terms with mentions of the business name (and name variants), for tweets issued within one kilometer of the business. Because some users are especially prolific on social media, and may dominate the tweets issued in that location, we estimate the term co-occurrence statistics with the user frequency of a term: the number of people using that term in a given location. We also extract salient terms from the Web pages of the business entity, but in this case the user frequency is not meaningful, so term co-occurrence is calculated with the term frequency.

We evaluate the term distributions describing a place in the context of local search for the U.S. and Brazil. We construct a corpus of search engine queries with local intent, and evaluate the retrieval of businesses in response to the queries. We compare several different strategies for augmenting the representation of the business to a baseline system described in Colombo et al. (2013). Augmenting with tweets improves precision at rank one for U.S. local search by 18%, and for Brazil by 9%. For precision at rank three, the improvement for the U.S. is 19%, and for Brazil 15%.

The rest of the paper is organized as follows: Section 2 surveys the related work. Section 3 details how salient terms are extracted from tweets and the Web. Section 4 illustrates the experimental setting and the evaluation of the impact of salient terms on retrieval. Section 5 presents a discussion of the results and Section 6 concludes the paper with remarks for future work.

[1] `www.twitter.com` visited March 2014
[2] `www.flickr.com` visited March 2014
[3] `www.facebook.com` visited March 2014
[4] `www.foursquare.com` visited March 2014
[5] `http://www.statisticbrain.com/twitter-statistics/` visited March 2014

2 Related Work

Modeling business entities from multiple sources like the Web and social media remains an open problem. Most of the work in this domain focuses on modeling locations and regions more generally (O'Hare and Murdock, 2013; Laere et al., 2012; Bennett et al., 2011), or on extracting mentions of business entities from text using NLP techniques (Rae et al., 2012). O'Hare and Murdock (2013) propose a statistical language modeling approach to characterize locations in text, based on user frequency. They utilize the geo-tagged public photos in Flickr. The primary difference between their work and ours is that they estimate the user frequency distribution, whereas we employ the user frequency in calculating the term co-occurrence. Also, the locations described in O'Hare and Murdock represent locations of one kilometer distance. They do not attempt to characterize specific points of interest or businesses.

There has been significant effort to leverage image content to characterize locations, due to the availability of geotagged Flickr photos. Much of the work uses Flickr photos and tag sets, and focuses on identifying the locations in photos. This is related, although not directly applicable, to work with Twitter. Ahern et al. (2007) identify geographically related tags by finding dense areas using geodesic distances between images. They rank the tags in these areas with $tf.idf$. In their subsequent work Kennedy et al. (2007) leverage tags that represent local events. Naaman et al. (2003) and Moxley et al. (2008) propose approaches for recommending tags to the user given a known location for an image. Some research efforts leverage image content to characterize locations. Crandall et al. (2009) employs image content and textual metadata to predict the location of a photograph at the city level and at the individual landmark level. Hays and Efros (2008) use visual features to predict geographic locations by nearest-neighbor classification.

Colombo et al. (2013) provide the baseline system for this paper, and it is described in more detail in Section 4.1. They use online reviews, comments and user tips about points of interest in location-based services like Yelp, Google+, and Qype to build a tag-based representation of a point of interest. They rank the tags by their $tf.idf$ score from a collection of location-based service related documents.

In terms of using geo-referenced information to represent locations, Rodrigues (2010) proposes to extract points of interest automatically from the Web, for example from Yahoo, Manta and Yellow Pages. He also infers points of interest based on geo-referenced content such as geo-tagged photos, blog posts and news feeds. They cluster content from multiple sources while building a language model for each cluster. Tags in each cluster are scored by $tf.idf$. This work is similar in spirit to the work proposed in this paper, although our work focuses more on obtaining the most unique and frequent tags associated with points of interest in tweets.

Hegde et al. (2013) assign tags to points of interest based on user interest profiles in online social networks and check-in logs of users at these places. They use probabilistic modeling to derive the point of interest tags followed by hierarchical clustering of most probable tags to filter out semantically irrelevant tags. Biancalana et al. (2013) use point of interest-related location-based service content to extract key phrases that could serve as tags characterizing each point of interest. The extracted phrases are weighted by user authority.

In terms of modeling locations from short microblog messages like tweets. Paradesi (2011) proposes TwitterTagger, a system that geo-tags tweets and shows them to users based on their current physical location. The tweets are geo-tagged by identifying the locations referenced in a tweet by part of speech tagging and a database of locations. Eisenstein et al. (2010) and Kinsella et al. (2011) present methods to identify the location of a user based on his or her tweets. Li et al. (2011) rank a set of candidate points of interest using language and temporal models. Given a query tweet, they build a unigram language model for each candidate point of interest and for the query tweet. Points of interest are then ranked by their KL-divergences with the tweet language model. Unlike our work, both approaches identify a location in tweets rather than modeling a certain location by the way it is mentioned in tweets.

3 Describing Businesses with Twitter and the Web

Salient terms are terms that uniquely characterize a place. As an overview, we extract terms from two sources namely geo-tagged tweets and business-related webpages. We extract terms from geo-tagged

tweets posted from locations within one kilometer of the business. We then identify the tweets about a given business from among the nearby tweets, by looking for mentions of the business name (along with naming variants). We compute the term co-occurrence between the business name, and the terms that occur in tweets mentioning the business.

We also extract terms from webpages related to the business entity. We issue a query with the business name to the Bing Search API.[6] We compute the term co-occurrence between the business name, and the terms that occur in these top three web pages resulting from Bing search.

There is no universal standard for representing locations. Some gazetteers are available for developers that represent places according to a hierarchy (such as Geonames[7] and Placemaker[8]). There is also proprietary data gathered by companies such as Nokia, YellowPages and Yelp, which provide some information about places like geo-location, address, and phone number. There are also open source data like Freebase and DBpedia. Both proprietary and open source data use structured representations for places.

There are three challenges with these representations. First, they do not provide a rich description of the place, as they are primarily designed to help users locate the place, via the name, address and phone number, or category ("restaurant" or "cinema," for example). However, the categories may be broad and in a language different from the language spoken by the user. Second, the coverage of points of interest and businesses focuses mostly on well-known places. Businesses are not usually well-represented because they are often relatively ephemeral. Finally, the data may be stale. For example, a restaurant that has closed, or moves location, should be flagged, and it may take time for the gazetteer to be updated. Social media provides fresh information about businesses, especially as more businesses promote themselves via these channels. Modeling businesses with tweets could complement the available data with fresh descriptions.

3.1 Text Pre-processing

We acquire geo-tagged tweets related to business entities in the United States and Brazil from the Twitter firehose, from January 1, 2013 to May 31, 2013. We chose these countries because of their high usage of Twitter, and to show that the approach is language agnostic. The tweets are primarily in English (in the U.S.) and in Portuguese (in Brazil).

We pre-process the tweets by removing stop words, using the Natural Language Toolkit (NLTK) library[9] and non-alphabetic characters. For our baseline implementation following Colombo (2013), we remove the non-English words using the English NLTK wordnet corpus. For removing non-Portuguese words in the baseline, we use the Enchant specll checking library.[10] In our proposed approach, we don't remove non-English and non-Portuguese words, but we rather remove twitter terms that did not appear in the Bing query logs in December 2013. Further, we remove tweets automatically generated by check-in services such as Foursquare by detecting the patterns "I'm at" and "mayor". We remove shortened URLs in the tweet text by detecting the pattern "http://t.co." URLs are removed as they do not carry salient terms. All text was lower-cased. All tweets are indexed in Solr,[11] an open-source search engine which allows for field search. The index carries the tweet text, geographic coordinates, time stamp, language, country, retweet count, source, URL and user information.

3.2 Computing Salient Terms

The business entities were submitted to the Solr index as queries, to retrieve the tweets related to the entity itself. We apply two sequential filters on the indexed tweets to obtain the relevant tweets. The first filter limits the search to those tweets whose geographic coordinates are within one kilometer of the business entity. This covers a wide range around the POI due to the small volume of geo-tagged tweets

[6] http://datamarket.azure.com/dataset/bing/search visited March 2014
[7] http://www.geonames.org visited March 2014
[8] http://developer.yahoo.com/boss/geo/ visited March 2014
[9] http://nltk.org/ visited March 2014
[10] http://pythonhosted.org/pyenchant/ visited March 2014
[11] http://lucene.apache.org/solr/ visited March 2014

in general. Enlarging the range to one kilometer retains a reasonable volume although it does introduce more irrelevant tweets. The second filter eliminates irrelevant tweets by searching with the canonical name of the business along with naming variants. The indexed tweets are searched by name, 70% of the name, and the name fully concatenated with no spaces separating the multiple words, and with spaces replaced with an underscore. The resulting set of tweets are those that are relevant to the business entity since they have been posted within its vicinity and they mention the entity directly.

To extract the salient terms from Twitter, we compute the term co-occurrence of the entity name with the set of terms co-occurring in the associated tweets. Term co-occurrence is traditionally computed as the number of times term t and term w appear in the same tweet C, divided by the number of times term t appears in any tweet in the same one-kilometer vicinity, plus the number of times term w appears in any tweet in the same one-kilometer vicinity:

$$score(t, w) = \frac{count_C(t, w)}{count_C(t) + count_C(w)}. \quad (1)$$

Some users of twitter are extremely prolific, and may generate a lot of data in a small set of places. Term frequency may produce an estimate of the term distribution biased toward a particular user or set of users. To prevent a single prolific user from dominating the representation of a place, we estimate the term co-occurrence with the user frequency. That is, the term counts are the number of people who used a term in a place, rather than the number of times a term was applied. This has been shown to be a more reliable estimate of term distributions in other work using social media to model places (O'Hare and Murdock, 2013). Note that the baseline implementation is based on the term frequency, and uses tf.idf rather than term co-occurrence.

We also enrich the business entities with terms from the web pages. We issue a query to Bing Search API with the business name. We then extract salient terms from the content of the top three results. We pre-process the text according to Section 3.1 to get the unigram terms. We filter out the terms that are substrings of the business name, and single character terms. The terms are weighted according to the term frequency (tf) and the terms with $tf > 0.001$ are considered salient to the business entity. This threshold has been selected empirically.

4 Experimental Setting

In our experiments we evaluated the effect of expanding the business entities with salient terms within the context of local search. We examined whether adding tags such as "conchiglie" to the entity "French Laundry" will improve the retrieval results for a query with local intent like "conchiglie Napa Valley". For this purpose, we sampled a set of 30,000 businesses from a proprietary database of business listings in the United States and Brazil. We then chose 80 entities from the two countries to formulate the test set of search queries as illustrated below.

4.1 Baseline Approach

Colombo et al. (2013) suggested a method for filtering the salient terms extracted from a set of documents relevant to a place of interest. We used their method to filter the salient terms extracted from the geo-tagged tweets selected and pre-processed as described above in Section 3.1. The terms remaining after these filtration steps are weighted using $tf.idf$, where a background corpus of all tweets relating to any business within one kilometer of the entity in question is used to calculate the idf of each term. Finally, we kept only the terms with a $tf.idf$ greater than a threshold of 0.04 as the baseline salient terms for the business.

4.2 Building the Search Corpus

Our database of businesses contains metadata about each business including the name, phone number, website, street address, city, country, geographic coordinates, and category information that are a subset of a taxonomy of categories both in English and in the language of the country of the business. We appended the extracted salient terms for each business as a field in our database. We removed twitter

terms that escaped initial filtering by removing any terms that did not appear in the Bing query logs in December 2013. We also filtered out twitter terms that are included in the category taxonomy, as these tags will not add value to the existing data, and are unlikely to improve retrieval over the naive baseline.

Some businesses are very popular, and are likely to generate more social media traffic. To make sure that the system is as general as possible, and that we don't build in an inherent bias toward popular businesses (or national chains) we construct the search corpus to represent varying popularity levels. The popularity of a business is quantified by the number of unique users tweeting about it. We stratify the selection of the businesses from our database of 30000 businesses such that the search corpus contains 15,000 businesses from the U.S. and 15,000 businesses in Brazil, which are distributed across a range of popularity scores. Finally we indexed the search corpus using Solr.

4.3 Generating Search Queries

We formulate search queries by selecting 40 businesses in each market with their attributes and salient terms. We formulated query templates from the business name, location, category and terms selected by three judges from associated tweets and Web pages. The information is detailed in Table 1. The query templates are shown in Table 2, along with an illustrative example of each one.

Attribute	**Description**
Name	business name and variants
Location	city and country
Categories	categories provided by the database
Terms	term selected by judges from Twitter and Web pages

Table 1: Information included in the baseline queries

Query Template	**Example**
Name	"French Laundry"
Name + Location	"French Laundry in Yountville" or "French Laundry in California"
Name + Category	"French Laundry Restaurant"
Name + Term	"conchiglie French Laundry"
Term + location	"conchiglie Yountville" or "conchiglie California"
Category + location	"Restaurants in Yountville" or "Restaurants in California"

Table 2: Query templates with examples

Some of the automatically generated queries (such as "happy in california" and "week in Houston") don't have a local intent because of uninformative terms (such as "good", "happy", or "week") or because of malformed substrings of names and categories. To filter out these uninformative queries we issued the query to Bing Search API and kept only the queries that generated a direct answer. An example of a direct answer is shown in Figure 1. The Bing Search API returns a direct answer when the query has been classified as having local intent. We use the Bing API in this way as a black box, because building a local intent classifier is a significant undertaking, and is beyond the scope of this paper. The resulting test set consists of 1000 local queries representing 80 business entities in Brazil and the U.S., with an equal distribution of each of the query templates in Table 2.

4.4 Evaluation

Our primary evaluation is of query expansion for the class of queries for which a business listing is a relevant result. However, representing a business entity with a term distribution estimated from social

media has other applications as well. For this reason, we would like to know the quality of the expansion terms, independent of any task. To this end, we asked three judges to pick all the relevant terms from among an unordered set of extracted terms salient to a business, for 100 businesses in each country. We divided the terms among the three judges equally and each term has been judged by only one judge. The number of tags extracted from the web pages is an order of magnitude larger than the number of tags extracted from Twitter for a given business. We consider the tag accuracy to be proportion of "good" tags accounted for by a single data source. That is, for Twitter, it is the number of "good" Twitter tags, divided by the total number of "good" tags, whereas the accuracy of the Web tags is the number of "good" tags derived from the web, divided by the total number of "good" tags. Based on this assessment, the accuracy of the Twitter tags for the U.S. data was 0.22, and the accuracy of the Web tags was 0.78. For the data from Brazil, the accuracy of the Twitter terms was 0.15, and the accuracy of terms derived from the Web was 0.85.

The effect of the expansion strategies on the retrieval of business entities. As Solr allows for field search, we can limit the fields to the entity and its metadata, or the entity metadata and the twitter tags, etc. Tables 3 and 4 show the results for various retrieval from fields representing document expansion strategies on data from the U.S. and Brazil, respectively. The results are averaged over 500 queries (from the query formulations described above) for each country. In Tables 3 and 4 we see that nearly 60% of queries return the correct result at rank one, when the entity is represented only by its metadata. The results reported in the other rows also include the entity metadata. (The baseline in Tables 3 and 4 is described in Section 4.1.) Expanding the represnetation of the point of interest with terms from the Web and from social media shows a clear benefit.

Mobile devices are becoming ubiquitous, and local search represents an important class of search on mobile devices. Because the devices are small, real estate to show results is extremely limited. For this reason, we choose to evaluate precision @ k, for $k <= 3$ for this task. To create a truth set, the top three results were evaluated by judges to determine their relevance to the query. Each result is judged by one assessor. Because precision at one is binary, we do not apply a statistical significance test. Percent change is reported for precision at rank one, with respect to the baseline (row two). The fact that the precision at rank three is lower than precision at rank one is an artifact of their being a single relevant result in most cases.

	P@1	**P@3**	**% Change in P@1**
Entity metadata	0.595	0.353	(oracle)
Baseline	0.627	0.358	NA
Entity metadata + twitter tags	0.667	0.389	+6.4%
Entity metadata + web terms	0.686	0.396	+9.4%
Entity metadata + web terms + twitter tags	0.738	0.425	+18%

Table 3: Precision @ k for local search in the U.S.

	P@1	**P@3**	**% Change in P@1**
Entity metadata	0.618	0.436	(oracle)
Baseline	0.643	0.460	NA
Entity metadata + twitter tags	0.650	0.474	+1%
Entity metadata + web terms	0.700	0.517	+8.9%
Entity metadata + web terms + twitter tags	0.708	0.533	+10%

Table 4: Precision @ k for local search in the Brazil.

5 Discussion

Since the set of queries consists of the entity name plus attributes from the index such as the location and the category information, the resulting precision from search just on the entity metadata itself shows the degree to which the bias in the data accounts for the results. That is, if you have the correct entity name, location and category, just searching for a business with matching metadata gives a precision at rank one of 0.595 (0.618 for Brazil). This is a naive baseline. The baseline results show that it is a competitive baseline because it demonstrates that there is a benefit to expand the representation of a business entity with text, beyond the naive baseline above it in the table.

The gains in precision suggest that the extracted salient terms with co-occurrence statistics and user frequency from twitter and the web pages are of better quality than the terms extracted by the baseline in Colombo et al. (2013) with term frequency only. This is attributed to the fact that co-occurrence statistics and user frequency capture the terms that people frequently use when describing a place. Further, the quality of the salient terms extracted from the web pages exceeds the quality of the twitter terms. This is to be expected if the main search results for a business entity are reasonable, and the top three results are relevant to the query. Social media is notoriously noisy, so it is not surprising that the web pages produce more reliable expansion terms. Furthermore, comparing the terms expanded from the web, to the terms expanded from Twitter, we see the relative improvement with respect to the baseline of the Web expansion terms is greater than the Twitter expansion terms. The fact that both expanding from twitter and the Web produces results better than either individually shows that the two term distributions cover different slices of the vocabulary.

We experimented with the number of tweets required to improve the representation of the point of interest. We focused on the portion of the test set with queries of the form *term + location* like "conchiglie Yountville," as those are the queries that are not answered with relevant results in the absence of the proper salient terms. We found that 10 to 30 tweets mentioning the business were sufficient to improve the retrieval results for these queries, and there was no benefit to increasing the number of tweets to 50 or 100. In the Brazil data, the results for four of the queries of the form *term + location* were degraded when sampling terms from 10 tweets compared to more. However, the results were the same for 30, 50, 100 or more tweets, suggesting that there is no benefit to increasing the number of tweets beyond 30. This suggests that a smaller number of tweets is better, in terms of extracting salient terms. One possible reason for this is that adding more tweets increases the number of noise terms, relative to the number of salient terms.

6 Conclusion and Future Work

In this paper, we present an effective representation of business entities with a term distribution generated from web data and from social media that more closely aligns with user search query terms. We evaluate our system with the local search task of ranking businesses given a query, in both the U.S. and in Brazil. Our method uses co-occurrence statistics and user frequency to extract relevant salient terms. The results demonstrate the effectiveness of this approach when compared with a competitive baseline that uses term frequency to extract salient terms. Furthermore, we show that query expansion with salient terms improves retrieval in the common task of retrieving a business listing in response to a user query.

We leave to future work applying query expansion from social media to larger collections of local search queries, and other methods for formulating query templates based on the metadata available with business listings.

References

Shane Ahern, Mor Naaman, Rahul Nair, and Jeannie Yang. 2007. World Explorer: Visualizing aggregate data from unstructured text in geo-referenced collections. In *JCDL '07*.

Paul N. Bennett, Filip Radlinski, Ryen W. White, and Emine Yilmaz. 2011. Inferring and using location metadata to personalize web search. In *Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011)*.

C. Biancalana, F. Gasparetti, A. Micarelli, and G. Sansonetti. 2013. An approach to social recommendation for context-aware mobile services. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(1).

G. B. Colombo, M. J. Chorley, V. Tanasescu, S. M. Allen, C. B. Jones, and R. M. Whitaker. 2013. Will you like this place? a tag-based place representation approach. In *International Workshop on the Impact of Human Mobility in Pervasive Systems and Applications*.

D.J. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg. 2009. Mapping the world's photos. In *Proceedings of the 18th International Conference on World Wide Web*, pages 761–770. ACM.

Jacob Eisenstein, Brendan O'Connor, Noah Smith, and Eric P. Xing. 2010. A latent variable model for geographic lexical variation. In *Proceedings of EMNLP*.

James Hays and Alexei A. Efros. 2008. IM2GPS: estimating geographic information from a single image. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Vinod Hegde, Josiane Xavier Parreira, and Manfred Hauswirth. 2013. Semantic tagging of places based on user interest profiles from online social networks. In *Advances in Information Retrieval: Lecture Notes in Computer Science*, volume 7814, pages 218–229. Springer.

Lyndon Kennedy, Mor Naaman, Share Ahern, Rahul Nair, and Tye Rattenbury. 2007. How flickr helps us make sense of the world: Context and content in community-contributed media collections. In *Proceedings of the 15th International Conference on Multimedia*, pages 631–640.

Sheila Kinsella, Vanessa Murdock, and Neil O'Hare. 2011. "I'm Eating a Sandwich in Glasgow": Modeling Locations with Tweets. In *Proceedings of the 3rd International Workshop on Search and Mining User-Generated Content*, pages 61–68.

Olivier Van Laere, Steven Schockaert, and Barth Dhoedt. 2012. Georeferencing flickr photos using language models at different levels of granularity: An evidence based approach. *Journal of Web Semantics*, 16.

Wen Li, Pavel Serdyukov, Arjen de Vries, Carsten Eickhoff, and Martha Larson. 2011. The where in the tweet. In *Proceedings of the 20th ACM international conference on Information and knowledge management (CIKM)*.

Emily Moxley, Jim Kleban, and B.S. Manjunath. 2008. Spritittagger: A geo-aware tag suggestion tool minded from flickr. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval (MIR'08)*, pages 24–30.

Mor Naaman, Andreas Paepcke, and Hector Garcia-Molina. 2003. From where to what: metadata sharing for digital photographs with geographic coordinates. *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, pages 196–217.

Neil O'Hare and Vanessa Murdock. 2013. Modeling locations with social media. *Journal of Information Retrieval*, 16(1).

Sharon Paradesi. 2011. Geotagging tweets using their content. In *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference*.

Adam Rae, Vanessa Murdock, Adrian Popescu, and Hugues Bouchard. 2012. Mining the web for points of interest. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Filipe Rodrigues. 2010. *POI Mining and Generation*. Ph.D. thesis, University of Coimbra.

A PAC-Bayesian Approach to Minimum Perplexity Language Modeling

Sujeeth Bharadwaj
University of Illinois
405 N. Mathews Ave.
Urbana, IL 61801, USA
sbhara3@illinois.edu

Mark Hasegawa-Johnson
University of Illinois
405 N. Mathews Ave.
Urbana, IL 61801, USA
jhasegaw@illinois.edu

Abstract

Despite the overwhelming use of statistical language models in speech recognition, machine translation, and several other domains, few high probability guarantees exist on their generalization error. In this paper, we bound the test set perplexity of two popular language models – the n-gram model and class-based n-grams – using PAC-Bayesian theorems for unsupervised learning. We extend the bound to sequence clustering, wherein classes represent longer context such as phrases. The new bound is dominated by the maximum number of sequences represented by each cluster, which is polynomial in the vocabulary size. We show that we can still encourage small sample generalization by sparsifying the cluster assignment probabilities. We incorporate our bound into an efficient HMM-based sequence clustering algorithm and validate the theory with empirical results on the resource management corpus.

1 Introduction

The ability to predict unseen events from a few training examples is the holy grail of statistical language modeling (SLM). Although the final test for any language model is its contribution to the performance of a real system, task-independent metrics such as perplexity are popular for evaluating the general quality of a model. Standard algorithms therefore attempt to minimize perplexity on some previously unobserved test set, assumed to be drawn from the same distribution as the training set. This begets the question of how the test set perplexity is related to training set perplexity – every paper on SLM has an answer, with varying levels of theoretical and empirical justification.

The problem of data sparsity and generalization can be traced back to at least as early as Good (1953), and possibly Laplace, who recognizes that the maximum likelihood (ML) estimate of event frequencies (n-grams) cannot handle unseen events. Smoothing techniques such as the add-one estimator (Lidstone, 1920) and the Good-Turing estimator (Good, 1953) assign a non-zero probability to events that have never been observed in the training set. Recently, Ohannessian and Dahleh (2012) strengthened the theory by showing that Good-Turing estimation is consistent when the data generating process is heavy-tailed. In the context of this paper, smoothing was perhaps the first attempt to bound generalization error, in that it successfully guarantees a finite test set perplexity.

It is evident that smoothing of the n-gram estimate alone is not sufficient. Techniques that incorporate lower and higher order n-grams, such as Katz (1987) smoothing, Jelinek-Mercer (1980) interpolation, and Kneser-Ney (1995) smoothing, have become standard (Rosenfeld, 2000). Chen and Goodman (1999) provide a thorough empirical comparison of smoothing methods and uncover useful relationships between the test set cross-entropy (log perplexity) and the size of the training set, model order, etc. A Bayesian interpretation further explains why some of the techniques (don't) work. Teh (2006) discusses fundamental limitations of the Dirichlet process (Mackay and Peto, 1995) and proposes the hierarchical Pitman-Yor language model as a better way of generating the heavy-tailed (power law) distributions exhibited in natural language.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 130–140, Dublin, Ireland, August 23-29 2014.

Instead of directly modeling a heavy-tailed distribution over words, class-based models address data sparsity by estimating n-grams over clusters of words. Intuitively, clustering is a transformation of the event space from the space of word n-grams, in which most events are rare, to the space of class n-grams, which is more densely measured and therefore requires fewer training examples. Brown et al. (1992) show that the clustering function that maximizes the training data likelihood must also maximize mutual information between adjacent clusters; although several useful clustering algorithms are based on this principle, no provable guarantees currently exist. Moreover, word transitions are never completely captured by the underlying class transitions, and some tradeoff between accurate estimation of frequent events (word n-grams) and generalization to unseen events (class n-grams) is desired – class-based models are therefore often interpolated with word n-grams using some of the previously described Bayesian methods (Rosenfeld, 2000).

Our survey of SLM techniques and their treatment of generalization error has been rather brief and certainly not comprehensive. We focus primarily on n-grams and related models since they have dominated SLM over the last several decades (Rosenfeld, 2000), and therefore serve as a good starting point for further analysis. The existing literature suggests that apart from empirical validation and intuition, no provable guarantees exist on the generalization error of language models. Bayesian techniques work well only to the extent the prior assumptions are valid; in this paper, we present theoretical guarantees that hold irrespective of the correctness of the prior.

Model selection approaches such as the Akaike Information Criterion (AIC) (Akaike, 1973) and its variants (Burnham and Anderson, 2002) quantify the tradeoff between complexity and goodness of fit. In the context of a language model, it can be shown that test set cross entropy is approximately the training set cross entropy plus the number of model parameters. Unfortunately, such bounds are loose and do not provide significant algorithmic insight – at best, they recommend the smallest model that works well on the training set. Chen (2009) obtained a very accurate relationship for exponential language models by estimating the test set performance with linear regression. Although empirical, his approximation leads to better models based on $l_1 + l_2^2$ regularization. Exponential models are often motivated with the minimum discrimination information (MDI) principle, which roughly states that of all distributions satisfying a particular set of features, the exponential family is the centroid (minimizes distortion relative to the farthest possible true distribution) (Rosenfeld, 1996). This does not bound the generalization error in the manner we wish to, but it is nevertheless a useful property that complements Chen's observations.

In this paper, we strive for the best of both worlds – we present PAC-Bayesian theory as a powerful tool for deriving high probability guarantees as well as efficient and well-motivated algorithms. In the next section, we state some useful PAC-Bayesian theorems. In Section 3, we present our main results. We apply the PAC-Bayesian bounds to n-grams, class-based n-grams, and also sequence clustering, where classes represent longer context such as phrases. We show that for sequence clustering, the bound is dominated by the maximum number of sequences represented by each cluster, and consequently requires many more training examples than a class-based model over words. We address this issue by sparsifying the cluster assignment probabilities using the l_α norm, $0 < \alpha < 1$, an effective proxy for the intractable l_0 norm. In Section 4, we show how our bound can be incorporated into an HMM-based clustering algorithm. In Section 5, we validate the theory presented in this paper with some empirical results on the resource management corpus.

2 PAC-Bayesian Bounds

PAC-Bayesian theory is a useful framework for combining frequentist bounds with the notion of a prior. Probably approximately correct (PAC) learning bounds the worst case generalization error of the best hypothesis selected from a hypothesis space – and therefore treats all hypotheses uniformly (Valiant, 1984). PAC-Bayesian bounds, however, place a prior over the hypothesis space while making no assumptions on the data generating distribution (McAllester, 1998). Thus, PAC-Bayesian bounds can both 1) incorporate prior information, and 2) provide frequentist guarantees on the expected performance. They have been successfully applied to classification settings such as the support vector machine (SVM) (McAllester, 2003; Langford, 2005), yielding significantly tighter bounds. Seldin and Tishby (2010) extend the frame-

work to include unsupervised learning tasks such as density estimation and clustering. Since statistical language modeling at its core is a discrete density estimation problem, we focus on the bounds developed by Seldin and Tishby (2010) and summarize key results in the following subsection.

2.1 Unsupervised Learning

Given a d-dimensional product space $\mathcal{X}^{(1)} \times ... \times \mathcal{X}^{(d)}$ and a collection of N samples, S, independent and identically distributed (i.i.d.) according to some unknown distribution $p(x_1, ..., x_d)$ over the product space, we want to estimate $p(x_1, ..., x_d)$ with some model $q(x_1, ..., x_d)$. In the case of clustering (e.g. class-based models), we make the following assumption on $q(x_1, ..., x_d)$ [Note: we make no assumptions on the true distribution $p(x_1, ..., x_d)$]:

$$q(x_1, ..., x_d) = \sum_{c_1,...,c_d} q(c_1, ..., c_d) \prod_{i=1}^{d} q(x_i|c_i) \tag{1}$$

where $c_i = h_i(x_i)$ for some clustering function $h_i : \mathcal{X}^{(i)} \mapsto \mathcal{C}^{(i)}$. We refer to them collectively as a clustering function h, $h = \{h_i\}_{i=1}^{d}$; hence $h : \mathcal{X}^{(1)} \times ... \times \mathcal{X}^{(d)} \mapsto \mathcal{C}^{(1)} \times ... \times \mathcal{C}^{(d)}$. We assume that the original space $\mathcal{X}^{(1)} \times ... \times \mathcal{X}^{(d)}$ has finite cardinality, with $n_i = |\mathcal{X}^{(i)}|$, and likewise for the clustered space $\mathcal{C}^{(1)} \times ... \times \mathcal{C}^{(d)}$, where $m_i = |\mathcal{C}^{(i)}|$ is the number of clusters. We define a hypothesis space, $\mathcal{H}$, to be the space of all possible clustering functions $h \in \mathcal{H}$.

For $h \in \mathcal{H}$, we define the distributions $p_h(c_1, ..., c_d) = \sum_{x_1,...,x_d} p(x_1, ..., x_d) \prod_{i=1}^{d} \delta(h_i(x_i) = c_i)$ and $\hat{p}_h(c_1, ..., c_d) = \sum_{x_1,...,x_d} \hat{p}(x_1, ..., x_d) \prod_{i=1}^{d} \delta(h_i(x_i) = c_i)$, where $p(x_1, ..., x_d)$ is the unknown true distribution, and $\hat{p}(x_1, ..., x_d)$ is the empirical (maximum likelihood) estimate. The delta function, $\delta(arg)$, takes a value of 1 only when arg is true, and 0 otherwise. We can extend to the original space with the model assumption in Equation (1). For example, $p_h(x_1, ..., x_d) = \sum_{c_1,...,c_d} p_h(c_1, ..., c_d) \prod_{i=1}^{d} q(x_i|c_i)$.

The key difference between PAC learning and the PAC-Bayesian framework is the following notion of a random predictor, which is a distribution $\mathcal{Q}(h)$, learnt over the hypothesis space $\mathcal{H}$. Inference works as follows: for a new sample $(x_1, ..., x_d)$, we first draw a hypothesis h from $\mathcal{H}$ at random according to the distribution $\mathcal{Q}(h)$. We then return $q(x_1, ..., x_d)$ according to the model described by Equation (1) and the clustering function h. The PAC-Bayesian framework therefore allows for a second level of averaging over $\mathcal{Q}$, and we can define the induced distributions: $p_{\mathcal{Q}}(c_1, ..., c_d) = \sum_h \mathcal{Q}(h) p_h(c_1, ..., c_d)$ and $\hat{p}_{\mathcal{Q}}(c_1, ..., c_d) = \sum_h \mathcal{Q}(h) \hat{p}_h(c_1, ..., c_d)$. Again, we can extend to the original space with $p_{\mathcal{Q}}(x_1, ..., x_d)$ and $\hat{p}_{\mathcal{Q}}(x_1, ..., x_d)$ using the model assumption in Equation (1). Note that $p_{\mathcal{Q}}(x_1, ..., x_d)$ is unknown since $p(x_1, ..., x_d)$ is unknown; but the goal is to bound some notion of generalization error, such as the KL-divergence $\mathbb{KL}(\hat{p}_{\mathcal{Q}}(x_1, ..., x_d)||p_{\mathcal{Q}}(x_1, ..., x_d))$.

The **Change of Measure Inequality (CMI)** (Seldin and Tishby, 2010) is central to almost every PAC-Bayesian bound, so we briefly state it here. For any measurable function $\phi(h)$ on $\mathcal{H}$ and for any distributions $\mathcal{Q}(h)$ and $\mathcal{P}(h)$:

$$\mathbb{E}_{\mathcal{Q}(h)}[\phi(h)] \leq \mathbb{KL}(\mathcal{Q}||\mathcal{P}) + \ln \mathbb{E}_{\mathcal{P}(h)} \left[e^{\phi(h)} \right] \tag{2}$$

where $\mathbb{KL}(\mathcal{Q}||\mathcal{P}) = \mathbb{E}_{\mathcal{Q}(h)} \left[\ln \frac{\mathcal{Q}(h)}{\mathcal{P}(h)} \right]$ is the KL-divergence between $\mathcal{Q}$ and $\mathcal{P}$. The proof is fairly straightforward and is a direct consequence of rewriting $\phi(h)$ as $\ln \left(e^{\phi(h)} \frac{\mathcal{Q}(h)}{\mathcal{P}(h)} \frac{\mathcal{P}(h)}{\mathcal{Q}(h)} \right)$.

Seldin and Tishby (2010) apply the CMI with $\phi(h) = N \cdot \mathbb{KL}(\hat{p}_h(x_1, ..., x_d)||p_h(x_1, ..., x_d))$ and simplify the KL-divergence term by recognizing that 1) $\{q(c_i|x_i)\}_{i=1}^{d}$ defines a distribution over all possible clusterings, and hence $\mathcal{Q} = \{q(c_i|x_i)\}_{i=1}^{d}$; and 2) a specific $\mathcal{P}$, which they call the prior, can be defined without making any assumptions on the true distribution $p(x_1, ..., x_d)$. Note that $\mathcal{P}$ is not a prior in the Bayesian sense: 1) it indicates preference on the structure of the hypothesis, not an assumption on the data generating distribution, although the latter could be a consequence of the former; 2) the bound holds regardless of $\mathcal{P}$; and 3) the bound holds regardless of $\mathcal{Q}$, which is not necessarily the Bayes posterior.

The following prior on $\mathcal{H}$ makes no assumptions on $p(x_1, ..., x_d)$. We present a simplified version of the prior developed by Seldin and Tishby (2010):

$$\mathcal{P}(h) \geq \frac{1}{\exp\left[\sum_{i=1}^{d} m_i \ln n_i + n_i \ln m_i\right]} \tag{3}$$

The prior is based on a combinatorial argument. In order to select a clustering function h_i for some i, we first need to pick a cardinality profile (number of elements per cluster) for the m_i clusters; there are $n_i^{m_i}$ such profiles, hence the first term in the sum. Next, given a cardinality profile, we need to bound the number of ways in which each of the n_i elements can be assigned to the clusters given their sizes; there are at most $m_i^{n_i}$ possibilities, hence the second term in the sum. The CMI with $\phi(h) = N \cdot \mathbb{KL}(\hat{p}_h(x_1, ..., x_d)||p_h(x_1, ..., x_d))$, our modified prior, and a few information theoretic results lead to the following bound.

PAC-Bayesian Clustering: For any distribution p over $\mathcal{X}^{(1)} \times ... \times \mathcal{X}^{(d)}$ and an i.i.d. sample S of size N according to p, with probability at least $1-\delta$, for all distributions of cluster functions $\mathcal{Q} = \{q(c_i|x_i)\}_{i=1}^{d}$, the following holds:

$$\mathbb{KL}(\hat{p}_{\mathcal{Q}}(x_1, ..., x_d)||p_{\mathcal{Q}}(x_1, ..., x_d)) \leq \frac{\sum_{i=1}^{d} n_i \ln m_i + K_1}{N} \tag{4}$$

where $K_1 = \sum_{i=1}^{d} m_i \ln n_i + (M-1)\ln(N+1) + \ln\frac{d+1}{\delta}$, and $M = \prod_{i=1}^{d} m_i$. Although this shows convergence, in applications such as language modeling, we are interested in directly bounding the test set perplexity or cross-entropy. Seldin and Tishby (2010) smooth $\hat{p}_{\mathcal{Q}}(x_1, ..., x_d)$ to bound $\mathbb{E}_{p(x_1,...,x_d)}[-\ln \hat{p}_{\mathcal{Q}(x_1,...,x_d)}]$ and provide the following useful result based on Equation (4).

Bound on Cross-Entropy: For any probability measure p over $\mathcal{X}^{(1)} \times ... \times \mathcal{X}^{(d)}$ and an i.i.d. sample S of size N according to p, with probability $1-\delta$ for all distributions of cluster functions $\mathcal{Q} = \{q(c_i|x_i)\}_{i=1}^{d}$:

$$\mathbb{E}_{p(x_1,...,x_d)}[-\ln \hat{p}_{\mathcal{Q}}(x_1, ..., x_d)] \leq -I(\hat{p}_{\mathcal{Q}}(c_1, ..., c_d)) + \ln(M)\sqrt{\frac{\sum_{i=1}^{d} n_i \ln m_i + K_1}{2N}} + K_2 \tag{5}$$

where $\hat{p}_{\mathcal{Q}}(x_1, ..., x_d)$ is now the *smoothed* empirical estimate induced by $\mathcal{Q}$, $I(\hat{p}_{\mathcal{Q}}(c_1, ..., c_d)) = \sum_{i=1}^{d} H(\hat{p}_{\mathcal{Q}}(c_i)) - H(\hat{p}_{\mathcal{Q}}(c_1, ..., c_d))$ is the multi-information of the clustering, M and K_1 are as defined in Equation (4), and K_2 is an additional term, $K_2 \geq I(\hat{p}_{\mathcal{Q}}(c_1, ..., c_d))$, and the bound is non-negative.

3 Language Models

Since language modeling is yet another density estimation problem in which we want to minimize the test set perplexity, the bound in Equation (5) readily applies to both word n-grams and class-based n-grams. Note that the bounds are on cross-entropy, which is log perplexity, but we use the two terms almost interchangeably. We are now interested in estimating the unknown true distribution $p(v_1, ..., v_n)$ over the space $\mathcal{V}^n$, where $\mathcal{V}$ is some vocabulary consisting of $V = |\mathcal{V}|$ words. The degenerate case, $d = 1$, $\mathcal{X}^{(1)} = \mathcal{V}^n$, is the case of word n-grams and results in a bound that is dominated by $n_1 = |\mathcal{X}^{(1)}| = V^n$. This suggests that the number of training samples, N, must be on the same order as V^n for the bound (and hence the estimate) to be meaningful.

It is also clear why class-based models are favored whenever they work. In this case, $d = n$, $\mathcal{X}^{(i)} = \mathcal{V}$ for all $1 \leq i \leq d$, and the bound in Equation (5) reduces to something linear in V (since $\forall i$, $n_i = |\mathcal{X}^{(i)}| = V$). Moreover, the clustering function is the same for all i – that is, word clusters do not depend on the position in the n-gram. Assuming K word clusters, the number of training examples, N, only needs to be on the order of $K^n + nV$, achieving effective small sample generalization especially when $K << V$. In the following subsections, we extend the bound to sequences and present a unique approach to regularize the bound.

3.1 Sequence Clustering

We have discussed two extreme cases, namely $d = 1$ and $d = n$, that correspond to word n-grams and class-based n-grams, respectively. In practice, they are often interpolated to retain the advantages of both, as shown in the following model:

$$q(v_1, ..., v_n) = \alpha q(v_1, ..., v_n) + (1 - \alpha) \sum_{c_1,...,c_n} q(c_1, ..., c_n) \prod_{i=1}^{n} q(v_i|c_i) \tag{6}$$

for some $0 < \alpha < 1$. A Bayesian interpretation of the above model is to select between the n-gram and the class-based model with probabilities α and $1 - \alpha$, respectively. In other words, for each n-gram $(v_1, ..., v_n)$, we simply flip an α-biased coin to decide on one of the two models. In this paper, we interpolate across the entire spectrum, $1 \leq d \leq n$, instead of just the extreme cases – that is, we capture clusters over not just words, but also sequences of words (phrases). Previous results by Deligne and Bimbot (1995), Ries et al. (1996), and Justo and Torres (2007) indicate that clustering over phrases is practically useful and leads to significant improvements.

Suppose our goal is to estimate the probability of a trigram, for example, "the cat sat." In the case of $d = 1$, we directly estimate the joint probability $p(the, cat, sat)$. In the standard class-based model, where $d = 3$, we estimate with the model $p(the, cat, sat) = \sum_{c_1,c_2,c_3} p(c_1, c_2, c_3)p(the|c_1)p(cat|c_2)p(sat|c_3)$. The intermediate cases, such as $d = 2$ in this example, are often neglected. The theory we subsequently develop interpolates over all four segmentations, including the missing ones: $p(the, cat, sat) = \sum_{c_1,c_2} p(c_1, c_2)p(the\ cat|c_1)p(sat|c_2)$ as well as $p(the, cat, sat) = \sum_{c_1,c_2} p(c_1, c_2)p(the|c_1)p(cat\ sat|c_2)$.

In general, an n-gram has 2^{n-1} possible segmentations, as illustrated in the previous example. Suppose $f \in \mathcal{F}$ is a particular segmentation from the space of all possible segmentations, and we explicitly define it as the following mapping:

$$f : \mathcal{V}^n \mapsto \mathcal{X}^{(1)} \times ... \times \mathcal{X}^{(d)} \tag{7}$$

where $1 \leq d \leq n$ and f is simply a segmentation that does not modify the joint distribution; that is, $p(v_1, ..., v_n) = p(x_1, ..., x_d)$. If f is fixed *a priori*, we can immediately apply the bounds derived in Equation (5) over the segmented space $\mathcal{X}^{(1)} \times ... \times \mathcal{X}^{(d)}$. This is the case where we decide on a model, such as the standard class-based model ($d = n$), and simply use it.

An extension to the case of interpolated models is straightforward. We modify the hypothesis space $\mathcal{H}$ to not only include all possible clusterings, but also all possible segmentations. The new random prediction $\mathcal{Q}$ over $\mathcal{H}$ works as follows: given an n-gram $(v_1, ..., v_n)$, draw a segmentation $f \in \mathcal{F}$ according to the distribution $\pi = (\pi_1, ..., \pi_{2^{n-1}})$, where the segmentations are indexed by $j = 1, ..., 2^{n-1}$ (the ordering does not matter), and π_j is the probability of drawing segmentation j; pick a clustering as in the random classifier described in Equation (5) for the new segmented space; and estimate $q(v_1, ..., v_n)$ according to the model described by the previous steps. The bound, in terms of π, is given below.

PAC-Bayes Sequence Clustering: For any probability measure p over $\mathcal{V}^n$, and an i.i.d. sample S of size N drawn according to p, with probability $1 - \delta$ for all distributions of segmentations π and for all distributions of cluster functions $\mathcal{Q}$:

$$\mathbb{E}_{p(v_1,...,v_n)}[-\ln \hat{p}_{\mathcal{Q}}(v_1, ..., v_n)] \leq \sum_{j=1}^{2^{n-1}} \left(K_3(j) + \ln(M(j)) \sqrt{\frac{\sum_{i=1}^{d(j)} V^{a_i(j)} \ln m_i(j) + K_1(j)}{2N}} \right) \pi_j$$
$$K_3(j) = -I(\hat{p}_{\mathcal{Q}}(c_1, ..., c_{d(j)})) + K_2(j) \tag{8}$$

where $\forall j\ \forall i,\ 1 \leq a_i(j) \leq n$, and $\forall j,\ \sum_{i=1}^{d(j)} a_i(j) = n$, and $V^{a_i(j)}$ simply replaces n_i in Equation (5) for a given j. The term $K_2(j)$ is from Equation (5). Note that all terms such as $m_i(j)$, the number of clusters corresponding to the space, their product $M(j)$, and additional terms $K_1(j)$, $K_2(j)$ now depend on the segmentation j since $X^{(i)}$ and $d(j)$ depend on j.

We can favor certain segmentations (e.g. those that require few training examples), but note that the bound above is true regardless of the distribution over possible segmentations, π. Also, the bound is dominated by the exponent $a_i(j)$ and the constraint $\sum_{i=1}^{d(j)} a_i(j) = n$. Hence, the bound is polynomial in V for all segmentations except the standard class-based setting where $d(j) = n$, in which case $\forall i,\ a_i(j) = 1$. For example, if $d(j) = n - 1$ for some segmentation j, there exists some i such that $a_i(j) = 2$ and hence represents clusters of bigrams. If $d(j) = n - 2$, there exists some segmentation j, and a space i such that $a_i(j) = 3$, and so on until $d(j) = 1$, and this is the case of word n-grams where $a_1(j) = n$.

3.2 Bound Minimization

Imposing the restriction $\forall j\ \forall i,\ a_i(j) = 1$ is simple, and although it can guarantee the small-sample benefits of a standard class-based model, it is not a useful strategy for incorporating the constraint. Since $a_i(j)$ corresponds to the original space $\mathcal{X}^{(i)}$ for a given j, restricting $a_i(j)$ would restrict $\mathcal{X}^{(i)}$ to an *a priori*, fixed set of V elements. To learn the best possible set of V elements, however, we need to minimize the *effective* size of $\mathcal{X}^{(i)}$. For example, suppose we are estimating trigrams over $\mathcal{V}^3$ using the following segmentation: $\mathcal{X}^{(1)} = \mathcal{V}$ and $\mathcal{X}^{(2)} = \mathcal{V}^2$ – i.e. a bigram over clusters of words and clusters of word bigrams. The unconstrained bound is dominated by $\mathcal{X}^{(2)}$. We can restrict the *effective* size of $\mathcal{X}^{(2)}$ by assigning zero probability to the vast majority of its elements, by constraining the hypothesis space to consider only cluster assignment functions $q(x_i|c_i)$ in which $n_2 << V^2$ of the elements have nonzero probability. Thus, every word sequence in $\mathcal{V}^d$ can be generated by the $d = n$ segmentation, but every other segmentation is constrained to generate at most a subset of $\mathcal{V}^d$ with nonzero probability.

We achieve this by imposing the restriction on the random predictor $\mathcal{Q}$. By Bayes rule, $q(c_i|x_i) = \frac{q(x_i|c_i)q(c_i)}{q(x_i)}$ and we can alternatively define $\mathcal{Q}$ as $\mathcal{Q} = \{q(c_i), q(x_i), q(x_i|c_i)\}_{i=1}^{d}$. Our goal is to learn a $\mathcal{Q}$ that minimizes the RHS of Equation (5), which includes maximizing the multi-information term, as well as constraining n_i. As expected, $q(x_i)$ controls the absolute size of $\mathcal{X}^{(i)}$ and $q(x_i|c_i)$ controls the effective size based on the clustering. The dominant term in all of our bounds is n_i (or a_i, with $n_i = V^{a_i}$), which results from the second term in the prior defined in Equation (3), since it bounds the number of ways in which the n_i items can be assigned to the m_i clusters. Alternatively, we can represent this quantity with an upper bound, $\left(\sum_{c_i} \|q(x_i|c_i)\|_0\right) \ln m_i$. We can write $q(x_i) = \sum_{c_i} q(x_i|c_i)q(c_i)$, and $n_i = \|q(x_i)\|_0 = \|\sum_{c_i} q(x_i|c_i)q(c_i)\|_0$; by the triangle inequality and scale invariance of the l_0 norm, this is less than or equal to $\sum_{c_i} \|q(x_i|c_i)\|_0$. We therefore limit the upper bound, $\sum_{c_i} \|q(x_i|c_i)\|_0$, by sparsifying $q(x_i|c_i)$ for every cluster c_i.

The Optimization Problem: Given some segmentation, we want to find a random predictor $\mathcal{Q}$ – a class-based model over the fixed segmentation – such that the bound in Equation (5) is minimized, which is given by the following optimization problem:

$$\begin{aligned} \underset{\mathcal{Q}}{\text{maximize}} \quad & I(\hat{p}_{\mathcal{Q}}(c_1, ..., c_d)) \\ \text{subject to} \quad & \|q(x_i|c_i)\|_0 \leq V,\ \forall\, c_i \in \mathcal{C}^{(i)}, i = 1, \ldots, d \end{aligned} \tag{9}$$

Since such optimization problems are known to be NP-complete, we use a computationally tractable proxy. The standard practice is to use the l_1 norm instead of the l_0 norm; although non-convex, we resort to the l_α norm, $0 < \alpha < 1$, since $q(x_i|c_i)$ is a probability vector with a fixed l_1 norm. We therefore solve the following problem:

$$\begin{aligned} \underset{\mathcal{Q}}{\text{maximize}} \quad & I(\hat{p}_{\mathcal{Q}}(c_1, ..., c_d)) \\ \text{subject to} \quad & \|q(x_i|c_i)\|_\alpha \leq V,\ \forall\, c_i \in \mathcal{C}^{(i)}, i = 1, \ldots, d \end{aligned} \tag{10}$$

We have shown that one way to regularize the bound for a non-trivial sequence clustering problem, regardless of whether the segmentation is fixed or if we are interpolating across all segmentations, is to sparsify the cluster assignment probabilities for every cluster. There are many ways to sparsify a probability vector (Pilanci et al., 2012; Kyrillidis et al., 2013), and we select the l_α norm, $0 < \alpha <$

1, for its simplicity and success in other applications (Chartrand and Staneva, 2008). Our approach guarantees manageable bounds on the test set cross-entropy for a general class of SLMs, without making any assumptions on the true distribution $p(v_1, ..., v_n)$.

The Bayesian Connection A Bayesian interpretation of our regularization provides additional insight into other successful models, such as the hierarchical Pitman-Yor language model (HPYLM). In our approach, we impose the restriction $\|q(x_i|c_i)\|_\alpha \leq V$, $0 < \alpha < 1$, for every cluster c_i. It can be shown that this is equivalent to a sub-exponential prior on $q(x_i|c_i)$ (Hastie et al., 2009). Since $q(x_i) = \sum_{c_i} q(x_i|c_i)q(c_i)$ and we make the assumption that $q(x_i|c_i)$ is sub-exponential for every c_i, we are consequently assuming that $q(x_i)$ is also sub-exponential. Although the PAC-Bayesian bounds hold regardless of the true distribution, our regularization technique implicitly assumes that it is heavy-tailed.

The key to HPYLM's success within the Bayesian setting is a better prior that matches the heavy-tailed distribution of natural language (Teh, 2006) – the regularization approach developed in this paper reassuringly corresponds to the assumption that the true distribution is heavy-tailed (sub-exponential). On the other hand, it may be possible to derive provable guarantees for HPYLM within the context of our clustering model. The main difference between HPYLM and the less successful Dirichlet process (DP) is the Chinese restaurant process, which assigns new tables (clusters) to customers (samples) much more aggressively in the former model than in the latter (Teh, 2006). HPYLM therefore has far fewer customers (samples) per table (cluster) than DP, resulting in significantly sparser $q(x_i|c_i)$.

4 An Efficient HMM Algorithm

The hidden Markov model (HMM) is a popular tool for modeling sequences and has been used in several speech and language clustering tasks (Rabiner, 1989; Smyth, 1997; Li and Biswas, 1999). Over its rich history, several techniques, including regularization and sparsification of the HMM parameters, have been developed (Bicego et al., 2007; Bharadwaj et al., 2013). The goal of this section is to show how our bound easily fits into a well-established model such as the HMM.

We can rewrite the standard class-based model by making a Markov assumption on $q(c_1, ..., c_n)$:

$$q(x_1, ..., x_d, c_1, ..., c_d) = \prod_{i=1}^{d} q(x_i|c_i)q(c_i|c_{i-1}) \tag{11}$$

where $\{x_i\}_{i=1}^d$ is some segmentation of $(v_1, ..., v_n) \in \mathcal{V}^n$. The HMM literature refers to c_i as the hidden state, $q(x_i|c_i)$ as the observation probability, and $q(c_i|c_{i-1})$ as the state transition probability (Rabiner, 1989). If we consider each state of the HMM to be a cluster, then as before, $q(c_i|x_i) = q(x_i|c_i)\frac{q(c_i)}{q(x_i)}$ is a distribution over all possible clustering functions. To solve the optimization problem described in Equation (10), we need to maximize the multi-information $I(q(c_1, ..., c_n))$ while satisfying the constraint $\|q(x_i|c_i)\|_\alpha \leq V$. We can rewrite the constrained optimization problem as an unconstrained problem using a Lagrangian, and solve for $q(x_i|c_i)$ with an l_α regularized version of the expectation maximization (EM) algorithm, similar to Bharadwaj et al. (2013).

To maximize the multi-information term $I(q(c_1, ..., c_d))$ in Equation (10), we sparsify the state transition probabilities $q(c_i|c_{i-1})$. This provably works when we use l_α regularization, $0 < \alpha < 1$ for sparsifying $q(c_i|c_{i-1})$. The Renyi α-entropy of a random variable with some probability distribution q is defined to be $H_\alpha(q) = \frac{\alpha}{1-\alpha} \log \|q\|_\alpha$ and there are two useful results we use (Principe, 2010): 1) $\lim_{\alpha \to 1} H_\alpha(q) = H(q)$, where $H(q)$ is the Shannon entropy; and 2) $H_\alpha(q)$ is non-increasing in α. Thus, for $\alpha < 1$, $H_\alpha(q)$ is an upper bound on the Shannon entropy. Since l_α regularization minimizes the Renyi α-entropy, which for $0 < \alpha < 1$ is an upper bound on the Shannon entropy, it effectively maximizes the mutual information between c_i and c_{i-1}, given that $I(\hat{q}_\mathcal{Q}(c_i, c_{i-1})) = H(\hat{q}_\mathcal{Q}(c_i)) - H(\hat{q}_\mathcal{Q}(c_i|c_{i-1}))$.

Thus, we have shown that at least in the context of clustering, sparsifying both the observation probabilities and the state transition probabilities of an HMM using the l_α prior directly minimizes generalization error.

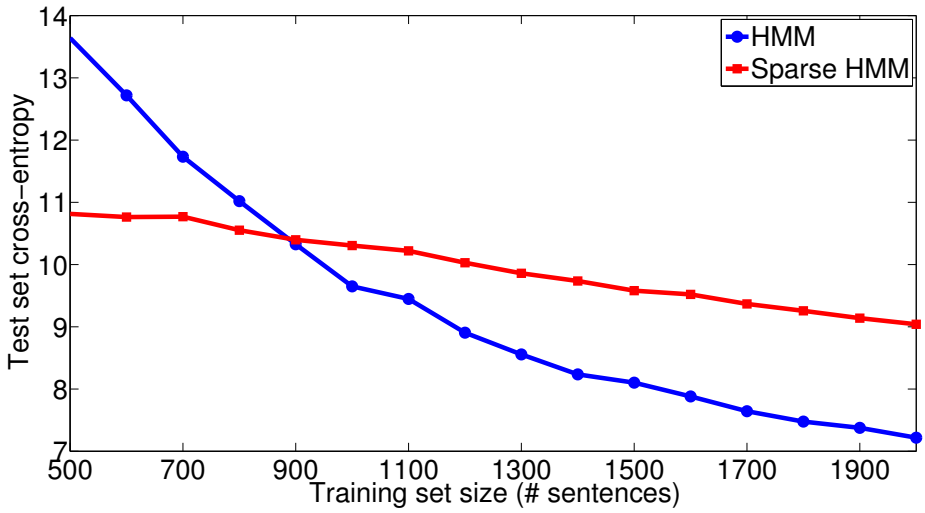


Figure 1: Test set cross-entropy of HMM vs l_α-regularized (sparse) HMM as a function of the number of training sentences

5 Experiments

We test our approach on a subset of the resource management (RM) corpus (Price et al., 1993), which consists of naval commands that span approximately $V = 1000$ words. First, we show that l_α regularization works. Figure 1 shows the estimated test set cross-entropy of an unregularized HMM and of an l_α-regularized HMM as a function of the number of training sentences. We vary the training set size from 10 to 2000 sentences and test the models on 800 sentences; Figure 1 reports the average cross-entropy on brackets of training sizes – 10-100, 110-200, and so on. The l_α-regularized HMM requires additional tunable parameters such as the value of α. To simplify the search on a separate 300 sentence development set, we make a (rather restrictive) assumption that α for both the transition and observation probabilities is the same, and that α is independent of the size of the training set. Our solutions are therefore not optimal, but adequate to demonstrate our claims. To ensure that the cross-entropy is bounded, we smooth all estimates with add-one smoothing. For small training datasets, the unregularized HMM learns models that assign near-zero likelihood to some of the test sentences; hence, we only present results for training set sizes greater than 500 sentences.

Like many other model selection results, Figure 1 suggests that model sparsity is essential when training datasets are small. In this example, about 900 sentences are required for the unregularized HMM to outperform the sparse HMM. In the context of the theory developed in earlier sections, it was shown that test set cross-entropy is proportional to $\frac{n_i}{N}$, where N is the number of training examples. In practical settings, N is fixed; hence, the only strategy for minimizing cross-entropy is to minimize n_i. Figure 1 confirms that l_α regularization successfully sparsifies $q(x_i|c_i)$, the observation probabilities of the HMM, thereby minimizing n_i.

We also compare how the test set cross-entropy improves as a function of the training set size for four different models: 1) a baseline bigram model estimated over words; 2) a baseline class-based model using Brown's algorithm (Brown et al., 1992) with $K = 20$ clusters, learnt over the entire dataset so that it is also representative of knowledge-based approaches in which the true clusters are known *a priori*; 3) l_α-regularized HMM with 20 ergodic states; and 4) a special case of 3) in which the state transitions are constrained to artificially form $m_1 = 10$ word clusters (10 states) and $m_2 = 5$ clusters that represent word bigrams (10 states, where the 5 clusters are modeled with 2 left-to-right states each); therefore, the model represents an interpolation between the standard class-based model and word bigrams, but is of the exact same complexity as 2) and 3).

Figure 2 shows the estimated test set cross-entropy for each of the four models. The values of α used in our experiments are $\alpha = 0.7$ for the words only case and $\alpha = 0.9$ for sequences. It is clear

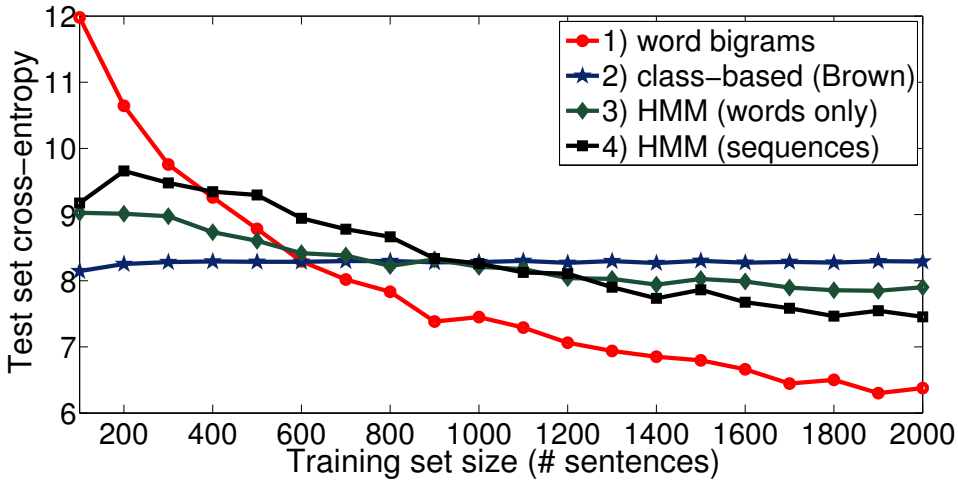


Figure 2: Test set cross-entropy as a function of the number of training sentences for the four settings

from Figure 2 that l_α regularization helps even in the case of a standard class-based model, the bound for which is already linear in V. With fewer than 100 sentences, l_α regularization can both learn the clusters and estimate their transitions reasonably well, and surpasses Brown for training set sizes of $N \geq 800$ sentences. Brown's algorithm in 2) finds clusters such that pairwise mutual information terms are maximized; in 3), we not only maximize the mutual information, but we also reduce the effective V by ensuring that each cluster (or state) specializes and represents as few words as possible. As the number of training examples increases, estimates of class transitions indeed improve, but the class-based assumption itself becomes too restrictive. In 4), which represents an interpolated model, we see the tradeoff achieved by incorporating sequences: for small training sets, the model achieves better generalization than word bigrams, but is worse than the class-based model; and for larger training sets, the interpolated model learns better representations of high frequency events and outperforms the class-based models represented by 2) and 3).

The value of α in 3) is 0.7, whereas α in 4) is 0.9; this seems counter-intuitive at first, but note that a smaller α does not necessarily imply sparser observation probabilities; however, it implies a heavier distribution in a Bayesian setting. A Bayesian interpretation therefore suggests that in 4), the model itself is better equipped to cope with heavy tails, whereas a more aggressive α is required in 3).

6 Conclusion

By defining a random clustering model (a model in which there is a distribution over possible cluster assignments, e.g. an HMM), it is possible to specialize published PAC-Bayesian cross-entropy bounds to the cases of n-gram and class-based n-gram estimation. A distribution over segmentations allows derivation of a cross-entropy bound on sequence clustering algorithms, which can be made useful by sparsifying the sequence cluster observation probabilities. An efficient l_α regularization technique can be used to maximize sparsity, thereby minimizing the test set cross-entropy.

Acknowledgements

We are grateful to the SST Group at Illinois and the anonymous reviewers for valuable feedback. Thanks also to Jitendra Ajmera, Om Deshmukh, and Ashish Verma for their contributions to the clustering algorithm. This work was supported by the NSF CDI Program Grant Number BCS 0941268 and ARO W9111NF-09-1-0383; the opinions expressed in this work are those of the authors and do not necessarily reflect the views of the funding agencies.

References

Hirotugu Akaike. 1973. Information theory and an extension of the maximum likelihood principle. In *Proceedings of the Second International Symposium on Information Theory*, pages 267–281.

Sujeeth Bharadwaj, Mark Hasegawa-Johnson, , Jitendra Ajmera, Om Deshmukh, and Ashish Verma. 2013. Sparse hidden Markov models for purer clusters. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 3098–3102.

Manuele Bicego, Marco Cristani, and Vittorio Murino. 2007. Sparseness achievement in hidden Markov models. In *Proceedings of of the International Conference on Inage Analysis and Processing*, pages 67–72.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Kenneth P. Burnham and David R. Anderson. 2002. *Model selection and multi-model inference: a practical information-theoretic approach*. Springer.

Rick Chartrand and Valentina Staneva. 2008. Restricted isometry properties and nonconvex compressive sensing. *Inverse Problems*, 24:1–14.

Stanley F. Chen and Josha Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13:359–394.

Stanley F. Chen. 2009. Performance prediction for exponential language models. In *Proceedings of NAACL HTL*.

Sabine Deligne and Frederic Bimbot. 1995. Language modeling by variable length sequences: theoretical formulation and evaluation of multigrams. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 169–172.

I.J. Good. 1953. The population frequencies of species and the estimation of population parameters. *Biometrica*, 40(3):237–264.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

Frederick Jelinek and Robert L. Mercer. 1980. Interpolated estimation of Markov source parameters from sparse data. In *Proceedings of Workshop on Pattern Recognition in Practice*, pages 381–397.

Raquel Justo and M. Ines Torres. 2007. Different approaches to class-based language models using word segments. *Computer Recognition Systems 2, Advances in Soft Computing*, 45:421–428.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 181–184.

Anastasios Kyrillidis, Stephen Becker, Volkan Cevher, and Christoph Koch. 2013. Sparse projections onto the simplex. *JMLR: Workshop and Conference Proceedings, Proceedings of the 30th International Conference on Machine Learning*, 28(2):235–243.

John Langford. 2005. Tutorial on practical prediction theory for classification. *The Journal of Machine Learning Research*, 6:273–306.

Cen Li and Gautam Biswas. 1999. Clustering sequence data using hidden Markov model representation. In *Proceedings of the SPIE '99 Conference on Data Mining and Knowledge Discovery*, pages 14–21.

G.J. Lidstone. 1920. Note on the general case of the Bayes-Laplace formula for inductive or $a\ posteriori$ probabilities. *Transactions of the Faculty of Actuaries*, 8:182–192.

David J.C. Mackay and Linda C. Bauman Peto. 1995. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1(03):289–308.

David McAllester. 1998. Some PAC-Bayesian theorems. In *COLT' 98 Proceedings of the eleventh annual conference on Computational Learning Theory*, pages 230–234.

David McAllester. 2003. Simplified PAC-Bayesian margin bounds. In *COLT' 03 Proceedings of the sixteenth annual conference on Computational Learning Theory*, pages 202–215.

Mesrob I. Ohannessian and Munther A. Dahleh. 2012. Rare probability estimation under regularly varying heavy tails. *JMLR: Workshop and Conference Proceedings, 25th Annual Conference on Learning Theory*, 23(21):1–24.

Mert Pilanci, Laurent El Ghaoui, and Venkat Chandrasekaran. 2012. Recovery of sparse probability measures via convex programming. In *Advances in Neural Information Processing Systems (NIPS)*, volume 25, pages 2429–2437.

P. Price, W.M. Fisher, J. Bernstein, and D.S. Pallett. 1993. Resource Management RM 2.0. In *Linguistic Data Consortium, Philadelphia*.

Jose C. Principe. 2010. *Information Theoretic Learning*. Springer.

Lawrence R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Klaus Ries, Finn Dag Buo, and Alex Waibel. 1996. Class phrase models for language modeling. In *ICSLP '96 Proceedings of the Fourth International Conference on Spoken Language*, pages 398–401.

Ronald Rosenfeld. 1996. A maximum entropy approach to adaptive statistical language modeling. *Computer, Speech, and Language*, 10:187–228.

Ronald Rosenfeld. 2000. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8).

Yevgeny Seldin and Naftali Tishby. 2010. PAC-Bayesian analysis of co-clustering and beyond. *The Journal of Machine Learning Research*, 11:3595–3646.

Padhraic Smyth. 1997. Clustering sequences with hidden Markov models. In *Advances in Neural Information Processing (NIPS)*, volume 9, pages 648–654.

Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992.

L.G. Valiant. 1984. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.

Co-learning of Word Representations and Morpheme Representations

Siyu Qiu
Nankai University
Tianjin, 300071, China
ppqq2356@gmail.com

Qing Cui
Tsinghua University
Beijing, 100084, China
cuiq12@mails.tsinghua.edu.cn

Jiang Bian
Microsoft Research
Beijing, 100080, China
jibian@microsoft.com

Bin Gao
Microsoft Research
Beijing, 100080, China
bingao@microsoft.com

Tie-Yan Liu
Microsoft Research
Beijing, 100080, China
tyliu@microsoft.com

Abstract

The techniques of using neural networks to learn distributed word representations (i.e., word embeddings) have been used to solve a variety of natural language processing tasks. The recently proposed methods, such as CBOW and Skip-gram, have demonstrated their effectiveness in learning word embeddings based on context information such that the obtained word embeddings can capture both semantic and syntactic relationships between words. However, it is quite challenging to produce high-quality word representations for rare or unknown words due to their insufficient context information. In this paper, we propose to leverage morphological knowledge to address this problem. Particularly, we introduce the morphological knowledge as both additional input representation and auxiliary supervision to the neural network framework. As a result, beyond word representations, the proposed neural network model will produce morpheme representations, which can be further employed to infer the representations of rare or unknown words based on their morphological structure. Experiments on an analogical reasoning task and several word similarity tasks have demonstrated the effectiveness of our method in producing high-quality words embeddings compared with the state-of-the-art methods.

1 Introduction

Word representation is a key factor for many natural language processing (NLP) applications. In the conventional solutions to the NLP tasks, discrete word representations are often adopted, such as the 1-of-v representations, where v is the size of the entire vocabulary and each word in the vocabulary is represented as a long vector with only one non-zero element. However, using discrete word vectors cannot indicate any relationships between different words, even though they may yield high semantic or syntactic correlations. For example, while *careful* and *carefully* have quite similar semantics, their corresponding 1-of-v representations trigger different indexes to be the hot values, and it is not explicit that *careful* is much closer to *carefully* than other words using 1-of-v representations.

To deal with the problem, neural network models have been widely applied to obtain word representations. In particular, they usually take the 1-of-v representations as the word input vectors in the neural networks, and learn new distributed word representations in a low-dimensional continuous embedding space. The principle of these models is that words that are highly correlated in terms of either semantics or syntactics should be close to each other in the embedding space. Representative works in this field include feed-forward neural network language model (NNLM) (Bengio et al., 2003), recurrent neural network language model (RNNLM) (Mikolov et al., 2010), and the recently proposed continues bag-of-words (CBOW) model and continues skip-gram (Skip-gram) model (Mikolov et al., 2013a).

However, there are still challenges for using neural network models to achieve high-quality word embeddings. First, it is difficult to obtain word embeddings for emerging words as they are not included in the vocabulary of the training data. Some previous studies (Mikolov, 2012) used one or more default indexes to represent all the unknown words, but such solution will lose information for the new words.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 141–150, Dublin, Ireland, August 23-29 2014.

Second, the embeddings for rare words are often of low quality due to the insufficient context information in the training data.

Fortunately, semantically or syntactically similar words often share some common morphemes such as roots, affixes, and syllables. For example, *probably* and *probability* share the same root, i.e., *probab*, as well as the same syllables, i.e., *pro* and *ba*. Therefore, morphological information can provide valuable knowledge to bridge the gap between rare or unknown words and well-known words in learning word representations. In this paper, we propose a novel neural network architecture that can leverage morphological knowledge to obtaining high-quality word embeddings. Specifically, we first segment the words in the training data into morphemes, and then employ the 1-of-v representations of both the words and their morphemes as the input to the neural network models. In addition, we propose to use morphological information as auxiliary supervision. Particularly, in the output layer of the neural network architecture, we predict both the words and their corresponding morphemes simultaneously. Moreover, we introduce extra coefficients into the network to balance the weights between word embeddings and morpheme embeddings. Therefore, in the back propagation stage, we will update the word embeddings, the morpheme embeddings, and the balancing coefficients simultaneously.

Our proposed neural network model yields two major advantages: on one hand, it can leverage three types of co-occurrence information, including co-occurrence between word and word (conventional), co-occurrence between word and morpheme (newly added), and co-occurrence between morpheme and morpheme (newly added); on the other hand, this new model allows to learn word embeddings and morpheme embeddings simultaneously, so that it is convenient to build the representations for unknown words from morpheme embeddings and enhance the representations for rare words. Experiments on large-scale public datasets demonstrate that our proposed approach can help produce improved word representations on an analogical reasoning task and several word similarity tasks compared with the state-of-the-art methods.

The rest of the paper is organized as follows. We briefly review the related work on word embedding using neural networks in Section 2. In Section 3, we describe the proposed methods to leverage morphological knowledge in word embedding using neural network models. The experimental results are reported in Section 4. The paper is concluded in Section 5.

2 Related Work

Neural Language Models (NLMs) (Bengio et al., 2003) have been applied in a number of NLP tasks (Collobert and Weston, 2008) (Glorot et al., 2011) (Mikolov et al., 2013a) (Mikolov et al., 2013b) (Socher et al., 2011) (Turney, 2013) (Turney and Pantel, 2010) (Weston et al.,) (Deng et al., 2013) (Collobert et al., 2011) (Mnih and Hinton, 2008) (Turian et al., 2010). In general, they learn distributed word representations in a continuous embedding space. For example, Mikolov et al. proposed the continuous bag-of-words model (CBOW) and the continuous skip-gram model (Skip-gram) (Mikolov et al., 2013a). Both of them assume that words co-occurring with the same context should be similar. Collobert et al. (Collobert et al., 2011) fed their neural networks with extra features such as the capital letter feature and the part-of-speech (POS) feature, but they still met the challenge of producing high-quality word embeddings for rare words.

Besides using neural network, many different types of models were proposed for estimating continuous representations of words, such as the well-known Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). However, Mikolov et al. (Mikolov et al., 2013c) have shown that words learned by neural networks are signicantly better than LSA for preserving linear regularities while LDA becomes computationally expensive on large datasets.

There were a lot of previous attempts to include morphology in continuous models, especially in the speech recognition field. Represent works include Letter n-gram (Sperr et al., 2013) and feature-rich DNN-LMs (Mousa et al., 2013). The first work improves the letter-based word representation by replacing the 1-of-v word input of restricted Boltzman machine with a vector indicating all n-grams of order n and smaller that occur in the word. Additional information such as capitalization is added as well. In the model of feature-rich DNN-LMs, the authors expand the inputs of the network to be a mixture of

selected full words and morphemes together with their features such as morphological tags. Both of these works intend to capture more morphological information so as to better generalize to unknown or rare words and to lower the out-of-vocabulary rate.

There are some other related works that consider morphological knowledge when learning the word embeddings, such as factored NLMs (Alexandrescu and Kirchhoff, 2006) and csmRNN (Luong et al., 2013), both of which are designed to handle rare words. In factored NLMs, each word is viewed as a vector of shape features (e.g., affixed, capitalization, hyphenation, and classes) and a word is predicted based on several previous vectors of factors. Although they made use of the co-occurrence of morphemes and words, the context information is lost after chopping the words and feeding the neural network with morphemes. In our model, we also utilize the co-occurrence information between morphemes, which has not been investigated before. In csmRNN, Luong *et al* proposed a hierarchical model considering the knowledge of both morphological constitutionality and context. The hierarchical structure looks more sophisticated, but the relatedness of words with morphological similarity are weaken by layers when combining morphemes into words. In addition, the noise accumulated in the hierarchical structure in building a word might be propagated to the context layer. In our model, the morphological and contextual knowledge are combined in parallel, and their contributions to the input vector are decided by a pair of learned tradeoff coefficients.

3 The Morpheme powered CBOW Models

In this section, we introduce the architecture of our proposed neural network model based on the CBOW model. In CBOW (see Figure 1), a sliding window is employed on the train text stream to obtain the training samples. In each sliding window, the model aims to predict the central word using the surrounding words as the input. Specifically, the input words are represented in the 1-of-v format. In the feed-forward process, these input words are first mapped into the embedding space by the same weight matrix M, and then the embedding vectors are summed up to a combined embedding vector. After that, the combined embedding vector is mapped back to the 1-of-v space by another weight matrix M', and the resulting vector is used to predict the central word after conducting softmax on it. In the back-propagation process, the prediction errors are propagated back to the network to update the two weight matrices. After the

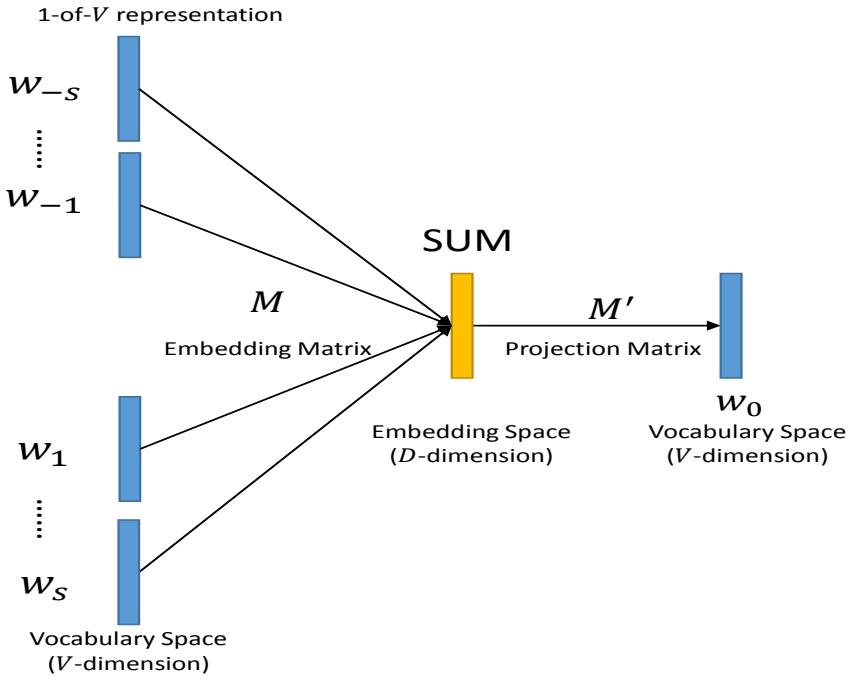


Figure 1: The CBOW model.

In our proposed model, we address the challenge of producing high-quality word embeddings for rare words and unknown words by leveraging the three types of co-occurrence information between words and morphemes.

On the input side, we segment the words into morphemes and put both the words and the morphemes as input. That is, the vocabulary for the 1-of-v representation contains both words and morphemes. As shown in Figure 2, the surrounding words in the sliding window are $w_{-s}, \cdots, w_{-1}, w_1, \cdots, w_s$ and their corresponding morphemes are $m_{-s,1}, m_{-s,2}, \cdots, m_{-s,t_{-s}}; \cdots; m_{-1,1}, m_{-1,2}, \cdots, m_{-1,t_{-1}}; m_{1,1}, m_{1,2}, \cdots, m_{1,t_1}; \cdots; m_{s,1}, m_{s,2}, \cdots, m_{s,t_s}$, where $2s$ is the number of the surrounding words and t_i is the number of morphemes for w_i $(i = -s, \cdots, -1, 1, \cdots, s)$. Note that t_i depends on the formation of w_i so that it may vary from word to word. If a word is also a morpheme, there will be two embedding vectors which are tagged differently. We use v_{w_i} and $v_{m_{i,j}}$ to represent the 1-of-v vectors of word w_i and morpheme $m_{i,j}$ respectively. On the input side, both the words and their morphemes are mapped into the embedding space by the same weight matrix M, and then the weighted sum v_I of the combination of word embeddings and the combination of morpheme embeddings is calculate as below,

$$v_I = \phi_w \cdot \sum_{\substack{i=-s \\ i \neq 0}}^{s} v_{w_i} + \phi_m \cdot \sum_{\substack{i=-s \\ i \neq 0}}^{s} \sum_{j=1}^{t_i} v_{m_{i,j}},$$

where ϕ_w and ϕ_m are the tradeoff coefficients between the combination of word embeddings and the combination of morpheme embeddings.

On the output side, we map the combined embedding vector v_I back to the 1-of-v space by another weight matrix M' to do the prediction. We have four settings of the structure. In the first setting, we only predict the central word w_0, and we name the model under this setting as *MorphemeCBOW*. In the second setting, we predict both the central word w_0 and its morphemes $m_{0,1}, m_{0,2}, \cdots, m_{0,t_0}$, and we name this setting as *MorphemeCBOW+*. In the above two settings, the tradeoff weights ϕ_w and ϕ_m are fixed. If we update the two weights in the learning process of *MorphemeCBOW*, we will get the third setting and we name it as *MorphemeCBOW**, while updating the two weights in *MorphemeCBOW+* yields the forth setting named *MorphemeCBOW++* .

Take *MorphemeCBOW+* as example, the objective is to maximize the following conditional co-occurrence probability,

$$\log(P(w_0 \mid \{w_i\}, \{m_{i,j}\})) + \log(\sum_{j=1}^{t_0} P(m_{0,j} \mid \{w_i\}, \{m_{i,j}\})), \tag{1}$$

where $\{w_i\}, \{m_{i,j}\}$ represent the bag of words and bag of morphemes separately. The conditional probability in the above formula is defined using the softmax function,

$$P(w_0 \mid \{w_i\}, \{m_{i,j}\}) = \frac{\exp(v'^T_{w_0} \cdot v_I)}{\sum_{v' \in V_O} \exp(v'^T \cdot v_I)} \quad , \quad P(m_{0,j} \mid \{w_i\}, \{m_{i,j}\}) = \frac{\exp(v'^T_{m_{0,j}} \cdot v_I)}{\sum_{v' \in V_O} \exp(v'^T \cdot v_I)}, \tag{2}$$

where V_O is the set of the output representations for the whole vocabulary; v' is used to differentiate with input representations; and $v'_{w_0}, v'_{m_{0,j}}$ represent the output embedding vectors of w_0 and $m_{0,j}$ respectively.

Usually, the computation cost for Formula (2) is expensive since it is proportional to the vocabulary size. In our model, we use negative sampling discussed in (Mikolov et al., 2013b) to speed up the computation. Particularly, we random select k negative samples $u_1, u_2, \cdots, u_k$ for each prediction target (word or morpheme). By using this technique, Formula (1) can be equally written as,

$$G(v_I) \equiv \log \sigma(v'^T_{w_0} \cdot v_I) + \sum_{j=1}^{t_0} \log \sigma(v'^T_{m_{0,j}} \cdot v_I) + \sum_{\substack{i=1 \\ u_i \neq w_0 \\ u_i \neq \forall m_{0,j}}}^{k} E_{u_i \sim P_n(u)}[\log \sigma(-v'^T_{u_i} \cdot v_I)],$$

where σ denotes the logistic function, and $P_n(u)$ is the vocabulary distribution used to select the negative samples. $P_n(u)$ is set as the 3/4rd power of the unigram distribution $U(u)$[1]. The negative samples should not be the same as any of the prediction targets w_0 and $m_{0,j}$ $(j = 1, \cdots, t_0)$. By using negative sampling, the training time spent on summing up the whole vocabulary in Formula (2) is greatly reduced so that it becomes linear with the number of the negative samples. Thus, we can calculate the gradient of $G(v_I)$ as below,

$$\frac{\partial G(v_I)}{\partial v_I} = (1 - \sigma(v'^T_{w_0} \cdot v_I)) \cdot \frac{\partial (v'^T_{w_0} \cdot v_I)}{\partial v_I} + \sum_{j=1}^{t_0} (1 - \sigma(v'^T_{m_{0,j}} \cdot v_I)) \cdot \frac{\partial (v'^T_{m_{0,j}} \cdot v_I)}{\partial v_I} - \sum_{\substack{i=1 \\ u_i \neq w_0 \\ u_i \neq \forall m_{0,j}}}^{k} [\sigma(v'^T_{u_i} \cdot v_I) \cdot \frac{\partial (v'^T_{u_i} \cdot v_I)}{\partial v_I}].$$

In the back-propagation process, the weights in the matrices M and M' are updated. When the training process converges, we take the matrix M as the learned word embeddings and morpheme embeddings.

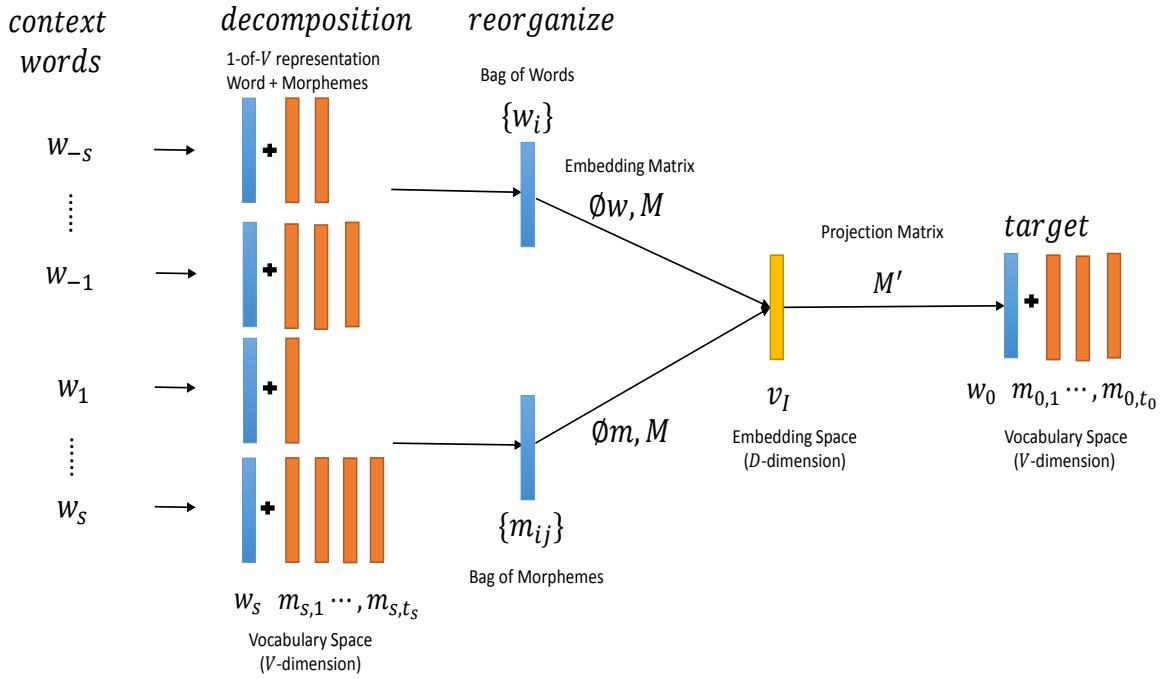


Figure 2: The proposed neural network model.

4 Experimental Evaluation

In this section we test the effectiveness of our model in generating high-quality word embeddings. We first introduce the experimental settings, and then we report the results on one analogical reasoning task and several word similarity tasks.

4.1 Datasets

We used two datasets for training: *enwiki9*[2] and *wiki2010*[3].

[1] http://www.cs.bgu.ac.il/~yoavg/publications/negative-sampling.pdf
[2] http://mattmahoney.ent/dc/enwik9.zip
[3] http://www.psych.ualberta.ca/~westburylab/downloads/westburylab.wikicorp.download.html

- The *enwiki9* dataset contains about 123.4 million words. We used Matt Mahoney's text pre-processing script[4] to process the corpus. Thus, we removed all non-Roman characters and mapped all digits to English words. In addition, words occurred less than 5 times in the training corpus were discarded. We used the learned word embeddings from *enwiki9* to test an analogical reasoning task described in (Mikolov et al., 2013a).

- The *wiki2010* dataset contains about 990 million words. The learned embeddings from this dataset were used on word similarity tasks as it was convenient to compare with the csmRNN model (Luong et al., 2013). We did the same data pre-processing as csmRNN did. That is, we removed all non-Roman characters and mapped all digits to zero.

4.2 Settings

In the analogical reasoning task, we used the CBOW model as the baseline. In both CBOW and our proposed model, we set the context window size to be 5, and generated three dimension sizes (100, 200, and 300) of word embeddings. We used negative sampling (Mikolov et al., 2013b) in the output layer and the number of negative samples is chosen as 3.

In the word similarity tasks, we used the csmRNN model as the baseline. The context window size of our model was set to be 5. To make a fair comparison with the csmRNN model, we conducted the same settings in our experiments as csmRNN. First, as csmRNN used the *Morfessor* (Creutz and Lagus, 2007) method to segment words into morphemes, we also used *Morfessor* as one of our word segmentation methods to avoid the influence caused by the segmentation methods. Second, as csmRNN used two existing embeddings C&W[5] (Collobert et al., 2011) and HSMN[6] (Huang et al., 2012) to initialize the training process, we also used the two embeddings as the initial weights of M in our experiments. Third, we set the dimension of the embedding space to 50 as csmRNN did.

In our model, we employed three methods to segment a word into morphemes. The first method is called *Morfessor*, which is a public tool implemented based on the minimum descriptions length algorithm (Creutz and Lagus, 2007). The second method is called *Root*, which segments a word into roots and affixes according to a predefined list in Longman Dictionaries. The third method is called *Syllable*, which is implemented based on the hyphenation tool proposed by Liang (Liang, 1983). Besides, the architecture of the proposed model can be specified into four types: *MorphemeCBOW*, *MorphemeCBOW**, *MorphemeCBOW+*, and *MorphemeCBOW++*. For the model *MorphemeCBOW* and *MorphemeCBOW+* with fixed tradeoff coefficients, we set the weights ϕ_w and ϕ_m to be 0.8 and 0.2 respectively; while for the other two models with updated tradeoff weights, the weights ϕ_w and ϕ_m are initialized as 1. These weight settings are chosen empirically.

4.3 Evaluation Tasks

4.3.1 Analogical reasoning task

The analogical reasoning task was introduced by Mikolov et al (Mikolov et al., 2013a). All the questions are in the form "a is to b is as c is to ?", denoted as $a : b \rightarrow c : ?$. The task consists of 19,544 questions involving semantic analogies (e.g., England: London $\rightarrow$ China: Beijing) and syntactic analogies (e.g., amazing: amazingly $\rightarrow$ unfortunate: unfortunately). Suppose that the corresponding vectors are $\overrightarrow{a}$, $\overrightarrow{b}$, and $\overrightarrow{c}$, we will answer the question by finding the word with the representation having the maximum cosine similarity to vector $\overrightarrow{b} - \overrightarrow{a} + \overrightarrow{c}$, i.e,

$$\max_{x \in V, x \neq b, x \neq c} (\overrightarrow{b} - \overrightarrow{a} + \overrightarrow{c})^T \overrightarrow{x}$$

where V is the vocabulary. Only when the computed word is exactly the answer word in evaluation set can the question be regarded as answered correctly.

[4] http://mattmahoney.net/dc/textdata.html
[5] http://ronan.collobert.com/senna/
[6] http://ai.stanford.edu/~ehhuang/

4.3.2 Word similarity task

The word similarity task was tested on five evaluation sets: WS353 (Finkelstein et al., 2002), SCWS* (Huang et al., 2012), MC (Miller and Charles, 1991), RG (Rubenstein and Goodenough, 1965) and RW (Luong et al., 2013), which contain 353, 1,762, 30, 65 and 2,034 pairs of words respectively. Table 1 shows some statistics about the datasets. Furthermore, the words in WS353, MC, RG are mostly frequent words, while SCWS* and RW have much more rare words and unknown words (i.e., unseen words in the training corpus) than the first three sets. The word distributions of these datasets are shown in Figure 3, from which we can see that RW contains the largest number of rare and unknown words. For the unknown words, we segmented them into morphemes, and calculated their word embeddings by summing up their corresponding morpheme embeddings. Each word pair in these datasets is associated with several human judgments on similarity and relatedness on a scale from 0 to 10 or 0 to 4. For example, (*cup*, *drink*) received an average score of 7.25, while (*cup*, *substance*) received an average score of 1.92. To evaluate the quality of the learned word embeddings, we computed Spearman's ρ correlation between the similarity scores calculated on the learned word embeddings and the human judgments.

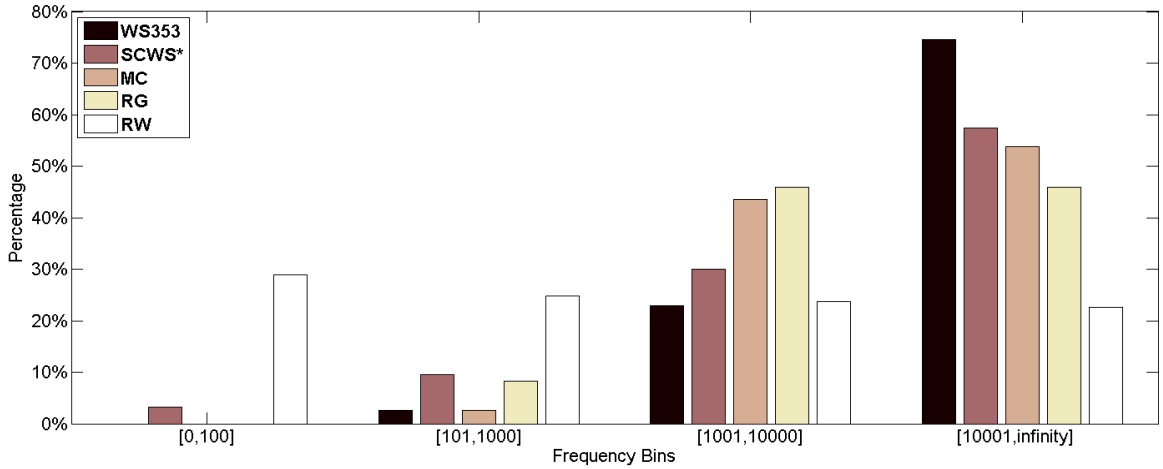


Figure 3: **Word distribution by frequency**. Distinct words in each test dataset are grouped according to frequencies. The figure shows the percentage of words in each bin.

Table 1: Statistics on the word similarity evaluation sets.

Dataset	Number of pairs	Number of words	Percentage of multi-segments words by *Morfessor*
WS353	353	437	28.15%
SCWS*	1726	1703	34.00%
RW	2034	2951	69.06%

4.4 Experimental Results

4.4.1 Results on analogical reasoning task

The experimental results on the analogical reasoning task are shown in Table 2, including semantic accuracy, syntactic accuracy, and total accuracy of all competition settings. Semantic/syntactic accuracy refers to the number of correct answers over the total number of all semantic/syntactic questions. From the results, we have the following observations:

- In MorphemeCBOW, we used the surrounding words and their morphemes to predict the central word. The total accuracies are all improved compared with baseline using the three word segmentation methods across three different dimensions of the embedding space. Generally, the improvements on semantic accuracies are less than those on syntactic accuracies. The reason is that the morphological information favors more for the syntactic tasks than the semantic tasks. Further-

more, the *Root* method achieved the best among the three segmentation methods, showing that the roots and affixes from the dictionary can help produce a high-quality morpheme segmentation tool.

- In MorphemeCBOW*, we predicted the central word, and updated the tradeoff coefficients in the learning process. We can see that the results are comparable or slightly better than MorphemeCBOW using the three word segmentation methods across three different dimensions of the embedding space, showing that updating the tradeoff coefficients may further boost the model performance under some specific settings.

- In MorphemeCBOW+, we predicted both the central word and its morphemes. MorphemeCBOW+ can provide slightly better results compared with MorphemeCBOW and MorphemeCBOW*, indicating that putting morphemes (especially roots) in the output layer can do extra help in generating high-quality word embeddings.

- In MorphemeCBOW++, we predicted the central word and its morphemes, and updated the tradeoff coefficients in the learning process. The performance under all of the three word segmentation methods got further improved compared with MorphemeCBOW+. It tells that the contributions from words and morphemes are different to the analogical reasoning task. According to our observations, the weight for words is usually higher than that for morphemes.

- By comparing MorphemeCBOW with MorphemeCBOW* as well as MorphemeCBOW+ with MorphemeCBOW++, we can observe that updating the weights of tradeoff coefficients seem to essentially boost syntactic accuracy by trading off a bit of semantic accuracy. As introduced in Section 4.2, in the fixed weight model the ratio of weight of morphemes to the weight of word is 0.25; while our experiment records show that the averaged ratio are 0.43 if the two weights are updated, meaning that the weight of the combination of morphemes increases and the contribution of the original word to the final combined embedding decreased. As a result, the syntactic accuracy which largely reflected in the morphological structure of a word increased, but the semantic accuracy hurts a little.

4.4.2 Results on word similarity task

Experimental results on the word similarity tasks are shown in Table 3[7],where the labels of C&W + csmRNN and HSMN + csmRNN mean that using C&W and HSMN to initialize csmRNN model as what had been introduced in the paper of Luong et al. In our experiments, the architecture of MorphemeCBOW* performs the best, so we only show the results related to MorphemeCBOW* in the table. We have the following observations from the results:

- On WS353, MC, RG, and SCWS*, MorphemeCBOW* performs consistently better than the csmRNN model, showing that our model can achieve better representations for common words.

[7]csmRNN embeddings are available on `http://www-nlp.stanford.edu/~lmthang/morphoNLM/`, Performances are tested based on the two embeddings.

Table 2: Performance of leveraging morphological information on the analogical reasoning task.

		(a) Baseline	(b) MorphemeCBOW			(c) MorphemeCBOW*			(d) MorphemeCBOW+			(e) MorphemeCBOW++		
Dimension	(%)	CBOW	*Morfessor*	*Syllable*	*Root*	*Morfessor*	*Syllable*	*Root*	*Morfessor*	*Syllable*	*Root*	*Morfessor*	*Syllable*	*Root*
100	Total	26.49	31.99	31.28	**32.49**	33.07	31.16	**34.04**	**33.26**	31.12	32.77	**38.86**	34.42	35.78
	Semantic	17.51	19.44	18.76	21.77	15.20	15.68	17.87	22.82	20.80	22.79	21.12	22.58	22.43
	Syntactic	33.96	42.42	41.68	41.40	47.92	44.02	47.48	41.93	39.70	41.07	53.59	44.26	46.87
200	Total	30.50	34.04	34.71	**36.29**	34.69	33.13	**36.50**	38.28	39.32	**39.53**	40.32	41.79	**43.29**
	Semantic	19.71	19.10	19.13	22.45	11.53	15.91	18.92	25.94	27.99	28.29	24.20	24.05	25.04
	Syntactic	39.46	46.45	47.65	47.79	53.92	47.44	51.10	48.52	48.74	48.86	53.72	56.53	58.45
300	Total	29.04	31.27	32.45	**36.12**	31.21	32.16	**35.63**	38.01	39.56	**39.70**	37.65	41.64	**41.96**
	Semantic	17.58	15.45	15.63	20.79	8.85	12.54	15.75	25.11	26.94	27.80	13.97	26.64	25.82
	Syntactic	38.56	44.41	46.44	48.86	49.79	48.47	52.14	48.72	50.05	49.58	57.32	54.10	55.36

Table 3: Performance of leveraging morphological information on the word similarity task.

Model	WS353 (%)	SCWS* (%)	MC(%)	RG(%)	RW(%)
C&W	49.73	48.45	57.33	48.22	21.93
C&W + csmRNN	58.27	49.09	60.22	58.92	**31.77**
C&W + MorphemeCBOW*	**63.81**	**53.30**	**74.33**	**61.22**	31.14
HSMN	62.58	32.09	66.18	64.51	1.97
HSMN + csmRNN	64.58	44.08	71.88	65.15	22.31
HSMN + MorphemeCBOW*	**65.19**	**53.40**	**81.62**	**67.41**	**32.13**
MorphemeCBOW*	63.45	53.40	77.40	63.78	32.88

- On RW, MorphemeCBOW* performs better than the csmRNN model when using the HSMN embeddings as the initialization. When using the C&W embeddings as the initialization, the performance of MorphemeCBOW* is also comparable with that of csmRNN. In particular, if we do not use any pre-trained embeddings to initialize our mode, it performed the best (32.88%), and it even beats the best performance of csmRNN with initializations (31.77%)[8]. The initialization is very important to a neural network. Suitable initialization will help increase the embedding quality which works like training with multi-epochs. However, as there are two matrix M and M' in our network structure, the initialization of both of them are more sensible. Furthermore, considering that the recursive structure of csmRNN will bring higher computation complexity, we can conclude that our model has excellent ability in learning the embeddings of rare words from pure scratch.

- The improvement on RW is more significant than those on the other four datasets. Considering that RW contains more rare and unknown words (See Figure 3), we verified our idea that leveraging morphological information will especially benefit the embedding of low-frequency words. More specifically, without sufficient context information for the rare words in the training data, building connections between words using morphemes will provide additional evidence for the model to generate effective embeddings for these rare words; and, by combining the high-quality morpheme embeddings to obtain the representations of the unknown words, the model does a good job in dealing with the new emerging words.

5 Conclusions and Future Work

We proposed a novel neural network model to learn word representations from text. The model can leverage several types of morphological information to produce high-quality word embeddings, especially for rare words and unknown words. Empirical experiments on an analogical reasoning task and several word similarity tasks have shown that the proposed model can generate better word representations compared with several state-of-the-art approaches.

For the future work, we plan to separate words and morphemes into several buckets according to their frequencies. Different buckets will be associated with different coefficients, so that we can tune the coefficients to approach even better word embeddings. We also plan to run our model on more training corpus to obtain the embedding vectors for rare words, especially those new words invented out recently. These emerging new words usually do not exist in standard training corpus such as Wikipedia, but exists in some noisy data such as news articles and web pages. How well our model performs on these new training corpus is an interesting question to explore.

References

Andrei Alexandrescu and Katrin Kirchhoff. 2006. Factored neural language models. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 1–4, New York City, USA, June. Association for Computational Linguistics.

[8] 34.36% in the paper of Luong et al; 32.06% in their project website, see note7

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March.

R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*, 12.

M. Creutz and K. Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *TSLP*.

L. Deng, X. He, and J. Gao. 2013. Deep stacking networks for information retrieval. In *ICASSP*, pages 3153–3157.

L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. 2002. Placing search in context: The concept revisited. In *ACM Transactions on Information Systems*.

X. Glorot, A. Bordes, and Y. Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving Word Representations via Global Context and Multiple Word Prototypes. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

F. M. Liang. 1983. Word hy-phen-a-tion by com-put-er. Technical report.

M.-T. Luong, R. Socher, and C. D. Manning. 2013. Better word representations with recursive neural networks for morphology. *CoNLL-2013*, 104.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013a. Efficient estimation of word representations in vector space. ICLR '13.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

T. Mikolov, W.-T. Yih, and G. Zweig. 2013c. Linguistic regularities in continuous space word representations. In *In NAACL-HLT*, pages 746–751.

T. Mikolov. 2012. *Statistical Language Models Based on Neural Networks*. Ph.D. thesis, Brno University of Technology.

G.A. Miller and W.G. Charles. 1991. Contextual correlates of semantic similarity. 6(1):1–28.

A. Mnih and G. E. Hinton. 2008. A scalable hierarchical distributed language model. In *NIPS*, pages 1081–1088.

Amr El-Desoky Mousa, Hong-Kwang Jeff Kuo, Lidia Mangu, and Hagen Soltau. 2013. Morpheme-based feature-rich language models using deep neural networks for lvcsr of egyptian arabic. In *ICASSP*, pages 8435–8439.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633, October.

R. Socher, C. C. Lin, A. Y. Ng, and C. D. Manning. 2011. Parsing natural scenes and natural language with recursive neural networks. In *ICML*.

Henning Sperr, Jan Niehues, and Alex Waibel. 2013. Letter n-gram-based input encoding for continuous space language models. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 30–39, Sofia, Bulgaria, August. Association for Computational Linguistics.

J. P. Turian, L.-A. Ratinov, and Y. Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *ACL*, pages 384–394.

P. D. Turney and P. Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

P. D. Turney. 2013. Distributional semantics beyond words: Supervised learning of analogy and paraphrase. *TACL*, pages 353–366.

J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*.

A Probabilistic Model for Learning Multi-Prototype Word Embeddings

Fei Tian[†], Hanjun Dai[*], Jiang Bian[‡], Bin Gao[‡],

Rui Zhang[⋆], Enhong Chen[†], Tie-Yan Liu[‡]

[†]University of Science and Technology of China, Hefei, P.R.China

[*]Fudan University, Shanghai, P.R.China

[‡]Microsoft Research, Building 2, No. 5 Danling Street, Beijing, P.R.China

[⋆]Sun Yat-Sen University, Guangzhou, P.R.China

[†]tianfei@mail.ustc.edu.cn, [†]cheneh@ustc.edu.cn, [*]daihanjun@gmail.com,

[‡]{jibian, bingao, tyliu}@microsoft.com, [⋆]rayz0620@hotmail.com

Abstract

Distributed word representations have been widely used and proven to be useful in quite a few natural language processing and text mining tasks. Most of existing word embedding models aim at generating only one embedding vector for each individual word, which, however, limits their effectiveness because huge amounts of words are polysemous (such as *bank* and *star*). To address this problem, it is necessary to build multi embedding vectors to represent different meanings of a word respectively. Some recent studies attempted to train multi-prototype word embeddings through clustering context window features of the word. However, due to a large number of parameters to train, these methods yield limited scalability and are inefficient to be trained with big data. In this paper, we introduce a much more efficient method for learning multi embedding vectors for polysemous words. In particular, we first propose to model word polysemy from a probabilistic perspective and integrate it with the highly efficient continuous Skip-Gram model. Under this framework, we design an Expectation-Maximization algorithm to learn the word's multi embedding vectors. With much less parameters to train, our model can achieve comparable or even better results on word-similarity tasks compared with conventional methods.

1 Introduction

Distributed word representations usually refer to low dimensional and dense real value vectors (a.k.a. word embeddings) to represent words, which are assumed to convey semantic information contained in words. With the exploding text data on the Web and fast development of deep neural network technologies, distributed word embeddings have been effectively trained and widely used in a lot of text mining tasks (Bengio et al., 2003) (Morin and Bengio, 2005) (Mnih and Hinton, 2007) (Collobert et al., 2011) (Mikolov et al., 2010) (Mikolov et al., 2013b).

While word embedding plays an increasingly important role in many tasks, most of word embedding models, which assume one embedding vector for each individual word, suffer from a critical limitation for modeling tremendous polysemous words (e.g. *bank*, *left*, *doctor*). Using the same embedding vector to represent the different meanings (we will call *prototype* of a word in the rest of the paper) of a polysemous word is somehow unreasonable and sometimes it even hurts the model's expression ability.

To address this problem, some recent efforts, such as (Reisinger and Mooney, 2010) (Huang et al., 2012), have investigated how to obtain multi embedding vectors for the respective different prototypes of a polysemous word. Specifically, these works usually take a two-step approach: they first train single prototype word representations through a multi-layer neural network with the assumption that one word only yields single word embedding; then, they identify multi word embeddings for each polysemous word by clustering all its context window features, which are usually computed as the average of single prototype embeddings of its neighboring words in the context window.

Compared with traditional single prototype model, these models have demonstrated significant improvements in many semantic natural language processing (NLP) tasks. However, they suffer from a

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 151–160, Dublin, Ireland, August 23-29 2014.

crucial restriction in terms of scalability when facing exploding training text corpus, mainly due to the deep layers and huge amounts of parameters in the neural networks in these models. Moreover, the performance of these multi-prototype models is quite sensitive to the clustering algorithm and requires much effort in clustering implementation and parameter tuning. The lack of probabilistic explanation also refrains clustering based methods from being applied to many text mining tasks, such as language modeling.

To address these challenges, in this work, we propose a new probabilistic multi-prototype model and integrate it into a highly efficient continuous Skip-Gram model, which was recently introduced in the well-known Word2Vec toolkit (Mikolov et al., 2013b). Compared with conventional neural network language models which usually set up a multi-layer neural network, Word2Vec merely leverages a three-layer neural network to learn word embeddings, resulting in greatly decreased number of parameters and largely increased scalability. However, similar to most of existing word embedding models, Word2Vec also assumes *one embedding for one word*. We break this limitation by introducing a new probabilistic framework which employs hidden variables to indicate which prototype each word belongs to in the context. In this framework, the conditional probability of observing word w_O conditioned on the presence of neighboring word w_I (i.e. $P(w_O|w_I)$) can be formulated as a mixture model, where *mixtures* corresponds to w_I's different *prototypes*. This is a more natural way to define $P(w_O|w_I)$, since it has taken the polysemy of word w_I into consideration. After defining the model, we design an efficient Expectation-Maximization (EM) algorithm to learn various word embedding vectors corresponding to each of w_I's prototypes. Evaluations on widely used word similarity tasks demonstrate that our algorithm produces comparable or even better word embeddings compared with either clustering-based multi-prototype models or the original Skip-Gram model. Furthermore, as a unified way to obtain multi word embeddings, our proposed method can effectively avoid the sensitivity to the clustering algorithm applied by previous multi-prototype word embedding approach.

The following of the paper is organized as follows: we introduce related work in Section 2. Then, Section 3 describes our new model and algorithm in details and conducts a comparison in terms of complexity between our algorithm and the previous method. We present our experimental results in Section 4. The paper is concluded in Section 5.

2 Related Work

Since the initial work (Bengio et al., 2003), there have been quite a lot of neural network based models to obtain distributed word representations (Morin and Bengio, 2005) (Mnih and Hinton, 2007) (Mikolov et al., 2010) (Collobert et al., 2011) (Mikolov et al., 2013b). Most of these models assume that one word has only one embedding, except the work of Eric Huang (Huang et al., 2012), in which the authors propose to leverage global context information and multi-prototype embeddings to achieve performance gains in word similarity task. To obtain multi-prototype word embeddings, this work conducts clustering on a word's all context words' features in the corpus. The features are the embedding vectors trained previously via a three-layer neural network. Each cluster's centroid is regarded as the embedding vector for each prototype. Their reported experimental results verify the importance of considering multi-prototype models.

Note that (Reisinger and Mooney, 2010) also proposes to deal with the word polysemy problem by assigning to each prototype a real value vector. However their embedding vectors are obtained through a tf-idf counting model, which is usually called as distributional representations (Turian et al., 2010), rather than through a neural network. Therefore, we do not regard their paper as very related to our work. The similar statement holds for other works on vector model for word meaning in context such as (Erk and Padó, 2008) (Thater et al., 2011) (Reddy et al., 2011) (Van de Cruys et al., 2011).

Our model is mainly based on the recent proposed Word2Vec model, more concretely, the continuous Skip-Gram model (Mikolov et al., 2013a) (Mikolov et al., 2013b). The continuous Skip-Gram model specifies the probability of observing the context words conditioned on the central word w_I in the window via a three-layer neural network. With less parameters to train (thus higher scalability), Word2Vec discovers interesting analogical semantic relations between words like *Japan - Tokyo = France - Paris*.

3 Model Description

In this section, we introduce our algorithm for learning multi-prototype embeddings in details. In particular, since our new model is based on the continuous Skip-Gram model, we first make a brief introduction to the Skip-Gram model. Then, we present our new multi-prototype algorithm and how we integrate it into the Skip-Gram model. After that, we propose an EM algorithm to conduct the training process. We also conduct a comparison on the number of parameters between the new EM algorithm and the state-of-the-art multi-prototype model proposed in (Huang et al., 2012), which can illustrate the efficiency superior of our algorithm.

3.1 Multi-Prototype Skip-Gram Model

In contrast to the conventional ways of using context words to predict the next word or the central word, the Skip-Gram model (Mikolov et al., 2013b) aims to leverage the central word to predict its context words. Specifically, assuming that the central word is w_I and one of its neighboring word is w_O, $P(w_O|w_I)$ is modeled in the following way:

$$P(w_O|w_I) = \frac{exp(V_{w_I}^T U_{w_O})}{\sum_{w \in W} exp(V_{w_I}^T U_w)}, \tag{1}$$

where W denotes the dictionary consisting of all words, $U_w \in R^d$ and $V_w \in R^d$ represent the d-dimensional 'output' and 'input' embedding vectors of word w, respectively. Note that all the parameters to be learned are the input and output embedding vectors of all words, i.e. $U = \{U_w | w \in W\}$ and $V = \{V_w | w \in W\}$. This corresponds to a three-layer neural network, in which U and V denote the two parameter matrices of the neural network. Compared with the conventional neural networks employed in the literature which yield at least four layers (including the look-up table layer), the Skip-Gram model greatly reduces the number of parameters and thus gives rise to a significant improvement in terms of training efficiency.

Our proposed Multi-Prototype Skip-Gram model is similar to the original Skip-Gram model in that it also aims to model $P(w_O|w_I)$ and uses two matrices (the input and output embedding matrices) as the parameters. The difference lies in that given word w_I, the occurrence of word w_O is described as a finite mixture model, in which each mixture corresponds to a prototype of word w_I. To be specific, suppose that word w has N_w prototypes and it appears in its h_w-th prototype, i.e., $h_w \in \{1, \cdots, N_w\}$ is the index of w's prototype. Then $P(w_O|w_I)$ is expanded as:

$$p(w_O|w_I) = \sum_{i=1}^{N_{w_I}} P(w_O|h_{w_I} = i, w_I) P(h_{w_I} = i|w_I) \tag{2}$$

$$= \sum_{i=1}^{N_{w_I}} \frac{exp(U_{w_O}^T V_{w_I,i})}{\sum_{w \in W} exp(U_w^T V_{w_I,i})} P(h_{w_I} = i|w_I), \tag{3}$$

where $V_{w_I,i} \in R^d$ refers to the embedding vector of w_I's i-th prototype. This equation states that $P(w_O|w_I)$ is a weighted average of the probabilities of observing w_O conditioned on the appearance of w_I's every prototype. The probability $P(w_O|h_{w_I} = i, w_I)$ takes the similar softmax form to equation (1) and the weight is specified as a prior probability of word w_I falls in its every prototype.

The general idea behind the Multi-Prototype Skip-Gram model is very intuitive: the surrounding words under different prototypes of the same word are usually different. For example, when the word *bank* refers to the side of a river, it is very possible to observe the corresponding context words such as *river*, *water*, and *slope*; however, when *bank* falls into the meaning of the financial organization, the surrounding word set is likely to be comprised of quite different words, such as *money*, *account*, and *investment*.

The probability formulation in (3) brings much computation cost because of the linear dependency of $|W|$ in the denominator $\sum_{w \in W} exp(U_w^T V_{w_I,i})$. To address this issue, several efficient methods have been proposed such as Hierarchical Softmax Tree (Morin and Bengio, 2005) (Mnih and Kavukcuoglu, 2013) and Negative Sampling (Mnih and Kavukcuoglu, 2013) (Mikolov et al., 2013b). Taking Hierarchical

Softmax Tree as an example, through a binary tree in which every word is a leaf node, word w_O is associated with a binary vector $b^{(w_O)} \in \{-1,+1\}^{L_{w_O}}$ specifying a path from the root of the tree to leaf w_O, where L_{w_O} is the length of vector $b^{(w_O)}$. Then the conditional probability is described as

$$P(w_O|h_{w_I}=i,w_I)=\prod_{t=1}^{L_{w_O}}P\left(b_t^{(w_O)}|w_I,h_{w_I}=i\right)=\prod_{t=1}^{L_{w_O}}\varsigma\left(b_t^{(w_O)}U_{w_O,t}^T V_{w_I,i}\right), \tag{4}$$

where $\varsigma(x)=1/(1+exp(-x))$ is the sigmoid function, and $U_{w_O,t}$ specifies the d-dimensional parameter vector associated with the t-th node in the path from the root to the leaf node w_O. Substituting (4) into (2) to replace the large softmax operator in (3) leads to a much more efficient probability form.

3.2 EM Algorithm

In this section, we describe the EM algorithm adopted to train the Multi-Prototype Skip-Gram model. Without loss of generality, we will focus on obtaining multi embeddings for a specified word $w \in W$ with N_w prototypes. Word w's embedding vectors are denoted as $V_w \in R^{d\times N_w}$. Suppose there are M word pairs for training: $\{(w_1,w),(w_2,w),\cdots,(w_M,w)\}$, where all the inputs words (i.e., word w) are the same, and the set of output words to be predicted are denoted as $\mathbb{X}=\{w_1,w_2,\cdots,w_M\}$. That is, $\mathbb{X}$ are M surrounding words of w in the training corpus.

For ease of reference and without loss of generality, we make some changes to the notations in Section 3.1. We will use h_m as the index of w's prototype in the pair (w_m,w), $m\in\{1,2,\cdots,M\}$. Besides, some new notations are introduced: $P(h_w=i|w_I)$ is simplified as π_i, and $\gamma_{m,k}$, where $m\in\{1,2,\cdots,M\}$, $k\in\{1,2,\cdots N_w\}$, are the hidden binary variables indicating whether the m-th presence of word w is in its k-th prototype, i.e. $\gamma_{m,k}=1_{h_m=k}$, where 1 is the indicator function. Other notations are the same as before: $V_{w,i}\in R^d$ is the embedding vector for word w's i-th prototype, $U_{w,t}\in R^d$ is the embedding vector for the t-th node on the path from the tree root to the leaf node representing word w, and $b_t^{(w)}\in\{-1,1\}$ is the t-th bit of the binary coding vector of word w along its corresponding path on the Hierarchical Softmax Tree.

Then the parameter set we aim to learn is $\Theta=\{\pi_1,\cdots,\pi_{N_w};U;V_w\}$. The hidden variable set is $\Gamma=\{\gamma_{m,k}|m\in(1,2,\cdots,M),k\in(1,2,\cdots,N_w)\}$. Considering equation (2) and (4), we have the log likelihood of $\mathbb{X}$ as below:

$$\begin{aligned}\log P(\mathbb{X},\Gamma|\Theta)&=\sum_{m=1}^{M}\sum_{k=1}^{N_w}\gamma_{m,k}\big(\log\pi_k+\log P(w_m|h_m=k,w)\big)\\&=\sum_{m=1}^{M}\sum_{k=1}^{N_w}\gamma_{m,k}\big(\log\pi_k+\sum_{t=1}^{L_{w_m}}\log\varsigma(b_t^{(w_m)}U_{w_m,t}^T V_{w,k})\big).\end{aligned} \tag{5}$$

With equation (5), the E-Step and M-Step are:

E-Step:

The conditional expectation of hidden variable $\gamma_{m,k}$, denoted as $\hat{\gamma}_{m,k}$, is:

$$\hat{\gamma}_{m,k}=P(\gamma_{m,k}=1|\mathbb{X},\Theta)=\frac{\pi_k P(w_m|h_m=k,w)}{\sum_{i=1}^{N_w}\pi_i P(w_m|h_m=i,w)}. \tag{6}$$

The Q function w.r.t. the parameters at the i-th iteration $\theta^{(i)}$ is written as:

$$\begin{aligned}Q(\theta,\theta^{(i)})&=\sum_{k=1}^{N_w}\sum_{m=1}^{M}\hat{\gamma}_{m,k}\left(\log\pi_k+\log P(w_m|h_m=k,w)\right)\\&=\sum_{m=1}^{M}\sum_{k=1}^{N_w}\hat{\gamma}_{m,k}\big(\log\pi_k+\sum_{t=1}^{L_{w_m}}\log\varsigma(b_t^{(w_m)}U_{w_m,t}^T V_{w,k})\big).\end{aligned} \tag{7}$$

M-Step:

π can be updated by

$$\pi_k = \frac{\sum_{m=1}^{M} \hat{\gamma}_{m,k}}{M}, \quad k = 1, 2 \cdots, N_w. \tag{8}$$

We leave the detailed derivations for equation (6), (7), and (8) to the appendix of the paper. Then we discuss how we obtain the update of the embedding parameters $U_{w_m,t}$ and $V_{w,k}$. Note that the optimization problem is non-convex, and it is hard to compute the exact solution of $\frac{\partial Q}{\partial U_{w_m,t}} = 0$ and $\frac{\partial Q}{\partial V_{w,k}} = 0$. Therefore, we use gradient ascent to optimize in the M-step. The gradients of Q function w.r.t. embedding vectors are given by:

$$\frac{\partial Q}{\partial U_{w_m,t}} = \sum_{k=1}^{N_w} \hat{\gamma}_{m,k} b_t^{(w_m)} \big(1 - \varsigma(b_t^{(w_m)} U_{w_m,t}^T V_{w,k})\big) V_{w,k}, \tag{9}$$

$$\frac{\partial Q}{\partial V_{w,k}} = \sum_{m=1}^{M} \hat{\gamma}_{m,k} \sum_{t=1}^{L_{w_m}} b_t^{(w_m)} \big(1 - \varsigma(b_t^{(w_m)} U_{w_m,t}^T V_{w,k})\big) U_{w_m,t}. \tag{10}$$

Iterating between **E-Step** and **M-Step** till the convergence of the value of function Q makes the EM algorithm complete.

In order to enhance the scalability of our approach, we propose a fast computing method to boost the implementation of the EM algorithm. Note that the most expensive computing operations in both the E-Step and M-Step are the inner product of the input and output embedding vectors, as well as the sigmoid function. However, if we take the Hierarchical Softmax Tree form as shown in Equation (4) to model $P(w_m|h_m = i, w)$, and perform only one step gradient ascent in M-Step, the aforementioned two expensive operations in M-Step will be avoided by leveraging the pre-computed results in the E-Step. Specifically, since the gradient of the function $f(x) = \log \varsigma(x)$ is given by $f'(x) = 1 - \varsigma(x)$, the sigmoid values computed in the E-Step to obtain $P(w_m|h_m = i, w)$ (i.e. the term $\varsigma(b_t^{(w_m)} U_{w_m,t}^T V_{w,k})$ in equation (5), (9), and (10)) can be re-used to derive the gradients in the M-Step.

However, such enhanced computation method cannot benefit the second order optimization methods in the M-Step such as L-BFGS and Conjugate Gradient, since they usually rely on multiple iterations to converge. In fact, we tried these two optimization methods in our experiments but they have brought no improvement compared with simple one-step gradient ascent method.

3.3 Model Comparison

To show that our model is more scalable than the former multi-prototype model in (Huang et al., 2012) (We denote it as EHModel in the rest of the paper), we conduct a comparison on the number of parameters with respect to each of these two models in this subsection.

We use $n_{embedding}$ and n_{window} to denote the numbers of all word embedding vectors and context window words, respectively. It is clear that $n_{embeddings} = \sum_{w \in W} N_w$. EHModel aims to compute two scores, i.e., the local score and the global score, both with hidden layer node activations. We denote the hidden layer node number as h_l and h_g for these two scores. The parameter numbers are listed in Table 1.

Model	EHModel	Our Model
#parameters	$dn_{words} + dn_{embeddings} + (dn_{window} + 1)h_l + (2d + 1)h_g$	$dn_{words} + dn_{embeddings}$

Table 1: Comparison of parameter numbers of two models

Note that d in Table 1 denotes the embedding vector size. It can be observed that EHModel has $(dn_{window} + 1)h_l + (2d + 1)h_g$ more parameters than our model, which is mainly because EHModel has one more layer in the neural network and it considers global context. In previous study (Huang et al., 2012), d, n_{window}, h_l, and h_g are set to be 50, 10, 100, 100, respectively, which greatly increases the gap of parameter numbers between the two models.

4 Experiments

In this section, we will present our experimental settings and results. Particularly, we first describe the data collection and the training configuration we used in the experiments; then, we conduct a qualitative case study followed by quantitative evaluation results on a public word similarity task to demonstrate the performance of our proposed model.

4.1 Experimental Setup

Dataset: To make a fair comparison with the state-of-the-art methods, we employ a publicly available dataset, which is used in (Huang et al., 2012), to train word embeddings in our experiments. Particularly, this training corpus is a snapshot of Wikipedia at April, 2010 (Shaoul, 2010), which contains about 990 million tokens. We removed the infrequent words from this corpus and kept a dictionary of about 1 million most frequent words. Similar to Word2Vec, we removed pure digit words such as *2014* as well as about 100 stop words like *how*, *for*, and *we*.

Training Configuration: In order to boost the training speed, we take advantage of the Hierarchical Softmax Tree structure. More concretely, we use the Huffman tree structure, as introduced in Word2Vec, to further increase the training speed. All the embedding size, including both word embedding vectors and the Huffman tree node embedding vectors, are set to be 50, which is the same as the size used in (Huang et al., 2012). To train word embedding, we set the context window size as 10, i.e., for a word w, 10 of the closest neighboring words to w are regarded as ws contexts. For the numbers of word prototypes, i.e., N_w introduced in Section 3.2, we set the top 7 thousand frequent words as multi-prototype words by experience, with all of them having 10 prototypes (i.e. $N_w = 10$).

During the training process, we used the same strategy to set the learning rate as what Word2Vec did. Specifically, we set the initial learning rate to 0.025 and diminished the value linearly along with the increasing number of training words. Our experimental results illustrate that this learning rate strategy can lead to the best results for our algorithm.

For the hyper parameters of the EM algorithm, we set the batch size to 1, i.e. $M = 1$ in Section 3.2, since our experimental results reveal that smaller batch size can result in better experimental results. The reason is explained as the following. Our optimization problem is highly non-convex. Smaller batch size yields more frequent updates of parameters, and thus avoids trapping in local optima, while larger batch size, associated with more infrequent parameter updating, may cause higher probability to encounter local optima. In our experiments, we observe that only one iteration of E-Step and M-Step can reach the embedding vectors with good enough performance on the word similarity task, whereas increasing the iteration number just leads to slight performance improvement with much longer training time. Under the above configuration, our model runs about three times faster than EHModel.

4.2 Case Study

This section gives some qualitative evaluations of our model by demonstrating how our model can effectively identify multi-prototype word embeddings on some specific cases. In Table 2, we list several polysemous words. For each word, we pick some of their prototypes learned by our model, including the prototype prior probability (i.e. π_i introduced in Section 3.2) and three of the most similar words with each prototype, respectively. The similarity is calculated by the cosine similarity score between the embedding vectors.

From the table we can observe some interesting results of the multi-prototype embedding vectors produced by our model:

- For a polysemous word, its different embedding vectors represent its different semantic meanings. For example, the first embedding vector of the word *apple* corresponds to its sense as a kind of fruit, whereas the second one represents its meaning as an IT company.

- The prior probability reflects the likelihood of the occurrence of various prototypes to some extent. For example, the word *cell* is more likely to represent the meaning of the smallest part of living structure (with probability 0.81), than to be used as the meaning of *cellphone* (with probability

Word	Prior Probability	Most Similar Words
apple_1	0.82	strawberry, cherry, blueberry
apple_2	0.17	iphone, macintosh, microsoft
bank_1	0.15	river, canal, waterway
bank_2	0.6	citibank, jpmorgan, bancorp
bank_3	0.25	stock, exchange, banking
cell_1	0.09	phones, cellphones, mobile
cell_2	0.81	protein, tissues, lysis
cell_3	0.01	locked, escape, handcuffed

Table 2: Most similar words with different prototypes of the same word

0.09) or *prisoned* (with probability 0.01). Note that the three prior probability scores of *cell* do not sum to 1. The reason is that there are some other embeddings not presented in the table which are found to have high similarities with the three embeddings. We do not present them due to the space limitation.

- By setting the prototype number to a fairly large value (e.g. $N_w = 10$), the model tends to learn more fine-grained separations of the word's different meanings. For example, we can observe from Table 2 that the second and the third prototypes of the word *bank* seem similar to each other as both of them denote a financial concept. However, there are subtle differences between them: the second prototype represents concrete banks, such as citibank and jpmorgan, whereas the third one denotes what is done in the banks, since it is most similar to the words *stock*, *exchange*, and *banking*. We believe that such a fine-grained separation will bring more expressiveness to the multi-prototype word embeddings learned by our model.

4.3 Results on Word Similarity in Context Dataset

In this subsection, we give quantitative comparison of our method with conventional word embedding models, including Word2Vec and EHModel (Huang et al., 2012).

The task we perform is the word similarity evaluation introduced in (Huang et al., 2012). Word similarity tasks evaluate a model's performance by calculating the Spearman's rank correlation between the ranking of ground truth similarity scores (given by human labeling) and the ranking based on the similarity scores produced by the model. Traditional word similarity tasks such as WordSim353 (Finkelstein et al., 2001) and RG (Rubenstein and Goodenough, 1965) are not suitable for evaluating multi-prototype models since there is neither enough number of polysemous words in these datasets nor context information to infer the prototype index. To address this issue, a new word similarity benchmark dataset including context information was released in (Huang et al., 2012). Following (Luong et al., 2013), we use SCWS to denote this dataset. Similar to WordSim353, SCWS contains some word pairs (concretely, 2003 pairs), together with human labeled similarity scores for these word pairs. What makes SCWS different from WS353 is that the words in SCWS are contained in sentences, i.e., there are 2003 pairs of sentences containing these words, while words in WS353 are not associated with sentences. Therefore, the human labeled scores are based on the meanings of the words in the context. Given the presence of the context, the word similarity scores, especially those scores depending on polysemous words, are much more convincing for evaluating different models' performance in our experiments.

Then, we propose a method to compute the similarity score for a pair of words $\{w_1, w_2\}$ in the context based on our model. Suppose that the context of a word w is defined as all its neighboring words in a $T+1$ sized window, where w is the central word in the window. We use $Context_1 = \{c_1^1, c_2^1, \cdots, c_T^1\}$ and $Context_2 = \{c_1^2, c_2^2, \cdots, c_T^2\}$ to separately denote the context of w_1 and w_2, where c_t^1 and c_t^2 are the t-th context word of w_1 and w_2, respectively. According to Bayesian rule, we have that for $i \in \{1, 2, \cdots, N_{w_1}\}$:

$$\begin{aligned} P(h_{w_1} = i | Context_1, w_1) &\propto P(Context_1 | h_{w_1} = i, w_1) P(h_{w_1} = i | w_1) \\ &= \prod_{t=1}^{T} P(c_t^1 | h_{w_1} = i, w_1) P(h_{w_1} = i | w_1), \end{aligned} \tag{11}$$

where $P(c_t^1|h_{w_1}=i,w_1)$ can be calculated by equation (4) and $P(h_{w_1}=i|w_1)$ is the prior probability we learned in the EM algorithm (equation (8)). The similar equation holds for word w_2 as well. Here we make an assumption that the context words are independent with each other given the central word. Furthermore, suppose that the most likely prototype index for w_1 given $Context_1$ is $\hat{h}_{w_1}$, i.e., we denote $\hat{h}_{w_1} = \arg\max_{i\in\{1,2,\cdots,N_{w_1}\}} P(h_{w_1}=i|Context_1,w_1)$. Similarly, $\hat{h}_{w_2}$ is denoted as the corresponding meaning for w_2.

We calculate two similarity scores base on equation (11), i.e., MaxSim Score and WeightedSim Score:

$$MaxSim(w_1,w_2) = Cosine(V_{w_1,\hat{h}_{w_1}}, V_{w_2,\hat{h}_{w_2}}), \tag{12}$$

$$WeightedSim(w_1,w_2) = \sum_{i=1}^{N_{w_1}}\sum_{j=1}^{N_{w_2}} P(h_{w_1}=i|Context_1,w_1)P(h_{w_2}=j|Context_2,w_2)Cosine(V_{w_1,i},V_{w_2,j}). \tag{13}$$

In the above similarity scores, $Cosine(x,y)$ denotes the cosine similarity score of vector x and y, and $V_{w,i} \in R^d$ is the embedding vector for the word w's i-th prototype.

The detailed experimental results are listed in Table 3, where ρ refers to the Spearman's rank correlation. The higher value of ρ indicates the better performance. The performance score of EHModel is borrowed from its original paper (Huang et al., 2012). For Word2Vec model, we use Hierarchical Huffman Tree rather than Negative Sampling to do the acceleration. Our **Model_M** uses the MaxSim score in testing and our **Model_W** uses the WeightedSim score. All of these models are run on the same aforementioned Wikipedia corpus, with the dimension of the embedding space to be 50.

From the table, we can observe that our **Model_W** (65.4%) outperforms the original Word2Vec model (61.7%), and achieves almost the same performance with the state-of-the-art EHModel (65.7%). Among the two similarity measures used in testing, the WeightedSim score performs better (65.4%) than the MaxSim score (63.6%), indicating that the overall consideration of all prototype probabilities are more effective.

Model	$\rho \times 100$
Word2Vec	61.7
EHModel	65.7
Model_M	63.6
Model_W	65.4

Table 3: Spearman's rank correlations on SCWS dataset.

5 Conclusion

In this paper, we introduce a fast and probabilistic method to generate multiple embedding vectors for polysemous words, based on the continuous Skip-Gram model. On one hand, our method addresses the drawbacks of the original Word2Vec model by leveraging multi-prototype word embeddings; on the other hand, our model yields much less complexity without performance loss compared with the former clustering based multi-prototype algorithms. In addition, the probabilistic framework of our method avoids the extra efforts to perform clustering besides training word embeddings.

For the future work, we plan to apply the proposed probabilistic framework to other neural network language models. Moreover, we would like to apply the multi-prototype embeddings to more real world text mining tasks, such as information retrieval and knowledge mining, with the expectation that the multi-prototype embeddings produced by our model will benefit these tasks.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. In *Journal of Machine Learning Research*, pages 1137–1155.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 897–906, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. *CoNLL-2013*, 104.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2265–2273.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *Proceedings of the international workshop on artificial intelligence and statistics*, pages 246–252.

Siva Reddy, Ioannis P Klapaftis, Diana McCarthy, and Suresh Manandhar. 2011. Dynamic and static prototype vectors for semantic composition. In *IJCNLP*, pages 705–713.

Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Westbury C Shaoul, C. 2010. The westbury lab wikipedia corpus.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *IJCNLP*, pages 1134–1143.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. 2011. Latent vector weighting for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1012–1022, Stroudsburg, PA, USA. Association for Computational Linguistics.

6 Appendix

6.1 Derivations for the EM Algorithm

We give detailed derivations for the updating rules used in the EM algorithms in Section 3.2., i.e., the derivations for equation (6), (7), and (8).

According to the properties of conditional probability, we have

$$\begin{aligned}\hat{\gamma}_{m,k} = P(\gamma_{m,k}=1|\mathbb{X},\Theta) &= \frac{P(\gamma_{m,k}=1,\mathbb{X}|\Theta)}{\sum_{i=1}^{N_w} P(\gamma_{m,i}=1,\mathbb{X}|\Theta)} \\ &= \frac{P(\gamma_{m,k}=1|\Theta)P(\mathbb{X}|\gamma_{m,k}=1,\Theta)}{\sum_{i=1}^{N_w} P(\gamma_{m,i}=1|\Theta)P(\mathbb{X}|\gamma_{m,i}=1,\Theta)} \\ &= \frac{\pi_k P(w_m|h_m=k,w)}{\sum_{i=1}^{N_w}\pi_i P(w_m|h_m=i,w)}.\end{aligned} \tag{14}$$

From equation (7), the Q function is calculated as:

$$\begin{aligned}Q(\theta,\theta^{(i)}) &= E[\log P(\mathbb{X},\Gamma|\Theta)|\Theta^{(i)}] \\ &= \sum_{k=1}^{N_w}\sum_{m=1}^{M} E[\gamma_{m,k}|\Theta^{(i)}]\big(\log\pi_k + \sum_{t=1}^{L_{w_m}} \log\varsigma(b_t^{(w_m)} U_{w_m,t}^T V_{w,k})\big) \\ &= \sum_{k=1}^{N_w}\sum_{m=1}^{M} \hat{\gamma}_{m,k}\big(\log\pi_k + \sum_{t=1}^{L_{w_m}} \log\varsigma(b_t^{(w_m)} U_{w_m,t}^T V_{w,k})\big) \\ &= \sum_{m=1}^{M}\sum_{k=1}^{N_w} \hat{\gamma}_{m,k}\big(\log\pi_k + \sum_{t=1}^{L_{w_m}} \log\varsigma(b_t^{(w_m)} U_{w_m,t}^T V_{w,k})\big).\end{aligned} \tag{15}$$

Then we give the derivations for π's updating rule, i.e., equation (8). Note that for parameters π_k, $k=\{1,2,\cdots,N_w\}$, they need to satisfy the condition that $\sum_{k=1}^{N_w}\pi_k = 1$. From equation (7) (or equivalently equation (15)), the loss with regard to π is:

$$L_{[\pi]} = \sum_{m=1}^{M}\sum_{k=1}^{N_w} \hat{\gamma}_{m,k}\log\pi_k + \lambda\big(\sum_{k=1}^{N_w}\pi_k - 1\big), \tag{16}$$

where λ is the Language multiplier. Letting $\frac{\partial L_{[\pi]}}{\partial \pi} = 0$, we obtain:

$$\pi_k \propto \sum_{m=1}^{M} \hat{\gamma}_{m,k}. \tag{17}$$

Further considering the fact that $\sum_{k=1}^{N_w}\sum_{m=1}^{M}\hat{\gamma}_{m,k} = M$, we have $\pi_k = \frac{\sum_{m=1}^{M}\hat{\gamma}_{m,k}}{M}$.

Learning Task-specific Bilexical Embeddings

Pranava Swaroop Madhyastha **Xavier Carreras** **Ariadna Quattoni**
TALP Research Center
Universitat Politècnica de Catalunya
Campus Nord UPC, Barcelona
pranava,carreras,aquattoni@lsi.upc.edu

Abstract

We present a method that learns bilexical operators over distributional representations of words and leverages supervised data for a linguistic relation. The learning algorithm exploits low-rank bilinear forms and induces low-dimensional embeddings of the lexical space tailored for the target linguistic relation. An advantage of imposing low-rank constraints is that prediction is expressed as the inner-product between low-dimensional embeddings, which can have great computational benefits. In experiments with multiple linguistic bilexical relations we show that our method effectively learns using embeddings of a few dimensions.

1 Introduction

We address the task of learning functions that compute compatibility scores between pairs of lexical items under some linguistic relation. We refer to these functions as bilexical operators. As an instance of this problem, consider learning a model that predicts the probability that an adjective modifies a noun in a sentence. In this case, we would like the bilexical operator to capture the fact that some adjectives are more compatible with some nouns than others. For example, a bilexical operator should predict that the adjective *electronic* has high probability of modifying the noun *device* but little probability of modifying the noun *case*.

Bilexical operators can be useful for multiple NLP applications. For example, they can be used to reduce ambiguity in a parsing task. Consider the following sentence extracted from a weblog: *Vynil can be applied to electronic devices and cases, wooden doors and furniture and walls*. If we want to predict the dependency structure of this sentence we need to make several decisions. In particular, the parser would need to decide (1) Does *electronic* modify *devices*? (2) Does *electronic* modify *cases*? (3) Does *wooden* modify *doors*? (4) Does *wooden* modify *furniture*? Now imagine that in the corpus used to train the parser none of these nouns have been observed, then it is unlikely that these attachments can be resolved correctly. However, if an accurate noun-adjective bilexical operator were available most of the uncertainty could be resolved. This is because a good bilinear operator would give high probability to the pairs *electronic-device*, *wooden-door*, *wooden-furniture* and low probability to the pair *electronic-case*.

The simplest way of inducing a bilexical operator is to learn it from a training corpus. That is, assuming that we are given some data annotated with a linguistic relation between a modifier and a head (e.g. adjective and noun) we can simply build a maximum likelihood estimator for $\Pr(m \mid h)$ by counting the occurrences of modifiers and heads under the target relation. For example, we could consider learning bilexical operators from sentences annotated with dependency structures. Clearly, this model can not generalize to head words not present in the training data.

To mitigate this we could consider bilexical operators that can exploit lexical embeddings, such as a distributional vector-space representation of words. In this case, we assume that for every word we can compute an n-dimensional vector space representation $\phi(w) \rightarrow \mathbb{R}^n$. This representation typically captures distributional features of the context in which the lexical item can occur. The key point is that

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 161–171, Dublin, Ireland, August 23-29 2014.

we do not need a supervised corpus to compute the representation. All we need is a large textual corpus to compute the relevant statistics. Once we have the representation we can exploit operations in the induced vector space to define lexical compatibility operators. For example we could define a bilexical operator as:

$$\Pr(m \mid h) = \frac{\exp\{\langle \phi(m), \phi(h) \rangle\}}{\sum_{m'} \exp\{\langle \phi(m'), \phi(h) \rangle\}} \quad (1)$$

where $\langle \phi(x), \phi(y) \rangle$ denotes the inner-product. Alternatively, given an initial high-dimensional distributional representation computed from a large textual corpus we could first induce a projection to a lower k dimensional space by performing truncated singular value decomposition. The idea is that the lower dimensional representation will be more efficient and it will better capture the relevant dimensions of the distributional representation. The bilexical operator would then take the form of:

$$\Pr(m|h) = \frac{\exp\{\langle U\phi(m), U\phi(h) \rangle\}}{\sum_{m'} \exp\{\langle U\phi(m'), U\phi(h) \rangle\}} \quad (2)$$

where $U \in \mathbb{R}^{k \times n}$ is the projection matrix obtained via SVD. The advantage of this approach is that as long as we can estimate the distribution of contexts of words we can compute the value of the bilexical operator. However, this approach has a clear limitation: to design a bilinear operator for a target linguistic relation we must design the appropriate distributional representation. Moreover, there is no clear way of exploiting a supervised training corpus.

In this paper we combine both the supervised and distributional approaches and present a learning algorithm for inducing bilexical operators from a combination of supervised and unsupervised training data. The main idea is to define bilexical operators using bilinear forms over distributional representations: $\phi(x)^\top W \phi(y)$, where $W \in \mathbb{R}^{n \times n}$ is a matrix of parameters. We can then train our model on the supervised training corpus via conditional maximum-likelihood estimation. To induce a low-dimensional representation, we first observe that the implicit dimensionality of the bilinear form is given by the rank of W. In practice controlling the rank of W can result in important computational savings in cases where one evaluates a target word x against a large number of candidate words y: this is because we can project the representations $\phi(x)$ and $\phi(y)$ down to the low-dimensional space where evaluating the function is simply an inner-product. This setting is in fact usual, for example for lexical retrieval applications (e.g. given a noun, sort all adjectives in the vocabulary according to their compatibility), or for parsing (where one typically evaluates the compatibility between all pairs of words in a sentence).

Consequently with these ideas, we propose to regularize the maximum-likelihood estimation using a nuclear norm regularizer that serves as a convex relaxation to the rank function. To minimize the regularized objective we make use of an efficient iterative proximal method that involves computing the gradient of the function and performing singular value decompositions.

We test the proposed algorithm on several linguistic relations and show that it can predict modifiers for unknown words more accurately than the unsupervised approach. Furthermore, we compare different types of regularizers for the bilexical operator W, and observe that indeed the low-rank regularizer results in the most efficient technique at prediction time.

In summary, the main contributions of this paper are:

- We propose a supervised framework for learning bilexical operators over distributional representations, based on learning bilinear forms W.
- We show that we can obtain low-dimensional compressions of the distributional representation by imposing low-rank constraints to the bilinear form. Combined with supervision, this results in lexical embeddings tailored for a specific bilexical task.
- In experiments, we show that our models generalize well to unseen word pairs, using only a few dimensions, and outperforming standard unsupervised distributional approaches. We also present an application to prepositional phrase attachment.

2 Bilinear Models for Bilexical Predictions

2.1 Definitions

Let $\mathcal{V}$ be a vocabulary, and let $x \in \mathcal{V}$ denote a word. Let $\mathcal{H} \subseteq \mathcal{V}$ be a set of head words, and $\mathcal{M} \subseteq \mathcal{V}$ be a set of modifier words. In the noun-adjective relation example, $\mathcal{H}$ is the set of nouns and $\mathcal{M}$ is the set of adjectives.

The task is as follows. We are given a training set of l tuples $\mathcal{D} = \{(m,h)^1, \ldots, (m,h)^l\}$, where $m \in \mathcal{M}$ and $h \in \mathcal{H}$ and we want to learn a model of the conditional distribution $\Pr(m \mid h)$. We want this model to perform well on all head-modifier pairs. In particular we will test the performance of the model on heads that do not appear in $\mathcal{D}$.

We assume that we are given access to a distributional representation function $\phi : \mathcal{V} \rightarrow \mathbb{R}^n$, where $\phi(x)$ is the n-dimensional representation of x. Typically, this function is computed from an unsupervised corpus. We use $\phi(x)_{[i]}$ to refer to the i-th coordinate of the vector.

2.2 Bilinear Model

Our model makes use of the bilinear form $W : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$, where $W \in \mathbb{R}^{n \times n}$, and evaluates as $\phi(m)^\top W \phi(h)$. We define the bilexical operator as:

$$\Pr(m \mid h) = \frac{\exp\left\{\phi(m)^\top W \phi(h)\right\}}{\sum_{m' \in \mathcal{M}} \exp\left\{\phi(m')^\top W \phi(h)\right\}} \tag{3}$$

Note that the above model is nothing more than a conditional log-linear model defined over n^2 features $f_{i,j}(m,h) = \phi(m)_{[i]}\phi(h)_{[j]}$ (this can be seen clearly when we write the bilinear form as $\sum_{i=1}^n \sum_{j=1}^n f_{i,j}(m,h) W_{i,j}$. The reason why it is useful to regard W as a matrix will become evident in the next section.

Before moving to the next section, let us note that the unsupervised SVD model in Eq. (2) is also a bilinear model as defined here. This can be seen if we set $W = UU^\top$, which is a bilinear form of rank k. The key difference is in the way W is learned using supervision.

3 Learning Low-rank Bilexical Operators

3.1 Low-rank Optimization

Given a training set $\mathcal{D}$ and a feature function $\phi(x)$ we can do standard conditional max-likelihood optimization and minimize the negative of the log-likelihood function, $\log \Pr(\mathcal{D})$:

$$\sum_{(m,h) \in \mathcal{D}} \phi(m)^\top W \phi(h) - \log \sum_{m' \in \mathcal{M}} \exp\left\{\phi(m')^\top W \phi(h)\right\} \tag{4}$$

We would like to control the complexity of the learned model by including some regularization penalty. Moreover, like in the low-dimensional unsupervised approach we want our model to induce a low-dimensional representation of the lexical space. The first observation is that the bilinear form computes a weighted inner product in some space. Consider the singular value decomposition: $W = U\Sigma V$. We can write the bilinear form as: $[\phi(m)^\top U]\ \Sigma\ [V\phi(h)]$, thus we can regard $\tilde{m} = \phi(m)^\top U$ as a projection of m and $\tilde{h} = V\phi(h)$ as a projection of h. Then the bilinear form can be written as: $\sum_{i=1}^n \Sigma_{[i,i]}\ \tilde{m}_{[i]}\ \tilde{h}_{[i]}$. The rank of W defines the dimensionality of the induced space. It is easy to see that if W has rank k it can be factorized as $U\Sigma V$ where $U \in \mathbb{R}^{n \times k}$ and $V \in \mathbb{R}^{k \times n}$.

Since the rank of W determines the dimensionality of the induced space, it would be reasonable to add a rank minimization penalty in the objective in (4). Unfortunately this would lead to a non-convex regularized objective. Instead, we propose to use as a regularizer a convex relaxation of the rank function, the nuclear norm $\|W\|_*$ (the ℓ_1 norm of the singular values of W). Putting it all together, our learning algorithm minimizes:

$$\sum_{(m,h) \in \mathcal{D}} -\log \Pr(m \mid h)) + \lambda \|W\|_* \tag{5}$$

Here λ is a constant that controls the trade-off between fitting the data and the complexity of the model. This objective is clearly convex since both the objective and the regularizer are convex. To minimize it we use the a proximal gradient algorithm which is described next.

3.2 A Proximal Algorithm for Bilexical Operators

We now describe the learning algorithm that we use to induce the bilexical operators from training data. We are interested in minimizing the objective (5), or in fact a more general version where we can replace the regularizer $\|W\|_*$ by standard ℓ_1 or ℓ_2 penalties. For any convex regularizer $r(W)$ (namely ℓ_1, ℓ_2 or the nuclear norm) the objective in (5) is convex. Our learning algorithm is based on a simple optimization scheme known as *forward-backward splitting (FOBOS)* (Duchi and Singer, 2009).

This algorithm has convergence rates in the order of $1/\epsilon^2$, which we found sufficiently fast for our application. Many other optimization approaches are possible, for example one could express the regularizer as a convex constraint and utilize a projected gradient method which has a similar convergence rate. Proximal methods are slightly more simple to implement and we chose the proximal approach.

The FOBOS algorithm works as follows. In a series of iterations $t = 1 \dots T$ compute parameter matrices W_t as follows:

1. Compute the gradient of the negative log-likelihood, and update the parameters

$$W_{t+0.5} = W_t - \eta_t g(W_t)$$

   where $\eta_t = \frac{c}{\sqrt{t}}$ is a step size and $g(W_t)$ is the gradient of the loss at W_t.

2. Update $W_{t+0.5}$ to take into account the regularization penalty $r(W)$, by solving

$$W_{t+1} = \operatorname*{argmin}_W ||W_{t+0.5} - W||_2^2 + \eta_t \lambda r(W)$$

   For the regularizers we consider, this step is solved using the *proximal operator* associated with the regularizer. Specifically:

   - For ℓ_1 it is a simple thresholding:

$$W_{t+1}(i,j) = \text{sign}(W_{t+0.5}(i,j)) \cdot \max(W_{t+0.5}(i,j) - \eta_t \lambda, 0)$$

   - For ℓ_2 it is a simple scaling:

$$W_{t+1} = \frac{1}{1 + \eta_t \lambda} W_{t+0.5}$$

   - For nuclear-norm, perform SVD thresholding. Compute the SVD to write $W_{t+0.5} = USV^\top$ with S a diagonal matrix and U, V orthogonal matrices. Denote by σ_i the i-th element on the diagonal of S. Define a new matrix $\bar{S}$ with diagonal elements $\bar{\sigma}_i = \max(\sigma_i - \eta_t \lambda, 0)$. Then set

$$W_{t+1} = U\bar{S}V^\top$$

Optimizing a bilinear model using nuclear-norm regularization involves the extra cost of performing SVD of W at each iteration. In our experiments the dimension of W was $2,000 \times 2,000$ and computing SVD was fast, much faster than computing the gradient, which dominates the cost of the algorithm. The optimization parameters of the method are the regularization constant λ, the step size constant c and the number of iterations T. In our experiments we ran a range of λ and c values for 200 iterations, and used a validation set to pick the best configuration.

4 Related Work

Research in learning representations for natural language processing can be broadly classified into two different paradigms based on the learning setting: unsupervised representation learning and semi-supervised representation learning. Unsupervised representation learning does not require any supervised training data, while semi-supervised representation learning requires the presence of supervised training data with the potential advantage that it can adapt the representation to the task at hand.

Unsupervised approaches to learning representations mainly involve representations that are learned not for a specific task, rather a variety of tasks. These representations rely more on the property of abstractness and generalization. Further, unsupervised approaches can be roughly categorized into (a) clustering-based approaches that make use of clusters induced using a notion of distributed similarity, such as the method by Brown et al. (1992); (b) neural-network-based representations that focus on learning multilayer neural network in a way to extract features from the data (Morin and Bengio, 2005; Mnih and Hinton, 2007; Bengio and Sénécal, 2008; Mnih and Hinton, 2009); (c) pure distributional approaches that principally follow the distributional assumption that the words which share a set of contexts are similar (Sahlgren, 2006; Turney and Pantel, 2010; Dumais et al., 1988; Landauer et al., 1998; Lund et al., 1995; Väyrynen et al., 2007).

We also induce lexical embeddings, but in our case we employ supervision. That is, we follow a semi-supervised paradigm for learning representations. Semi-supervised approaches initially learn representations typically in an unsupervised setting and then induce a representation that is jointly learned for the task with a labeled corpus. A high-dimensional representation is extracted from unlabeled data, while the supervised step compresses the representation to be low-dimensional in a way that favors the the task at hand.

Collobert and Weston (2008) present a neural network language model, where given a sentence, it performs a set of language processing tasks (from part of speech tagging, chunking, extracting named entity, extracting semantic roles and decisions on the correctness of the sentence) by using the learned representations. The representation itself is extracted from unlabeled corpora, while all the other tasks are jointly trained on labeled corpus.

Socher et al. (2011) present a model based on recursive neural networks that learns vector space representations for words, multi-word phrases and sentences. Given a sentence with its syntactic structure, their model assings vector representations to each of the lexical tokens of the sentence, and then traverses the syntactic tree bottom-up, such that at each node a vector representation of the corresponding phrase is obtained by composing the vectors associated with the children.

Bai et al. (2010) use a technique similar to ours, using bilinear forms with low-rank constraints. In their case, they explicitly look for a low-rank factorization of the matrix, making their optimization non-convex. As far as we know, ours is the first convex formulation, where we employ a relaxation of the rank (i.e. the nuclear norm) to make the objective convex. They apply the method to document ranking, and thus optimize a max-margin ranking loss. In our application to bilexical models, we perform conditional max-likelihood estimation. Hutchinson et al. (2013) propose an explicitly sparse and low-rank maximum-entropy language model. The sparse plus low rank setting is learned in such a way that the low rank component learns the regularities in the training data and the sparse component learns the exceptions like multiword expressions etc.

Chechik et al. (2010) also learned bilinear operators using max-margin techniques, with pairwise similarity as supervision, but they did not consider low-rank constraints.

One related area where bilinear operators are used to induce embeddings is distance metric learning. Weinberger and Saul (2009) used large-margin nearest neighbor methods to learn a non-sparse embedding, but these are computationally intensive and might not be suitable for large-scale tasks in NLP.

5 Experiments on Syntactic Relations

We conducted a set of experiments to test the ability of our algorithm to learn bilexical operators for several linguistic relations. As supervised training data we use the gold standard dependencies of the WSJ training section of the Penn Treebank (Marcus et al., 1993). We consider the following relations:

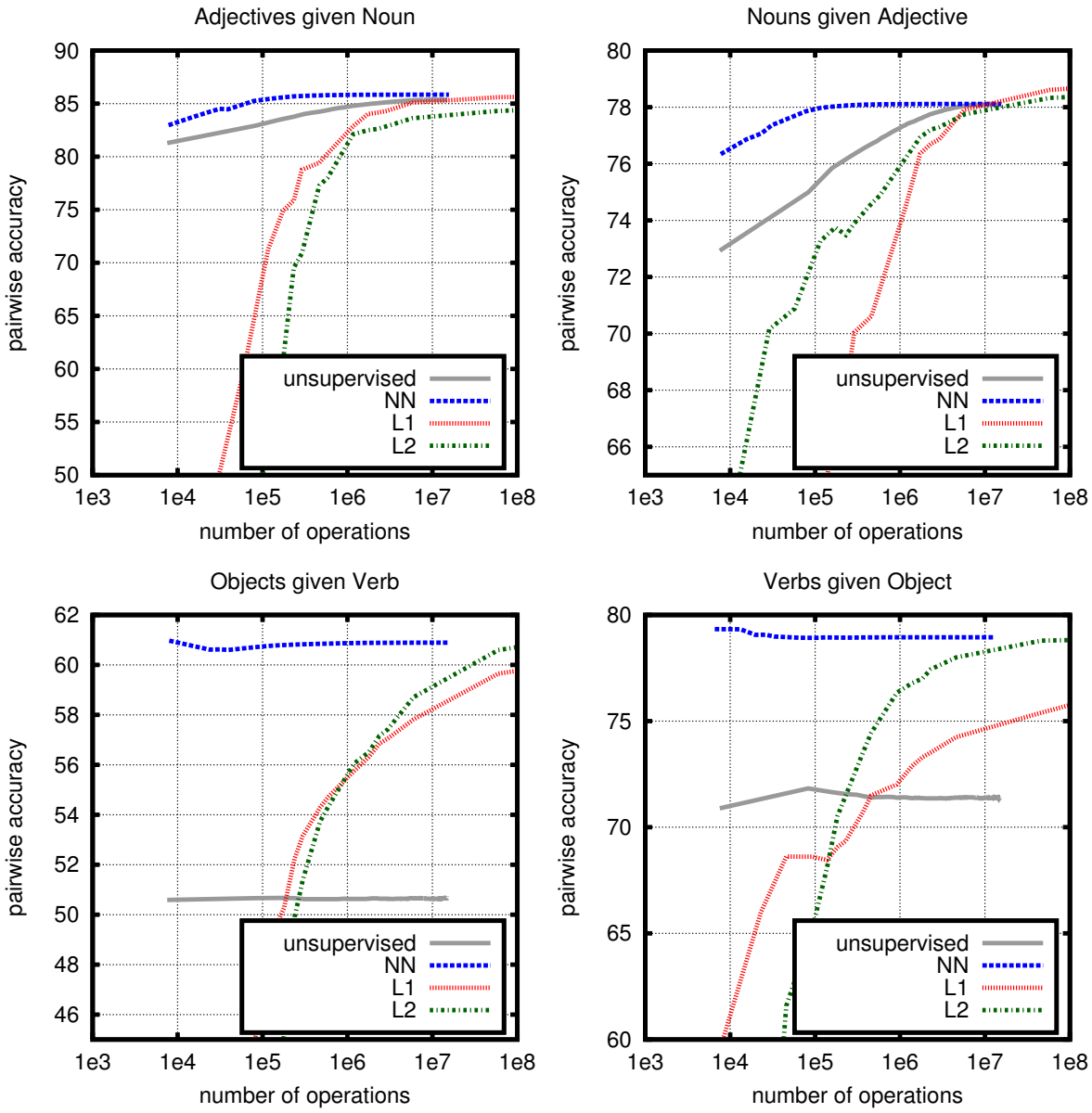


Figure 1: Pairwise accuracy with respect to the number of double operations required to compute the distribution over modifiers for a head word. Plots for noun-adjective and verb-object relations, in both directions.

- Noun-Adjective: we model the distribution of adjectives given a noun; and a separate distribution of nouns given an adjective.
- Verb-Object: we model the distribution of object nouns given a verb; and a separate distribution of verbs given an object.
- Prepositions: in this case we consider bilexical operators associated with a preposition, which model the probability of a head noun or verb above the preposition *given* the noun below the preposition. We present results for prepositional relations given by "with", "for", "in" and "on".

The distributional representation $\phi(x)$ was computed using the BLLIP corpus (Charniak et al., 2000). We compute a bag-of-words representation for the context of each lexical item, that is $\phi(w)_{[i]}$ corresponds to the frequency of word i appearing in the context of w. We use a context window of size 10 and restrict our bag-of-words vocabulary to contain only the 2,000 most frequent words present in the corpus. Vectors were normalized.

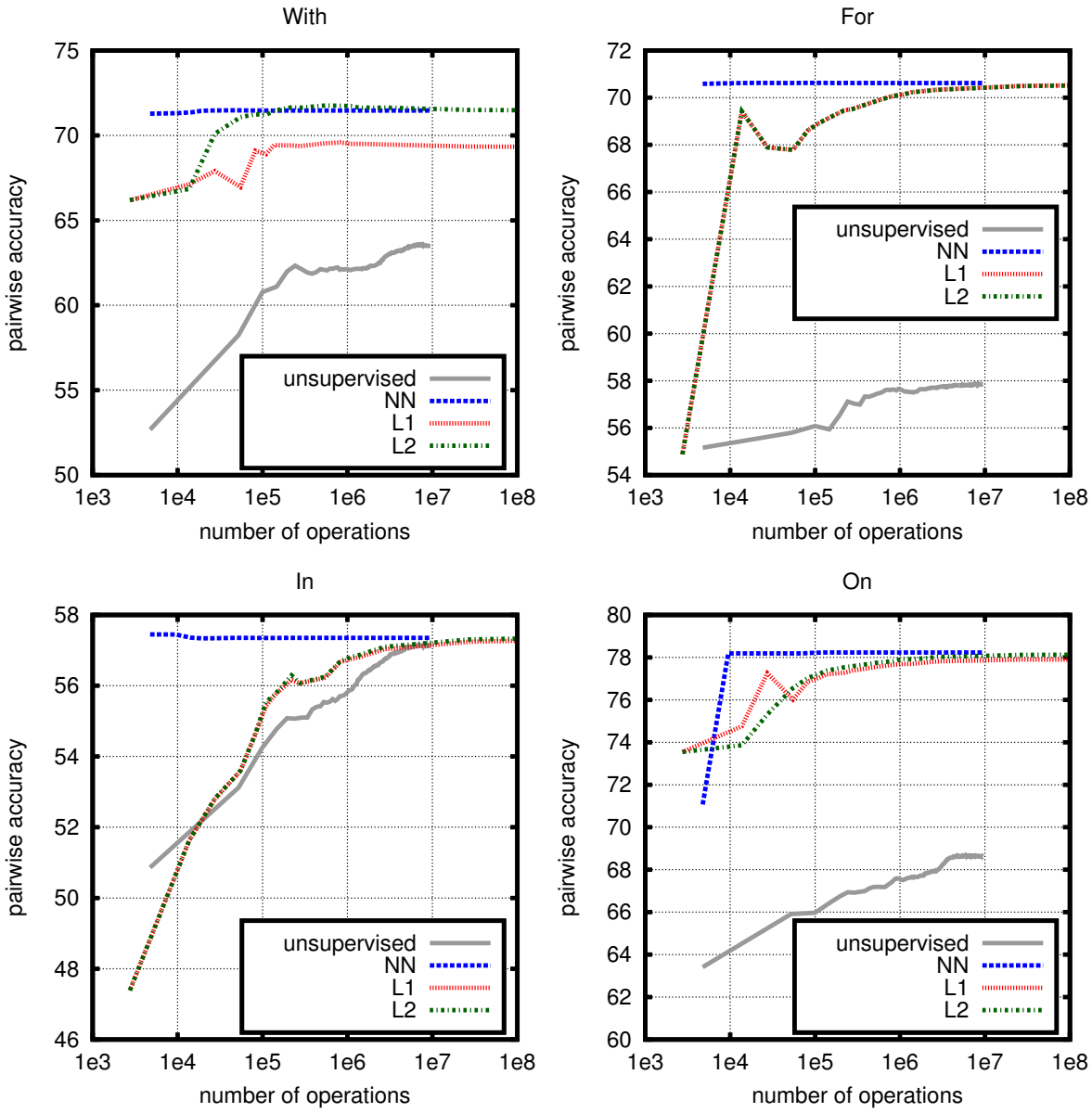


Figure 2: Pairwise accuracy with respect to the number of double operations required to compute the distribution over modifiers for a head word. Plots for four prepositional relations: with, for, in, on. The distributions are of verbs and objects above the preposition given the noun below the preposition.

To test the performance of our algorithm for each relation we partition the set of heads into a training and a test set, 60% of the heads are use for training, 10% of the heads are used for validation and 30% of the heads are used for testing. Then, we consider all observed modifiers in the data to form a vocabulary of modifier words. The goal of this task is to learn conditional distribution over all these modifers given a head word without context. In our experiments, the number of modifiers per relation ranges from 2,500 to 7,500 words. For each head word, we create a list of *compatible* modifiers from the annotated data, by taking all modifiers that occur at least once with the head. Hence, for each head the set of all modifiers is partitioned into compatible and non-compatible. For testing, we measure a *pairwise accuracy*, the percentage of compatible/non-compatible pairs of modifiers where the former obtains higher probability. Let us stress that none of the test head words has been observed in training, while the list of modifiers is the same for training, validation and testing.

We compare the performance of the bilexical model trained with nuclear norm regularization (NN) with other regularization penalties (L1 and L2). We also compare these supervised methods with an

Noun	Predicted Adjectives
president	executive, senior, chief, frank, former, international, marketing, assistant, annual, financial
wife	former, executive, new, financial, own, senior, old, other, deputy, major
shares	annual, due, net, convertible, average, new, high-yield, initial, tax-exempt, subordinated
mortgages	annualized, annual, three-month, one-year, average, six-month, conventional, short-term, higher, lower
month	last, next, fiscal, first, past, latest, early, previous, new, current
problem	new, good, major, tough, bad, big, first, financial, long, federal
holiday	new, major, special, fourth-quarter, joint, quarterly, third-quarter, small, strong, own

Table 1: 10 most likely adjectives for some test nouns.

unsupervised model: a low-dimensional SVD model as in Eq. (2), which corresponds to an inner product as in Eq. (1) when all dimensions are considered.

To report performance, we measure pairwise accuracy with respect to the capacity of the model in terms of number of active parameters. To measure the capacity of a model we consider the number of double operations that are needed to compute, given a head, the scores for all modifiers in the vocabulary (we exclude the exponentiations and normalization needed to compute the distribution of modifiers given a head, since this is a constant cost for all the models we compare, and is not needed if we only want to rank modifiers). Recall that the dimension of $\phi(x)$ is n, and assume that there are m total modifiers in the vocabulary. In our experiments $n = 2,000$ and m ranges from $2,500$ to $7,500$. The correspondances with operations are:

- Assume that the L1 and L2 models have k non-zero weights in W. Then the number of operations to compute a distribution is km.
- Assume that the NN and the unsupervised models have rank k. We assume that the modifier vectors are alredy projected down to k dimensions. For a new head, one needs to project it and perform m inner products, hence the number of operations is $kn + km$.

Figure 1 shows the performance of models for noun-adjective and verb-object relations, while Figure 2 shows plots for prepositional relations. [1] The first observation is that supervised approaches outperform the unsupervised approach. In cases such as noun-adjetive relations the unsupervised approach performs close to the supervised approaches, suggesting that the pure distributional approach can sometimes work. But in most relations the improvement obtained by using supervision is very large. When comparing the type of regularizer, we see that if the capacity of the model is unrestricted (right part of the curves), all models tend to perform similarly. However, when restricting the size, the nuclear-norm model performs much better. Roughly, 20 hidden dimensions are enough to obtain the most accurate performances (which result in $\sim 140,000$ operations for initial representaions of $2,000$ dimensions and $5,000$ modifier candidates). As an example of the type of predictions, Table 1 shows the most likely adjectives for some test nouns.

6 Experiments on PP Attachment

We now switch to a standard classification task, prepositional phrase attachment, that we frame as a bilexical prediction task. We start from the formulation of the task as a binary classification problem by Ratnaparkhi et al. (1994): given a tuple $x = \langle v, o, p, n \rangle$ consisting of a verb v, noun object o, preposition

[1] To obtain curves for each model type with respect to a range of number of operations, we first obtained the best model on validation data and then forced it to have at most k non-zero features or rank k by projecting, for a range of k values.

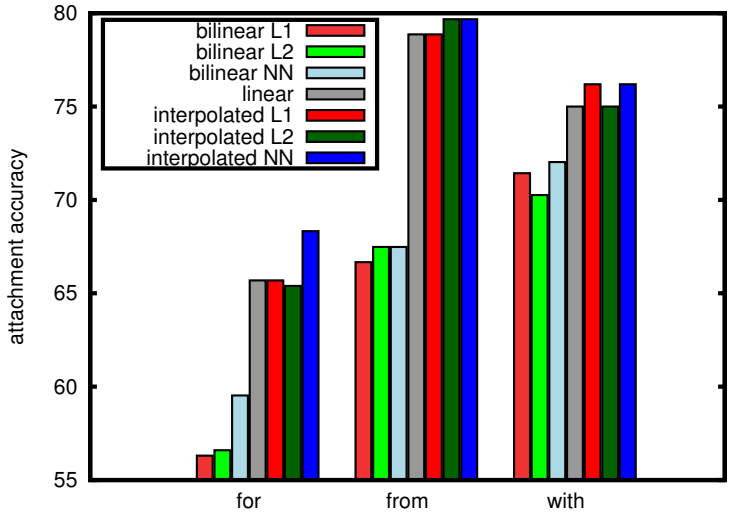


Figure 3: Attachment accuracies of linear, bilinear and interpolated models for three prepositions.

p and noun n, decide if the prepositional phrase p-n attaches to v ($y = \text{V}$) or to o ($y = \text{O}$). For example, in $\langle$ `meet,demand,for,products` $\rangle$ the correct attachment is O.

Ratnaparkhi et al. (1994) define a linear maximum likelihood model of the form $\Pr(y \mid x) = \exp\{\langle w, f(x,y)\rangle\} * Z(x)^{-1}$, where $f(x,y)$ is a vector of d features, w is a parameter vector in $\mathbb{R}^d$, and $Z(x)$ is the normalizer summing over $y = \{\text{V}, \text{O}\}$. Here we define a bilexical model of the form that uses a distributional representation ϕ:

$$\Pr(\text{V}|\langle v,o,p,n\rangle) = \frac{\exp\{\phi(v)^\top W^p_\text{V}\, \phi(n)\}}{Z(x)} \qquad \Pr(\text{O}|\langle v,o,p,n\rangle) = \frac{\exp\{\phi(o)^\top W^p_\text{O}\, \phi(n)\}}{Z(x)} \tag{6}$$

The bilinear model is parameterized by two matrices W_V and W_O per preposition, each of which captures the compatibility between nouns below a certain preposition and heads of V or O prepositional relations, respectively. Again $Z(x)$ is the normalizer summing over $y = \{\text{V}, \text{O}\}$, but now using the bilinear form. It is straighforward to modify the learning algorithm in Section 3 such that the loss is a negative log-likelihood for binary classification, and the regularizer considers the sum of norms of the model matrices.

We ran experiments using the data by Ratnaparkhi et al. (1994). We trained separate models for different prepositions, focusing on the prepositions that are more ambiguous: `for, from, with`. We compare to a linear "maxent" model following Ratnaparkhi et al. (1994) that uses the same feature set. Figure 3 shows the test results for the linear model, and bilinear models trained with L1, L2, NN regularization penalties. The results of the bilinear models are significantly below the accuracy of the linear model, suggesting that some of the non-lexical features of the linear model (such as prior weighting of the two classes) might be difficult to capture by the bilinear model over lexical representations. To check if the bilinear model might complement the linear model or just be worse than it, we tested simple combinations based on linear interpolations. For a constant $\lambda \in [0,1]$ we define:

$$\Pr(y \mid x) = \lambda \Pr_\text{L}(y \mid x) + (1-\lambda) \Pr_\text{B}(y \mid x) \qquad . \tag{7}$$

We search for the best λ on the validation set, and report results of combining the linear model with each of the three bilinear models. Results are shown also in Figure 3. Interpolation models improve over linear models, though only the improvement for `for` is significant (2.6%). Future work should exploit finer combinations between standard linear features and distributional bilinear forms.

7 Conclusions

We have presented a model for learning bilexical operators that can leverage both supervised and unsupervised data. The model is based on exploiting bilinear forms over distributional representations. The

learning algorithm induces a low-dimensional representation of the lexical space by imposing low-rank constraints on the parameters of the bilinear form. By means of supervision, our model induces two low-dimensional lexical embeddings, one on each side of the bilexical linguistic relation, and computations can be expressed as an inner-product between the two embeddings. This factorized form of the model can have great computational advantages: in many applications one needs to evaluate the function multiple times for a fixed set of lexical items, for example in dependency parsing. Hence, one can first project the lexical items to their embeddings, and then compute all pairwise scores as inner-products. In experiments, we have shown that the embeddings we obtain in a number of linguistic relations can be modeled with a few hidden dimensions.

As future work, we would like to apply the low-rank approach to other model forms that can employ lexical embeddings, specially when supervision is available. For example, dependency parsing models, or models of predicate-argument structures representing semantic roles, exploit bilexical relations. In these applications, being able to generalize to word *pairs* that are not observed during training is essential.

We would also like to study how to combine low-rank bilexical operators, which in essence induce a task-specific representation of words, with other forms of features that capture class or contextual information. One desires that such combinations can preserve the computational advantages behind low-rank embeddings.

Acknowledgements

We thank the reviewers for their helpful comments. This work was supported by projects XLike (FP7-288342), ERA-Net CHISTERA VISEN and TACARDI (TIN2012-38523-C02-00). Xavier Carreras was supported by the Ramón y Cajal program of the Spanish Government (RYC-2008-02223).

References

Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Weinberger. 2010. Learning to rank with (a lot of) word features. *Information Retrieval*, 13(3):291–314, June.

Yoshua Bengio and Jean-Sébastien Sénécal. 2008. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Transactions on Neural Networks*, 19(4):713–722.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Eugene Charniak, Don Blaheta, Niyu Ge, Keith Hall, and Mark Johnson. 2000. BLLIP 1987–89 WSJ Corpus Release 1, LDC No. LDC2000T43. Linguistic Data Consortium.

Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. 2010. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, pages 1109–1135.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA. ACM.

John Duchi and Yoram Singer. 2009. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934.

Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Scott Deerwester, and Richard Harshman. 1988. Using latent semantic analysis to improve access to textual information. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 281–285. ACM.

Brian Hutchinson, Mari Ostendorf, and Maryam Fazel. 2013. Exceptions in language as learned by the multi-factor sparse plus low-rank language model. In *ICASSP*, pages 8580–8584.

Thomas K. Landauer, Peter W. Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes*, 25:259–284.

Kevin Lund, Curt Burgess, and Ruth A. Atchley. 1995. Semantic and associative priming in high-dimensional semantic space. In *Cognitive Science Proceedings, LEA*, pages 660–665.

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

Andriy Mnih and Geoffrey E. Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th International Conference on Machine Learning*, pages 641–648.

Andriy Mnih and Geoffrey E. Hinton. 2009. A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems*, pages 1081–1088.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *AISTATS05*, pages 246–252.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the workshop on Human Language Technology*, HLT '94, pages 250–255, Stroudsburg, PA, USA. Association for Computational Linguistics.

Magnus Sahlgren. 2006. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Ph.D. thesis, Stockholm University.

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, January.

Jaakko J. Väyrynen, Timo Honkela, and Lasse Lindqvist. 2007. Towards explicit semantic features using independent component analysis. In *Proceedings of the Workshop Semantic Content Acquisition and Representation (SCAR)*, Stockholm, Sweden. Swedish Institute of Computer Science. SICS Technical Report T2007-06.

Kilian Q. Weinberger and Lawrence K. Saul. 2009. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, June.

Building Large-Scale Twitter-Specific Sentiment Lexicon : A Representation Learning Approach

Duyu Tang[♮*], **Furu Wei**[‡], **Bing Qin**[♮†], **Ming Zhou**[‡], **Ting Liu**[♮]
[♮]Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology, China
[‡]Microsoft Research, Beijing, China
{dytang, qinb, tliu}@ir.hit.edu.cn
{fuwei, mingzhou}@microsoft.com

Abstract

In this paper, we propose to build large-scale sentiment lexicon from Twitter with a representation learning approach. We cast sentiment lexicon learning as a phrase-level sentiment classification task. The challenges are developing effective feature representation of phrases and obtaining training data with minor manual annotations for building the sentiment classifier. Specifically, we develop a dedicated neural architecture and integrate the sentiment information of text (e.g. sentences or tweets) into its hybrid loss function for learning sentiment-specific phrase embedding (**SSPE**). The neural network is trained from massive tweets collected with positive and negative emoticons, without any manual annotation. Furthermore, we introduce the Urban Dictionary to expand a small number of sentiment seeds to obtain more training data for building the phrase-level sentiment classifier. We evaluate our sentiment lexicon (**TS-Lex**) by applying it in a supervised learning framework for Twitter sentiment classification. Experiment results on the benchmark dataset of SemEval 2013 show that, TS-Lex yields better performance than previously introduced sentiment lexicons.

1 Introduction

A sentiment lexicon is a list of words and phrases, such as "*excellent*", "*awful*" and "*not bad*", each of which is assigned with a positive or negative score reflecting its sentiment polarity and strength. Sentiment lexicon is crucial for sentiment analysis (or opining mining) as it provides rich sentiment information and forms the foundation of many sentiment analysis systems (Pang and Lee, 2008; Liu, 2012; Feldman, 2013). Existing sentiment lexicon learning algorithms mostly utilize propagation methods to estimate the sentiment score of each phrase. These methods typically employ parsing results, syntactic contexts or linguistic information from thesaurus (e.g. WordNet) to calculate the similarity between phrases. For example, Baccianella et al. (2010) use the glosses information from WordNet; Velikovich et al. (2010) represent each phrase with its context words from the web documents; Qiu et al. (2011) exploit the dependency relations between sentiment words and aspect words. However, parsing information and the linguistic information from WordNet are not suitable for constructing large-scale sentiment lexicon from Twitter. The reason lies in that WordNet cannot well cover the colloquial expressions in tweets, and it is hard to have reliable tweet parsers due to the informal language style.

In this paper, we propose to build large-scale sentiment lexicon from Twitter with a representation learning approach, as illustrated in Figure 1. We cast sentiment lexicon learning as a phrase-level classification task. Our method contains two part: (1) a representation learning algorithm to effectively learn the continuous representation of phrases, which are used as features for phrase-level sentiment classification, (2) a seed expansion algorithm that enlarge a small list of sentiment seeds to collect training data for building the phrase-level classifier. Specifically, we learn sentiment-specific phrase embedding (**SSPE**), which is a low-dimensional, dense and real-valued vector, by encoding the sentiment information and

[*]This work was partly done when the first author was visiting Microsoft Research.
[†]Corresponding author.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 172–182, Dublin, Ireland, August 23-29 2014.

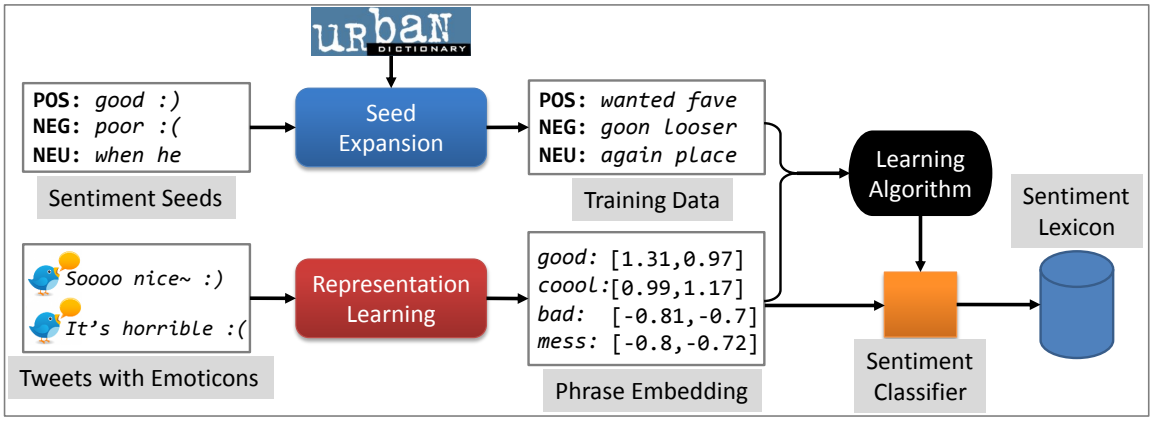


Figure 1: The representation learning approach for building Twitter-specific sentiment lexicon.

syntactic contexts into the continuous representation of phrases [1]. As a result, the nearest neighbors in the embedding space of SSPE are favored to have similar semantic usage as well as the same sentiment polarity. To this end, we extend the existing phrase embedding learning algorithm (Mikolov et al., 2013b), and develop a dedicated neural architecture with hybrid loss function to incorporate the supervision from sentiment polarity of text (e.g. tweets). We learn SSPE from tweets, leveraging massive tweets containing positive and negative emoticons as training set without any manual annotation. To obtain more training data for building the phrase-level sentiment classifier, we exploit the similar words from Urban Dictionary [2], which is a crowd-sourcing resource, to expand a small list of sentiment seeds. Finally, we utilize the classifier to predict the sentiment score of each phrase in the vocabulary of SSPE, resulting in the sentiment lexicon.

We evaluate the effectiveness of our sentiment lexicon (**TS-Lex**) by applying it in a supervised learning framework (Pang et al., 2002) for Twitter sentiment classification. Experiment results on the benchmark dataset of SemEval 2013 show that, TS-Lex yields better performance than previously introduced lexicons, including two large-scale Twitter-specific sentiment lexicons, and further improves the top-performed system in SemEval 2013 by feature combination. The quality of SSPE is also evaluated by regarding SSPE as the feature for sentiment classification of the items in existing sentiment lexicons (Hu and Liu, 2004; Wilson et al., 2005). Experiment results show that SSPE outperforms existing embedding learning algorithms. The main contributions of this work are as follows:

- To our best knowledge, this is the first work that leverages the continuous representation of phrases for building large-scale sentiment lexicon from Twitter;
- We propose a tailored neural architecture for learning the sentiment-specific phrase embedding from massive tweets selected with positive and negative emoticons;
- We report the results that our lexicon outperforms existing sentiment lexicons by applying them in a supervised learning framework for Twitter sentiment classification.

2 Related Work

In this section, we give a brief review about building sentiment lexicon and learning continuous representation of words and phrases.

2.1 Sentiment Lexicon Learning

Sentiment lexicon is a fundamental component for sentiment analysis, which can be built manually (Das and Chen, 2007), through heuristics (Kim and Hovy, 2004) or using machine learning algorithms (Turney, 2002; Li et al., 2012; Xu et al., 2013). Existing studies typically employ machine learning methods

[1] Word/unigram is also regarded as phrase in this paper.
[2] http://www.urbandictionary.com/

and adopt the propagation method to build sentiment lexicon. In the first step, a graph is built by regarding each item (word or phrase) as a node and their similarity as the edge. Then, graph propagation algorithms, such as pagerank (Esuli and Sebastiani, 2007), label propagation (Rao and Ravichandran, 2009) or random walk (Baccianella et al., 2010), are utilized to iteratively calculate the sentiment score of each item. Under this direction, parsing results, syntactic contexts or linguistic clues in thesaurus are mostly explored to calculate the similarity between items. Wiebe (2000) utilize the dependency triples from an existing parser (Lin, 1994). Qiu et al. (2009; 2011) adopt dependency relations between sentiment words and aspect words. Esuli and Sebastiani (2005) exploit the glosses information from Wordnet. Hu and Liu (2004) use the synonym and antonym relations within linguistic resources. Velikovich et al. (2010) represent words and phrases with their syntactic contexts within a window size from the web documents. Unlike the dominated propagation based methods, we explore the classification framework based on representation learning for building large-scale sentiment lexicon from Twitter.

To construct the Twitter-specific sentiment lexicon, Mohammad et al. (2013) use pointwise mutual information (PMI) between each phrase and hashtag/emoticon seed words, such as *#good*, *#bad*, *:)* and *:(*. Chen et al. (2012) utilize the Urban Dictionary and extract the target-dependent sentiment expressions from Twitter. Unlike Mohammad et al. (2013) that only capture the relations between phrases and sentiment seeds, we exploit the semantic and sentimental connections between phrases through phrase embedding and propose a representation learning approach to build sentiment lexicon.

2.2 Learning Continuous Representation of Word and Phrase

Continuous representation of words and phrases are proven effective in many NLP tasks (Turian et al., 2010). Embedding learning algorithms have been extensively studied in recent years (Bengio et al., 2013), and are dominated by the syntactic context based algorithms (Bengio et al., 2003; Collobert et al., 2011; Dahl et al., 2012; Huang et al., 2012; Mikolov et al., 2013a; Lebret et al., 2013; Sun et al., 2014). To integrate the sentiment information of text into the word embedding, Maas et al. (2011) extend the probabilistic document model (Blei et al., 2003) and predict the sentiment of a sentence with the embedding of each word. Labutov and Lipson (2013) learn task-specific embedding from an existing embedding and sentences with gold sentiment polarity. Tang et al. (2014) propose to learn sentiment-specific word embedding from tweets collected by emoticons for Twitter sentiment classification. Unlike previous trails, we learn sentiment-specific phrase embedding with a tailored neural network. Unlike Mikolov et al. (2013b) that only use the syntactic contexts of phrases to learn phrase embedding, we integrate the sentiment information of text into our method. It is worth noting that we focus on learning the continuous representation of words and phrases, which is orthogonal with Socher et al. (2011; 2013) that learn the compositionality of sentences.

3 Methodology

In this section, we describe our method for building large-scale sentiment lexicon from Twitter within a classification framework, as illustrated in Figure 1. We leverage the continuous representation of phrases as features, without parsers or hand-crafted rules, and automatically obtain the training data by seed expansion from Urban Dictionary. After the classifier is built, we employ it to predict the sentiment distribution of each phrase in the embedding vocabulary, resulting in the sentiment lexicon. To encode the sentiment information into the continuous representation of phrases, we extend an existing phrase embedding learning algorithm (Mikolov et al., 2013b) and develop a tailored neural architecture to learn sentiment-specific phrase embedding (**SSPE**), as described in subsection 3.1. To automatically obtain more training data for building the phrase-level sentiment classifier, we use the similar words from Urban Dictionary to expand a small list of sentiment seeds, as described in subsection 3.2.

3.1 Sentiment-Specific Phrase Embedding

Mikolov et al. (2013b) introduce Skip-Gram to learn phrase embedding based on the context words of phrases, as illustrated in Figure 2(a).

Given a phrase w_i, Skip-Gram maps it into its continuous representation e_i. Then, Skip-Gram utilizes

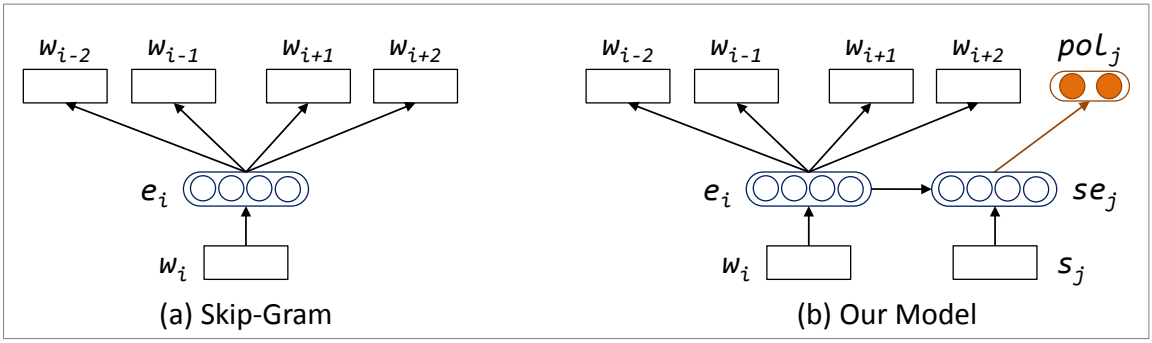


Figure 2: The traditional Skip-Gram model and our neural architecture for learning sentiment-specific phrase embedding (SSPE).

e_i to predict the context words of w_i, namely w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}, et al. Hierarchical softmax (Morin and Bengio, 2005) is leveraged to accelerate the training procedure because the vocabulary size of phrase table is typically huge. The objective of Skip-Gram is to maximize the average log probability:

$$f_{syntactic} = \frac{1}{T}\sum_{i=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log\ p(w_{i+j}|e_i) \tag{1}$$

where T is the occurrence of each phrase in the corpus, c is the window size, e_i is the embedding of the current phrase w_i, w_{i+j} is the context words of w_i, $p(w_{i+j}|e_i)$ is calculated with hierarchical softmax. The basic $softmax$ unit is calculated as $softmax_i = exp(z_i)/\sum_k exp(z_k)$. We leave out the details of hierarchical softmax (Morin and Bengio, 2005; Mikolov et al., 2013b) due to the page limit. It is worth noting that, Skip-Gram is capable to learn continuous representation of words and phrases with the identical model (Mikolov et al., 2013b).

To integrate sentiment information into the continuous representation of phrases, we develop a tailored neural architecture to learn SSPE, as illustrated in Figure 2(b). Given a triple $\langle w_i, s_j, pol_j\rangle$ as input, where w_i is a phrase contained in the sentence s_j whose gold sentiment polarity is pol_j, our training objective is to (1) utilize the embedding of w_i to predict its context words, and (2) use the sentence representation se_j to predict the gold sentiment polarity of s_j, namely pol_j. We simply average the embedding of phrases contained in a sentence as its continuous representation (Huang et al., 2012). The objective of the sentiment part is to maximize the average of log sentiment probability:

$$f_{sentiment} = \frac{1}{S}\sum_{j=1}^{S} log\ p(pol_j|se_j) \tag{2}$$

where S is the occurrence of each sentence in the corpus, $\sum_k pol_{jk} = 1$. For binary classification between positive and negative, the distribution of [0,1] is for positive and [1,0] is for negative. Our final training objective is to maximize the linear combination of the syntactic and sentiment parts:

$$f = \alpha \cdot f_{syntactic} + (1-\alpha)\cdot f_{sentiment} \tag{3}$$

where α weights the two parts. Accordingly, the nearest neighbors in the embedding space of SSPE are favored to have similar semantic usage as well as the same sentiment polarity.

We train our neural model with stochastic gradient descent and use AdaGrad (Duchi et al., 2011) to update the parameters. We empirically set embedding length as 50, window size as 3 and the learning rate of AdaGrad as 0.1. Hyper-parameter α is tuned on the development set. To obtain large-scale training corpus, we collect tweets from April, 2013 through TwitterAPI. After filtering the tweets that are too short ($<$ 5 words) and removing *@user* and *URLs*, we collect 10M tweets (5M positive and 5M negative) with positive and negative emoticons [3], which is are utilized as the training data to train our neural model. The vocabulary size is 750,000 after filtering the 1$\sim$4 grams through frequency.

[3]We use the emoticons selected by Hu et al. (2013), namely :) :) :-) :D =) as positive and :(: (:-(as negative ones.

3.2 Seed Expansion with Urban Dictionary

Urban Dictionary is a web-based dictionary that contains more than seven million definitions until March, 2013 [4]. It was intended as a dictionary of slang, cultural words or phrases not typically found in standard dictionaries, but it is now used to define any word or phrase. For each item in Urban Dictionary, there is a list of similar words contributed by volunteers. For example, the similar words of "*cooool*" are "*cool*", "*awesome*", "*coooool*", et al [5] and the similar words of "*not bad*" are "*good*", "*ok*" and "*cool*", et al [6]. These similar words are typically semantically close to and have the same sentiment polarity with the target word. We conduct preliminary statistic on the items of Urban Dictionary from "*a*" to "*z*", and find that there are total 799,430 items containing similar words and each of them has about 10.27 similar words on average.

We utilize Urban Dictionary to expand little sentiment seeds for collecting training data for building the phrase-level sentiment classifier. We manually label the top frequent 500 words from the vocabulary of SSPE as positive, negative or neutral. After removing the ambiguous ones, we obtain 125 positive, 109 negative and 140 neutral words, which are regarded as the sentiment seeds [7]. Afterwards, we leverage the similar words from Urban Dictionary to expand the sentiment seeds. We first build a k-nearest neighbors (KNN) classifier by regarding the sentiment seeds as gold standard. Then, we employ the KNN classifier on the items of Urban Dictionary containing similar words, and predict a three-dimensional discrete vector $[knn_{pos}, knn_{neg}, knn_{neu}]$ for each item, reflecting the hits numbers of sentiment seeds with different sentiment polarity in its similar words. For example, the vector value of "*not bad*" is $[10, 0, 0]$, which means that there are 10 positive seeds, 0 negative seeds and 0 neutral seeds occur in its similar words. To ensure the quality of the expanded words, we set threshold for each category to collect the items with high quality as expanded words. Take the positive category as an example, we keep an item as positive expanded word if it satisfies $knn_{pos} > knn_{neg} + threshold_{pos}$ and $knn_{pos} > knn_{neu} + threshold_{pos}$ simultaneously. We empirically set the thresholds of positive, negative and neutral as 6,3,2 respectively by balancing the size of expanded words in three categories. After seed expansion, we collect 1,512 positive, 1,345 negative and 962 neutral words, which are used as the training data to build the phrase-level sentiment classifier. We also tried the propagation methods to expand the sentiment seeds, namely iteratively added the similar words of sentiment seeds from Urban Dictionary into the expanded word collection. However, the quantity of expanded words is less than the KNN-based results and the quality is relatively poor.

After obtaining the training data and feature representation of phrases, we build the phrase-level classifier with $softmax$, whose length is two for the positive *vs* negative case:

$$\mathbf{y}(w) = softmax(\theta \cdot e_i + b) \tag{4}$$

where θ and b are the parameters of classifier, e_i is the embedding of the current phrase w_i, $\mathbf{y}(w)$ is the predicted sentiment distribution of item w_i. We employ the classifier to predict the sentiment distribution of each phrase in the vocabulary of SSPE, and save the phrases as well as their sentiment probability in the positive (negative) lexicon if the positive (negative) probability is larger than 0.5.

4 Experiment

In this section, we conduct experiments to evaluate the effectiveness of our sentiment lexicon (**TS-Lex**) by applying it in the supervised learning framework for Twitter sentiment classification, as given in subsection 4.1. We also directly evaluate the quality of SSPE as it forms the fundamental component for building sentiment lexicon. We use SSPE as the feature for sentiment classification of items in existing sentiment lexicons, as described in subsection 4.2.

[4] http://en.wikipedia.org/wiki/Urban_Dictionary
[5] http://www.urbandictionary.com/define.php?term=cooool
[6] http://www.urbandictionary.com/define.php?term=not+bad
[7] We will publish the sentiment seeds later.

4.1 Twitter Sentiment Classification

Experiment Setup and Dataset We conduct experiments on the benchmark Twitter sentiment classification dataset (message-level) from SemEval 2013 (Nakov et al., 2013). The training and development sets were completely released to task participants. However, we were unable to download all the training and development sets because some tweets were deleted or not available due to modified authorization status. The statistic of the positive and negative tweets in our dataset are given in Table 1(b). We train positive *vs* negative classifier with LibLinear (Fan et al., 2008) with default settings on the training set, tune parameters *-c* on the dev set and evaluate on the test set. The evaluation metric is Macro-F1.

(a) Sentiment Lexicons

Lexicon	Positive	Negative	Total
HL	2,006	4,780	6,786
MPQA	2,301	4,150	6,451
NRC-Emotion	2,231	3,324	5,555
TS-Lex	**178,781**	**168,845**	**347,626**
HashtagLex	216,791	153,869	370,660
Sentiment140Lex	480,008	260,158	740,166

(b) SemEval 2013 Dataset

	Positive	Negative	Total
Train	2,642	994	3,636
Dev	408	219	627
Test	1,570	601	2,171

Table 1: Statistic of sentiment lexicons and Twitter sentiment classification datasets.

Results and Analysis We compare TS-Lex with *HL*[8] (Hu and Liu, 2004), *MPQA*[9] (Wilson et al., 2005), *NRC-Emotion*[10] (Mohammad and Turney, 2012), *HashtagLex* and *Sentiment140Lex*[11] (Mohammad et al., 2013). The statistics of TS-Lex and other sentiment lexicons are illustrated in Table 1(a). *HL*, *MPQA* and *NRC-Emotion* are traditional sentiment lexicons with a relative small lexicon size. *HashtagLex* and *Sentiment140Lex* are Twitter-specific sentiment lexicons. We can find that, TS-Lex is larger than the traditional sentiment lexicons.

We evaluate the effectiveness of TS-Lex by applying it as the features for Twitter sentiment classification in the supervised learning framework (Pang et al., 2002). We conduct experiments in two settings, namely only utilizing the lexicon features (*Unique*) and appending lexicon feature to existing feature sets (*Appended*). In the first setting, we design the lexicon features as same as the top-performed Twitter sentiment classification system in SemEval2013 [12] (Mohammad et al., 2013). For each sentiment polarity (positive *vs* negative), the lexicon features are:

- total count of tokens in the tweet with score greater than 0;
- the sum of the scores for all tokens in the tweet;
- the maximal score;
- the non-zero score of the last token in the tweet;

In the second experiment setting, we append the lexicon features to the existing basic feature. We use the feature sets of Mohammad et al. (2013) excluding the lexicon feature as the basic feature, including bag-of-words, pos-tagging, emoticons, hashtags, elongated words, etc. Experiment results of the *Unique* features and *Appended* features from different sentiment lexicons on Twitter sentiment classification are given in Table 2(a).

From Table 2(a), we can find that TS-Lex yields best performance in both *Unique* and *Appended* feature sets among all sentiment lexicons, including two large-scale Twitter-specific sentiment lexicons. The reason is that the classifier for building TS-Lex utilize (1) the well developed feature representation of phrases (SSPE), which captures the semantic and sentiment connections between phrases, and (2) the enlarged sentiment words through web intelligence as training data. *HashtagLex* and *Sentiment140Lex*

[8] http://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html#lexicon
[9] http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/
[10] http://www.saifmohammad.com/WebPages/ResearchInterests.html
[11] We utilize the unigram and bigram lexicons from *HashtagLex* and *Sentiment140Lex*.
[12] http://www.saifmohammad.com/WebPages/Abstracts/NRC-SentimentAnalysis.htm

(a)

Lexicon	Unique	Appended
HL	60.49	79.40
MPQA	59.15	76.54
NRC-Emotion	54.81	76.79
HashtagLex	65.30	76.67
Sentiment140Lex	72.51	80.68
TS-Lex	**78.07**	**82.36**

(b)

Lexicon	Unique
Seed	57.92
Expand	60.69
Lexicon(seed)	74.64
TS-Lex	**78.07**

Table 2: Macro-F1 on Twitter sentiment classification with different lexicon features.

only utilize the relations between phrases and hashtag/emoticon seeds, yet do not well capture the connections between phrases. In the *Unique* setting, the performances of the traditional lexicons (*HL*, *MPQA* and *NRC-Emotion*) are lower than large-scale Twitter-specific lexicons (*HashtagLex*, *Sentiment140Lex* and our lexicon). The reason is that, tweets have the informal language style and contain slangs and diverse multi-word phrases, which are not well covered by the traditional sentiment lexicons with a small size. After incorporating the lexicon feature of TS-Lex into the top-performs system (Mohammad et al., 2013), we further improve the macro-F1 from 84.70% to 85.65%.

Effect of Seed Expansion with Urban Dictionary To verify the effectiveness of seed expansion through Urban Dictionary, we conduct experiments by applying (1) sentiment seeds (*Seed*), (2) words after expansion (*Expand*), (3) sentiment lexicon generated from the classifier only utilizing sentiment seeds as training data (*Lexicon(seed)*), (4) the final lexicon (*TS-Lex*) exploiting the expanded words as training data to build sentiment classifier, to produce lexicon features, and only use them for Twitter sentiment classification (*Unique*). From Table 2(b), we find that the performance of sentiment seeds and expanded words are relatively poor due to their low coverage. Under this scenario, seed expansion yields 2.77% improvement (from 57.92% to 60.69%) on macro-F1. By utilizing the expanded words as training data to build the phrase-level sentiment classifier, TS-Lex obtains 3.43% improvements on Twitter sentiment classification (from 74.64% to 78.07%), which verifies the effectiveness of seed expansion through Urban Dictionary. In addition, we find that only using a small number of sentiment seeds as the training data, we can obtain superior performance (74.64%) than all baseline lexicons. This indicates that the representation learning approach effectively capture the semantic and sentimental connections between phrases through SSPE, and leverage them for building the sentiment lexicon.

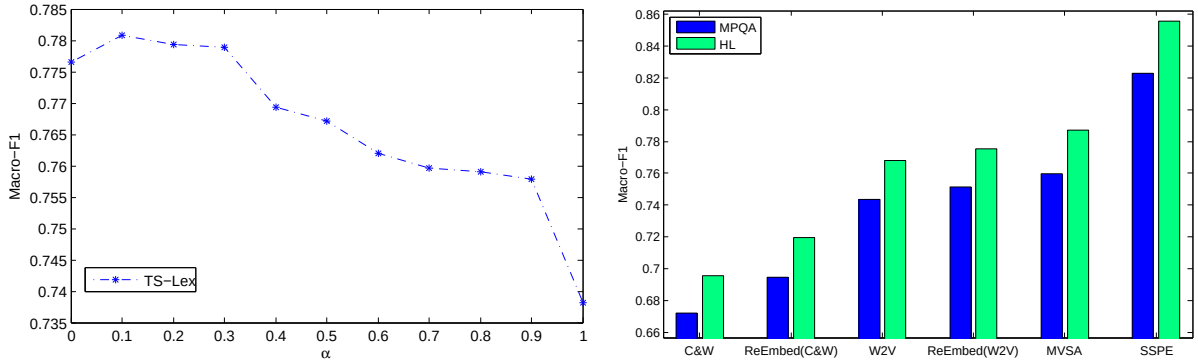
Effect of α in SSPE We tune the hyper-parameter α of SSPE on the development set of SemEval 2013, and study its influence on the performance of Twitter sentiment classification by applying the generated lexicon as features. We utilize the expanded words as training data to train $softmax$ and only utilize the lexicon features (*Unique*) for Twitter sentiment classification. Experiment results with different α are illustrated in Figure 3(a).

From Figure 3(a), we can see that that SSPE performs better when α is in the range of [0.1, 0.3], which is dominated by the sentiment information. The model with $\alpha = 1$ stands for Skip-Gram model. The sharp decline at $\alpha = 1$ indicates the importance of sentiment information in learning sentiment-specific phrase embedding for building sentiment lexicon.

Discussion In the experiment, we do not apply TS-Lex into the unsupervised learning framework for Twitter sentiment classification. The reason is that the lexicon-based unsupervised method typically require the sentiment lexicon to have high precision, yet our task is to build large-scale lexicon (TS-Lex) with broad coverage. We leave this as the future work, although we may set higher threshold (e.g. larger than 0.5) to increase the precision of TS-Lex and loose the recall.

4.2 Evaluation of Different Representation Learning Methods

Experiment Setup and Dataset We conduct sentiment classification of items in two traditional sentiment lexicons, *HL* (Hu and Liu, 2004) and *MPQA* (Wilson et al., 2005), to evaluate the effective of the



(a) SSPE with different α on the development set for Twitter sentiment classification.

(b) Sentiment classification of lexicons with different embedding learning algrithms.

Figure 3: Experiment results with different settings.

sentiment-specific phrase embedding (SSPE). We train the positive *vs* negative classifier with LibLinear (Fan et al., 2008). The evaluation metric is the macro-F1 of 5-fold cross validation. The statistics of *HL* and *MPQA* are listed in Table 1(a).

Baseline Embedding Learning Algorithms We compare SSPE with the following embedding learning algorithms:

(1) *C&W*. C&W is one of the most representative embedding learning algorithms (Collobert et al., 2011) for learning word embedding, which has been proven effective in many NLP tasks.

(2) *W2V*. Mikolov et al. (2013a) introduce Word2Vec for learning the continuous vectors for words and phrases. We utilize Skip-Gram as it performs better than CBOW in the experiments.

(3) *MVSA*. Maas et al. (2011) learn word vectors for sentiment analysis with a probabilistic model of documents utilizing the sentiment polarity of documents.

(4) *ReEmbed*. Lebret et al. (2013) learn task-specific embedding from existing embedding and task-specific corpus. We utilize the training set of Twitter sentiment classification as the labeled corpus to re-embed words. *ReEmbed(C&W)* and *ReEmbed(W2V)* stand for the use of different embedding results as the reference word embedding.

The embedding results of the baseline algorithms and SSPE are trained with the same dataset and parameter sets.

Results and Analysis Experiment results of the baseline embedding learning algorithms and SSPE are given in Figure 3(b). We can see that SSPE yields best performance on both lexicons. The reason is that SSPE effectively encode the sentiment information of tweets as well as the syntactic contexts of phrases from massive data into the continuous representation of phrases. The performances of *C&W* and *W2V* are relatively low because they only utilize the syntactic contexts of items, yet ignore the sentiment information of text, which is crucial for sentiment analysis. *ReEmbed(C&W)* and *ReEmbed(W2V)* achieve better performance than *C&W* and *W2V* because the sentiment information of sentences are incorporated into the continuous representation of phrases. There is a gap between *ReEmbed* and SSPE because SSPE leverages more sentiment supervision from massive tweets collected by positive and negative emoticons.

5 Conclusion

In this paper, we propose building large-scale Twitter-specific sentiment lexicon with a representation learning approach. Our method contains two parts: (1) a representation learning algorithm to effectively learn the embedding of phrases, which are used as features for classification, (2) a seed expansion algorithm that enlarge a small list of sentiment seeds to obtain training data for building the phrase-level sentiment classifier. We introduce a tailored neural architecture and integrate the sentiment information of tweets into its hybrid loss function for learning sentiment-specific phrase embedding (**SSPE**). We learn SSPE from the tweets collected by positive and negative emoticons, without any manual annota-

tion. To collect more training data for building the phrase-level classifier, we utilize the similar words from Urban Dictionary to expand a small list of sentiment seeds. The effectiveness of our sentiment lexicon (**TS-Lex**) has been verified through applied in the supervised learning framework for Twitter sentiment classification. Experiment results on the benchmark dataset of SemEval 2013 show that, TS-Lex outperforms previously introduced sentiment lexicons and further improves the top-perform system in SemEval 2013 with feature combination. In future work, we plan to apply TS-Lex into the unsupervised learning framework for Twitter sentiment classification.

Acknowledgements

We thank Nan Yang, Yajuan Duan and Yaming Sun for their great help. This research was partly supported by National Natural Science Foundation of China (No.61133012, No.61273321, No.61300113).

References

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 2200–2204.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Analysis and Machine Intelligence*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Lu Chen, Wenbo Wang, Meenakshi Nagarajan, Shaojun Wang, and Amit P Sheth. 2012. Extracting diverse sentiment expressions with target-dependent polarity from twitter. In *ICWSM*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

George E Dahl, Ryan P Adams, and Hugo Larochelle. 2012. Training restricted boltzmann machines on word observations. *ICML*.

Sanjiv R Das and Mike Y Chen. 2007. Yahoo! for amazon: Sentiment extraction from small talk on the web. *Management Science*, 53(9):1375–1388.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, pages 2121–2159.

Andrea Esuli and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 617–624. ACM.

Andrea Esuli and Fabrizio Sebastiani. 2007. Pageranking wordnet synsets: An application to opinion mining. In *ACL*, volume 7, pages 442–431.

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

Ronen Feldman. 2013. Techniques and applications for sentiment analysis. *Communications of the ACM*, 56(4):82–89.

Ming Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 168–177.

Xia Hu, Jiliang Tang, Huiji Gao, and Huan Liu. 2013. Unsupervised sentiment analysis with emotional signals. In *Proceedings of the International World Wide Web Conference*, pages 607–618.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *ACL*, pages 873–882. ACL.

Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1367. Association for Computational Linguistics.

Igor Labutov and Hod Lipson. 2013. Re-embedding words. In *Annual Meeting of the Association for Computational Linguistics*.

Rémi Lebret, Joël Legrand, and Ronan Collobert. 2013. Is deep learning really necessary for word embeddings? *NIPS workshop*.

Fangtao Li, Sinno Jialin Pan, Ou Jin, Qiang Yang, and Xiaoyan Zhu. 2012. Cross-domain co-extraction of sentiment and topic lexicons. In *Proceedings of the 50th ACL*, pages 410–419. ACL, July.

Dekang Lin. 1994. Principar: an efficient, broad-coverage, principle-based parser. In *Proceedings of the 15th conference on COLING*, pages 482–488. Association for Computational Linguistics.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *ICLR*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *The Conference on Neural Information Processing Systems*.

Saif M Mohammad and Peter D Turney. 2012. Crowdsourcing a word–emotion association lexicon. *Computational Intelligence*.

Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *Proceedings of the International Workshop on Semantic Evaluation*.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *Proceedings of the international workshop on artificial intelligence and statistics*, pages 246–252.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the International Workshop on Semantic Evaluation*, volume 13.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *IJCAI*, volume 9, pages 1199–1204.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27.

Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 675–682. Association for Computational Linguistics.

Richard Socher, J. Pennington, E.H. Huang, A.Y. Ng, and C.D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Conference on Empirical Methods in Natural Language Processing*, pages 151–161.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2014. Radical-enhanced chinese character embedding. *arXiv preprint arXiv:1404.4714*.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Procedding of the 52th Annual Meeting of Association for Computational Linguistics.*

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. *Annual Meeting of the Association for Computational Linguistics*.

Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pages 417–424.

Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 2010. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 777–785. Association for Computational Linguistics.

Janyce Wiebe. 2000. Learning subjective adjectives from corpora. In *AAAI/IAAI*, pages 735–740.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 347–354.

Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen, and Jun Zhao. 2013. Mining opinion words and opinion targets in a two-stage framework. In *Proceedings of the 51st ACL*, pages 1764–1773. ACL.

Political Tendency Identification in Twitter using Sentiment Analysis Techniques

Ferran Pla and Lluís-F. Hurtado
Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València
Camí Vera s/n 46022 València (Spain)
{lhurtado|fpla}@dsic.upv.es

Abstract

This paper describes an approach for political tendency identification of Twitter users. We define some metrics that take into account the polarity of the political entities in the tweets of each user. To obtain this polarities we present the sentiment analysis system developed. The evaluation was performed on the general corpus developed at TASS2013 workshop for Spanish. To our knowledge, the results obtained for the sentiment analysis task and the political tendency identification task are the best results published until now using this data set.

1 Introduction

Social media are usually used to express opinions and feelings about companies, products, services, hobbies, politics, etc. Therefore, enterprises, organizations, governments, and different groups in general have shown interest in the opinions that users have for their activities. They are also interested to known the way users use these media, the communication behaviour, and some users attributes such as gender, age, geographical location, political orientation, etc. In general, the main aim is to provide personalized services, particularized offers, or simply to know what people think about something in order to improve their activities.

The scientific community has made a great effort to provide effective solutions to analyse, structure, and process the large amount of on-line reviews in social media. A wide set of techniques of Sentiment Analysis (SA) are used in micro-blogging texts to extract the polarity (positive, negative, mixed or neutral) that users express in these texts. In this respect, Twitter has become a popular micro-blogging site in which users express their opinions on a variety of topics in real time. The texts used in Twitter are called tweets, which are short texts of a maximum of 140 characters and a language that does not have any restriction on the form and content. The nature of these texts poses new challenges for researchers in Natural Language Processing (NLP). In some cases, the tweets are written with ungrammatical sentences with a lot of emoticons, abbreviations, specific terminology, slang, etc. Therefore, the usual techniques of NLP must be adapted to these characteristics of the language, and new approaches must be proposed in order to successfully address this problem. NLP tools like POS taggers, parsers, or Named Entity Recognition (NER) tools usually fail when processing tweets because they generally are trained on grammatical texts and they perform poorly in micro-blogging texts.

In this work we present a system for addressing the task of political tendency identification of Twitter users based on SA techniques. For each user, we collect all their tweets and we extract all the entities related to the political subject. Then, we automatically assign a polarity to these entities and we define a political tendency metric that uses this entity polarity information combined with another tendency metric for classifying the political tendency of each user in four categories: *Left*, *Right*, *Center*, or *Undefined*. The evaluation of our system is performed on the General Corpus, a corpus of Spanish tweets provided by the organization of the TASS2013 workshop.

The paper is organized as follows. In section 2 we present relevant works for Twitter user classification and Sentiment Analysis. In Section 3 we present a description of the corpus used to evaluate our user

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 183–192, Dublin, Ireland, August 23-29 2014.

political tendency system. This system is based on SA techniques. A description of our SA system is described in section 4. In Section 5 we describe the way we classify users according to their political leading. The evaluation and discussion of the results obtained are presented in section 6. Finally, in section 7 we present some conclusions and possible directions for future works.

2 Related works

The different approaches for estimating the political leaning of Twitter users explore features that range from *text content*, *users behavior* (taking into account the tweets and retweets information) and the *Twitter structure* (by considering the followers users, following users, etc.). An interesting study of some useful features to classify latent users attributes (gender, age, regional origin, and political orientation) is presented in (Rao et al., 2010). In (Conover et al., 2011a; Conover et al., 2011b) and study of the political alignment of Twitter users is performed by analyzing the way users communicates by means of retweets and user mentions. In (O'Connor et al., 2010a) SA techniques are used to determine the positive and negative polarity of Twitter messages. They also study the connexion between these polarities and the public opinion derived from traditional polling in order to substitute or complement them. (Pennacchiotti and Popescu, 2011) present a machine learning approach to Twitter user classification in democrats or republicans. With respect to the linguistic content they considered prototypical words and hashtags that are common in democrats or republicans users which provides clues for the classification. They also use SA tecniques based on lexicons for the classification task. In (Boutet et al., 2012) polical leading of users is performed by counting the amount of tweets related to political parties analysing the hashtags. They also consider the interaction among parties by analyzing the retweets and mentions. Users interaction by analysing tweets and retweets is also the main idea of the work presented in (Wong et al., 2013).

In (Cohen and Ruths, 2013) previous works on political orientation of Twitter users are analyzed to conclude that the accuracy results reported are overstimated do to the way the data sets are constructed. When these approaches are applied to normal Twitter users accuracy results significantly decrease.

Sentiment Analysis (SA) has been widely studied in the last decade in multiple domains. Most work focuses on classifying the polarity of the texts as positive, negative, mixed, or neutral. The pioneering works in this field used supervised (Pang et al., 2002) or unsupervised (knowledge-based) (Turney, 2002) approaches. In (Pang et al., 2002), the performance of different classifiers on movie reviews was evaluated. In (Turney, 2002), some patterns containing POS information were used to identify subjective sentences in reviews to then estimate their semantic orientation.

The construction of polarity lexicons is another widely explored field of research. Opinion lexicons have been obtained for English language (Liu et al., 2005) (Wilson et al., 2005) and also for Spanish language (Perez-Rosas et al., 2012). A good presentation of the SA problem and a description of the state-of-the-art of the more relevant approaches to SA can be found in (Liu, 2012). An overview of the current state of different approaches to the subjectivity and SA task is presented in (Montoyo et al., 2012).

Research works about SA on Twitter are much more recent. Twitter appeared in the year 2006 and the early works in this field are from 2009 when Twitter started to achieve popularity. Some of the most significant works are (Barbosa and Feng, 2010), (Jansen et al., 2009), and (O'Connor et al., 2010b). A survey of the most relevant approaches to SA on Twitter can be see in (Martínez-Cámara et al., 2012) , (Vinodhini and Chandrasekaran, 2012). The SemEval2013 competition has also dedicated a specific task for SA on Twitter (Wilson et al., 2013), which shows the great interest of the scientific community in this field. The TASS2013 workshop has proposed different tasks for SA and political tendency identification focused on the Spanish language (Villena-Román and García-Morera, 2013).

3 The Corpus

The General Corpus of TASS2013[1](Villena-Román and García-Morera, 2013) contains approximately 68000 Twitter messages (*tweets*) written in Spanish (between November 2011 and March 2012) by 158 well-known personalities of the world of politics, economy, communication, mass media, and culture.

[1]This corpus is freely available on the web page of TASS2013 (http://www.daedalus.es/TASS2013).

The corpus is encoded in XML. Each tweet includes its ID (*tweetid*), the creation date (*date*), and the user ID (*user*). It is tagged with its global polarity using N and N+ labels for negative polarity with different intensity, P and P+ labels for positive polarity with different intensity, and the NEU label for neutral polarity. Label NONE was used to represent tweets with no polarity at all. Moreover, the polarity to the entities that are mentioned in the tweet was also included. The level of agreement of the expressed sentiment is annotated both for global and entity level. Also, a selection of a set of topics was made based on the thematic areas covered by the corpus, such as politics, soccer, literature, entertainment, etc. Each message is also assigned to one or several of these topics.

	N	N+	NEU	NONE	P	P+
training	1,335 (18.49%)	847 (11.73%)	670 (9.28%)	1,483 (20.54%)	1,232 (17.07%)	1,652 (22.88%)
test	11,287 (18.56%)	4,557 (7.50%)	1,305 (2.15%)	21,416 (35.22%)	1,488 (2.45%)	20,745 (34.12%)

Table 1: The distribution of the polarity of the tweets in the corpus.

Table 1 shows the distribution of tweets per polarity in the corpus. It is divided into two sets: training (about 10%, 7219 tweets) and test (about 90%, 60798 tweets). It can be observed that this distribution is not balanced for the different polarities. Finally, each user from the test set of the General corpus is labeled with their political tendency in four possible values: $Left$, $Right$, $Centre$, and $Undefined$.

4 Description and Evaluation of the Sentiment Analysis System

Figure 1 shows an overview of our system for the SA problem. The system consists of 4 modules. The first module is the Pre-processing module, which performs the tokenization, lemmatization, and Named Entities recognition of the input tweet. A lemma reduction and a POS tagging process is also carried out in this module. The second module is optional. It allows us to obtain the polarity of the entities contained in the tweet. If we omitted this step the global polarity of the tweet is obtained. The third module is the Feature Extraction module, which selects the features from the pre-processed tweet (or from the segments of tweets) and obtains a feature vector. Some features require the use of a polarity lexicon of lemmas and words. To determine the best features, a tuning process is required during the training phase. The fourth module is the Polarity Classifier module, which uses a classifier (learned from feature vectors of the training set) to assign a polarity label to the tweet.

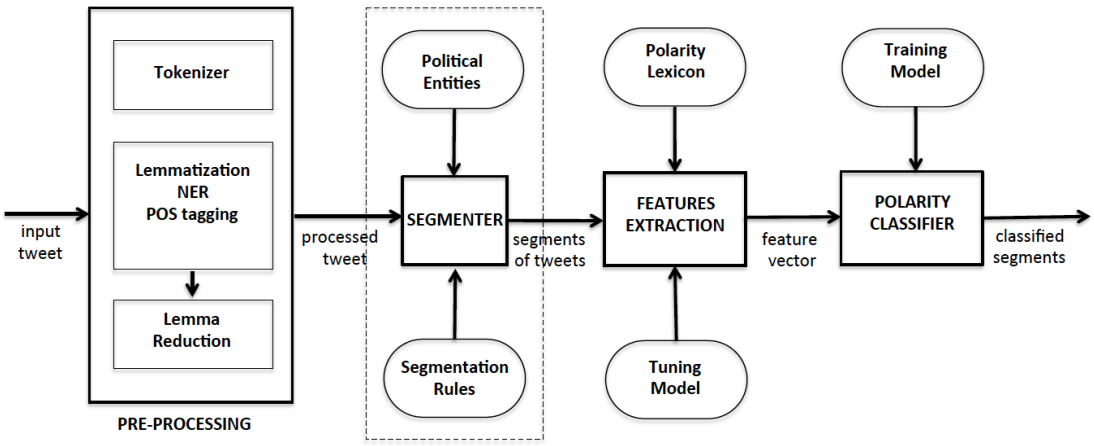


Figure 1: Sentiment Analysis System Overview

4.1 Pre-processing of Tweets

Before addressing the SA task, it is necessary to make a proper tokenization of the tweets. Although there are a lot of tokenizers available on the web, they need to be adapted in order to address the segmentation of tokens of a tweet. Furthermore, most of these resources are for the English language, which adds a degree of difficulty for their use in processing Spanish tweets.

Moreover, the use of NLP resources such as stemmers, POS taggers, parsers, NER systems, Word Sense Disambiguation (WSD) systems, etc. are impractical if the characteristics of the tweets are not taken into account. Therefore, an adjustment and adaptation must be made for the Twitter domain.

In our system, we decided to use and adapt available tools for tokenization, lemmatization, NER, and POS tagging. We adapted the package *Tweetmotif* [2] that is described in (O'Connor et al., 2010b) to process Spanish tweets. We also used *Freeling* [3] (Padró and Stanilovsky, 2012) (with the appropriate modifications for handling Twitter messages) for stemming, Named Entity Recognition, and POS tagging.

We added some functions to process special tokens (e.g., grouping all *hashtags* into a single token, grouping all *web* addresses into a single token or grouping all *url* into a single token). We also grouped the *dates* into a single token, the *numbers* into a single token, and the *punctuation* marks into a single token.

4.2 The Segmenter

For the proposed approach we need to determine the polarity of political entities that contains a tweet. It is because the polarity of each entity could be different of the global polarity of the tweet. In the tweet of the corpus[4]: *"Rajoy's government goes up the pensions. PSOE cuts back all things except the unemployment."* we have two entities, *Rajoy* (the president of Spanish government from the right-wing party PP) and *PSOE* (a Spanish left-wing party). This tweet is labeled with a neutral global polarity, but each entity have a different polarity (ENTITY (Rajoy): *Positive*. ENTITY (PSOE): *Negative*).

Even for tweets with only one entity we must decide what fragments of text refers to that entity. In the example: *"Rajoy already has been talking for an hour. Not saving anywhere only expenses, all reforms cost a lot of money. Did he tell us something at the end?"*, to determine the polarity of entity Rajoy, we must take into account all the tweet, because the two last sentences references to *ENTITY(Rajoy)*. In contrast, in the example: *"Today 349 members attending to the formation of the lower house. Only the AMAIUR deputy for Navarra is missing"*, only the sentence containing the AMAIUR entity is being required to determine its polarity.

Obtaining the polarity at entity level is a hard problem and introduces additional complexity because the part of the tweet refers to each of the entities must be determined. To resolve this problem it should make a deep parsing of the tweet and perform a study of such dependencies. This is not a solved problem in NLP even considering normative texts and is further aggravated in Twitter texts. Besides, in many cases, the dependencies are between different sentences, and problems such as coreference must be taking into account in order to determine, for example, which pronoun refers to a certain entity. Other problems such as synonyms and acronyms of certain entities can make this problem harder.

We have chosen a more simple and practice approach that consists in defining a set of heuristics to determine which segment of the tweet refers to each of the entities present on it. We defined some rules to do this segmentation. If the tweet contains only one entity the context considered was all the tweet. We evaluated other alternatives, but due to the short length of tweets, with that decision the best global results were obtained. If the tweet contains two entities, the casuistry is greater. If both entities are placed together at the beginning or the end of the tweet all the tweet is considered as a context for both entities. By contrast, if separate, and has sufficient context, the tweet is segmented by defining the context of each entity. Next, we show some examples and the segmentation produced by the defined rules.

Example 1 is the easier case due the two entities are in separated sentences. When both entities are in the same sentence, in Example 2 the rule applied determines that the context for the first entity is from the beginning until the second entity, and the rest of the sentence is the context for the second entity. Example 3 is more difficult, and the rules applied produce segmentations like this *[On March 25 we elect between the immobility of the @PSOE] [and the renovation and the hope of the @ppandaluz.])*, that are not correct but can be useful for determining the polarity of each entity. In addition, due to the short length of the tweets, the context of an entity is often so small that it does not contain information enough

[2] https://github.com/brendano/tweetmotif.

[3] http://nlp.lsi.upc.edu/freeling/

[4] All the exemples have been translated to English.

Example 1
*[***Rajoy***'s government goes up the pensions.] [***PSOE** *cuts back all things except the unemployment.]*
GLOBAL POLARITY: *NEU*. ENTITY (Rajoy): *Positive*. ENTITY (PSOE): *Negative*
Example 2
[As **IU** *gains confidence in Andalucía] [***PP** *loses members.]*
GLOBAL POLARITY: *NEU*. ENTITY (IU): *Positive*. ENTITY (PP): *Negative*
Example 3
On March 25 we elect between [the immobility of the **@PSOE***] and [the renovation and the hope of the* **@ppandaluz.]**
GLOBAL POLARITY: *NEU*. ENTITY (@PSOE): *Negative*. ENTITY (@ppandaluz): *Positive*

to correctly classify the polarity of the entity. In such case, the option that was chosen is to establish a threshold of context, and if it is below than this threshold, it was assigned the same polarity to all the entities of the tweet. When the number of entities is greater than two in much cases we assigned the same polarity to all the entities of the tweet because we had not enough context.

4.3 Feature Selection

The feature selection process was performed by cross validation (10-fold validation) using the training set to select the set of relevant features.

We considered the following set of features: unigrams and bigrams of lemmas obtained in the preprocessing of the tweets that belong to a set of selected POS. We considered only the lemmas of a minimum frequency (*f*) in the training set. We unified all *hashtags*, *user references*, *dates*, *punctuations* as a single feature. We classified the emoticons in the following categories: *happy, sad, tongue, wink, and other*. Finally, we used external polarity lexicons of lemmas and words.

Some of the features required further adjustment. For the POS feature we selected the lemas that belongs to the *nouns, verbs, adjectives, and adverbs* POS and also *exclamations* and *emoticons*. We estimated the minimum frequency of the lemmas to be selected (*f =2*). Finally, we selected the external lexicons to be used. One of the lexicons used was originally for English language (Wilson et al., 2005) that was translated into Spanish automatically, and other (Perez-Rosas et al., 2012) lexicon was a list of words that was originally in Spanish. Then, we combined these two resource with the lexicon presented in (Saralegi and San Vicente, 2013).

4.4 Polarity Classifier

The task was addressed as a classification problem that consisted of determining the polarity of each tweet. We used WEKA[5], which is a tool that includes (among other utilities) a collection of machine-learning algorithms that can be used for classification tasks. Specifically, we used a SVM-based approach because it is a well-founded formalism, that has been successfully used in many classification problems. In the SA task, SVM has shown it ability to handle large feature spaces and to determine the relevant features (Joachims, 1998).

We used the NU-SVM algorithm (Schölkopf et al., 2000) from an external library called *LibSVM*[6], which is very efficient software for building SVM classifiers. It is easy to integrate this software with WEKA thus allowing us to use all of WEKA's features. We used the *bag_of_words* approach to represent each tweet as a feature vector that contains the frequency of the selected features of the training set.

4.5 Evaluation of the Sentiment Analysis System

We evaluated our system on the SA tasks defined at the TASS2013 workshop. Two different sub-tasks called *5-level* and *3-level* were proposed. Both sub-tasks differ only in the polarity granularity considered. The *5-leve*l sub-task uses the labels *N*, *N+*, *P*, *P+*, and *NEU*. The *3-level* sub-task uses the labels *N*, *P*, and *NEU*. In both sub-tasks, an additional label (*NONE*) was used to represent tweets with no polarity.

The accuracy results obtained on the unseen data test were: 62.88%±0.38% for *5-level* task and 70.25%±0.36% for *3-level task*. This results outperformed all the approaches at TASS2013 workshop with statistical significance (with a 95% level of confidence). The official results ranged from 61.6% to 13.5% for the *5-level* task and from 66.3% to 38.8% for the *3-level* task. The F_1 result obtained in the

[5]http://www.cs.waikato.ac.nz/ml/weka/
[6]http://www.cs.iastate.edu/~ yasser/wlsvm/

Sentiment Analysis at Entity level task was worse (F_1=0.40), but it still is the best result reported in the sentiment analysis at entity level task at TASS2013 competition.

5 Political tendency identification

The objective of this task is to estimate the political tendency of each user from the test set of the General corpus in four possible values: $Left$, $Right$, $Centre$, and $Undefined$. Next, we describe the approach we proposed for this task. This approach uses the SA system previously described in section 4.

To perform the classification of users we assume the following hypothesis: the positive opinions on a political party is a political orientation similarly to the user performing the review for this party, on the contrary, a negative opinion about a party is a political orientation opposite to that shown by this party.

In this way, to classify users by their political orientation, first we identify entities associated with political parties and secondly we analyze the polarity of these entities in the tweets of each user.

We consider three types of entities: entities labeled by *Freeling* as proper names (i.e., comité_del_pp_de_madrid), Twitter users (i.e., @38congresopsoe), and Twitter hashtags (i.e., #upyd). Among all possible entities we selected those containing the acronym for a political party or the name of a political leader. A total of 864 entities related to political parties and political leaders were detected. Table 2 shows the parties and political leaders considered and some examples of the selected entities.

Party	**Tendency**	**Examples of Entities**
PP	right	#17congesoPP, congreso_nacional_pp, ppopular, congresopp, #ppfachas
PSOE	left	elpsoe, #adiosalpsoeenandalucia, #38congresopsoe
IU	left	asamblea_de_iu, iumalaga, diputados_de_iu, #iu
UPyD	centre	upydeuskadi, #demagogiaupyd, #mareamagenta, upyd_asturias
CiU	right	ciu+tripartito, #ciu, ciu-mintiendo-crujen

Political Leader	**Party**	**Examples of Entities**
Rajoy	PP	#rajoynoeslasolución, españa_de_rajoy, irpf_de_rajoy
González Pons	PP	@gonzalezpons, rajoy_para_gonzález_pons
Rubalcaba	PSOE	#rubalcabaenlaser, @conrubalcaba, rubalcaba_para_el_psoe
Zapatero	PSOE	nueva_via_de_zapatero, presidente_zapatero, zapatero_tv
Cayo Lara	IU	@cayo_lara, cayo_lara, cayo

Table 2: Tendency of political parties and political leaders.

We defined a tendency measure $Tendency$ that assigned a value of -1 to those entities related to left parties, a value of $+1$ to entities related to right parties and a value of 0 to the entities related to centre parties.

Next we show how has been numerically calculated the political orientation of users. For each user U_i of the General corpus we obtain the set T_i that includes all of their tweets that contain political entities. For users who do not have any tweet that contain political entities the *Undefined* label is assigned.

For each tweet $T_{i_j} \in T_i$, $j = 1 \cdots |T_i|$, we identify the political entities that are contained on it. Let E_{i_j} be the set of entities of the tweet T_{i_j}. We denote each of the entities contained in E_{i_j} as $E_{i_{j_k}} \in E_{i_j}, k = 1 \cdots |E_{i_j}|$.

We obtained the polarity of each entity by using the system described in section 4. After that, we assigned a numerical value to each polarity. In this respect, we assigned $Polarity = +1$ to the entities with positive polarity (label P), $Polarity = -1$ to the entities with negative polarity (label N) and finally, $Polarity = 0$ to the entities without polarity, that is, to the NEU and NONE labels.

We combined[7] the $Tendency$ and $Polarity$ measures previously presented to define a new measure ($Political\_Tendency$) to obtain the political orientation of each user.

[7]We have considered multiple combination strategies, in this work we present the combination with the best results.

$$Political\_Tendency(U_i) = \frac{\sum_{j=1\cdots|T_i|} \sum_{k=1\cdots|E_{i_j}|} Polarity(E_{i_{j_k}}) \cdot Tendency(E_{i_{j_k}})}{\sum_{j=1\cdots|T_i|} |E_{i_j}|} \quad (1)$$

From the $Political\_Tendency$ values obtained for each user, we classified the user tendency tanking into account the following: users without political entities in their tweets are classified as $Undefined$; users with $Political\_Tendency$ between -0.05 and +0.05 are classified as $Centre$; users with $Political\_Tendency$ lower than -0.05 are classified as $Left$; and users with $Political\_Tendency$ greater than +0.05 are classified as $Right$.

6 Experimental Evaluation of the Political Identification System

The measures selected to evaluate our approach were the Precision, the Recall, and the F-measure for $\beta = 1$ (F_1). Table 3 summarizes the experimental results of our proposal. The table includes both the overall results (*Global*) and the results for each one of the political tendencies (*Left*, *Right*, *Centre*, and *Undefined*). It also includes the distribution of the tendencies in the gold-standard (%Ref). For the global result, the precision and the recall are the same since each user in the test set had a tendency assigned and the task consist to assign a tendency to all the users.

Tendency	%Ref	Precision	Recall	F_1
Left	21.5	0.658	0.735	0.694
Centre	17.7	0.478	0.393	0.431
Right	39.9	0.786	0.698	0.739
Undefined	20.9	0.780	0.970	0.865
Global	100	0.709	0.709	0.709

Table 3: Experimental results obtained in the political tendency identification task of TASS2013.

The result obtained by our system (0.709) is the best result reported so far for this corpus, to our knowledge. The tendency for what we get better results is the *Undefined* (F_1=0.865). We consider the political tendency of a user to be *Undefined* if he did not have any tweet that references any of the majority parties. This assumption may be too strict for common users, but it seems reasonable for the well-know users that form the test corpus.

The tendency that our system had more trouble identifying was *Centre* (F_1=0.431). The tendency of a user can be identified as *Centre* when he expressed -in his tweets- opinions about entities related to centre parties, even when these opinions were negative. This is because the neutral value of *Centre* entities. In addition, users with opinions on right and left parties with the same polarity may be identified as *Centre*, which can be wrong in many cases.

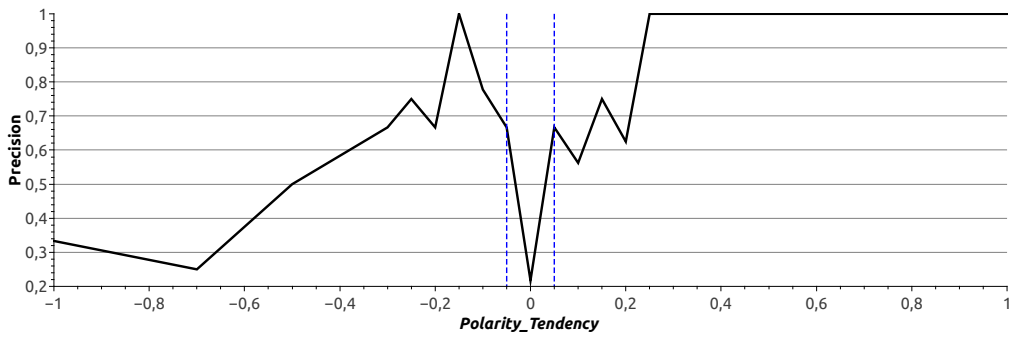


Figure 2: Precision results depending on the $Political\_Tendency$ assigned by the system.

Although it seems that the ability of our system to identify *Left* and *Right* tendencies was similar (F_1=0.694 for *Left* and F_1=0.739 for *Right*), analyzing the results considering the values of *Political_Tendency* some significant differences can be observed. Figure 2 shows the results, in terms of Precision, considering the value of *Political_Tendency* assigned to each user by our system, from a value of -1 (the maximum value for *Left*) to a value of +1 (the maximum for *Right*).

As expected, most identification errors occurred for *Political_Tendency* value near zero, should remember that values between -0.05 and 0.05 were considered *Centre*. Considering the *Right* tendency, all users that obtained *Political_Tendency* value greater than 0.25 were correctly identified as *Right*, performed better than would be expected. However, the behavior of the *Left* tendency was not symmetrical. It seems that values between -0.3 and -0.1 were better to determine correctly this tendency.

Although we have no clear explanation for this behavior, it could be due to multiple factors, including: the simplicity of the proposal, labeling errors in the polarity of certain entities, or the greater difficulty of numerically identify the *Left* tendency (at least in this corpus).

7 Conclusions

We have described our approach for political tendency identification of Twitter users. We have defined a metric, called *Political_Tendency*, that takes into account the polarity of entities related to political parties that appear in the tweets of the user. The Sentiment Analysis system developed in order to obtain the polarity of these entities was also presented.

The evaluation was performed using a corpus of Spanish tweets developed at TASS2013 workshop. This corpus was used for a specific political tendency identification task at this workshop. To our knowledge, the results obtained by our system are the best results published until now using this corpus.

We are very interested in SA tasks and in identifying tendencies in social media. In this sense, we have several ideas on how to improve our approach to identifying the political tendency in Twitter.

It would be interesting to test our approach using a larger corpus of tweets from normal user. We think that the characteristics of the users of the test corpus -figures of culture, journalism and politics in Spain- made the task a little easier. Perhaps the political tendency of ordinary users would be more difficult to identify. Moreover, the political spectrum would be more diverse and should increase the catalog of political parties. Moreover, the political spectrum would be more varied and, consequently, the catalog of political parties should be increased.

It should be emphasized the difficulty of building an annotated corpus of tweets that could be used to evaluate and compare different alternative systems. A great effort of acquisition of the tweets and a subsequent manual labeling process is required. In addition, a validation process is needed to correct the errors introduced by manual labeling. Even using crowdsourcing-based solutions it is a very expensive task both in money and time. In this context, to have a labeled corpus as the one provided by TASS2013 is a great help for the scientific community.

On the portability of the system, we think that it will be easy to adapt our proposal to another political context. This adaptation should focus on two different aspects. First, the Sentiment Analysis System should be adapted to a new language. In the case of languages with linguistic resources freely available the adaptation would be very simple. Second, political entities should be changed to fit the political context where we want to test the system. It would be sufficient to identify the most relevant parties and their leaders and classify them according to their political tendency. However, it is possible that in other political contexts different to Spanish, the *Left*, *Centre*, and *Right* tendencies also need to be adapted.

Finally, we have interest in using Machine Learning techniques for the task of identifying political tendency on twitter. On this point, we are working on a system in which *Political_Tendency*, as defined in this paper, will be a feature within a wider classification system. In this new system, we want to include additional information (not available in the TASS2013 corpus) about user behavior and Twitter structure in order to improve our approach.

Acknowledgements

This work has been partially funded by the Spanish MEC projects DIANA (TIN2012-38603-C02-01) and Tímpano (TIN2011-28169-C05-01).

References

Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 36–44. Association for Computational Linguistics.

Antoine Boutet, Hyoungshick Kim, and Eiko Yoneki. 2012. What's in Your Tweets? I Know Who You Supported in the UK 2010 General Election. In *The International AAAI Conference on Weblogs and Social Media (ICWSM)*, Dublin, Irlande, June.

Raviv Cohen and Derek Ruths. 2013. Classifying political orientation on twitter: It's not easy! In *International AAAI Conference on Weblogs and Social Media*.

M Conover, B Gonçalves, J Ratkiewicz, A Flammini, and F Menczer. 2011a. Predicting the political alignment of twitter users. In *Proceedings of 3rd IEEE Conference on Social Computing (SocialCom)*.

Michael Conover, Jacob Ratkiewicz, Matthew Francisco, Bruno Gonçalves, Alessandro Flammini, and Filippo Menczer. 2011b. Political polarization on twitter. In *Proc. 5th International AAAI Conference on Weblogs and Social Media (ICWSM)*.

Bernard J Jansen, Mimi Zhang, Kate Sobel, and Abdur Chowdury. 2009. Twitter power: Tweets as electronic word of mouth. *Journal of the American society for information science and technology*, 60(11):2169–2188.

Thorsten Joachims. 1998. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 137–142, Chemnitz, DE. Springer Verlag, Heidelberg, DE.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, pages 342–351, New York, NY, USA. ACM.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining. A Comprehensive Introduction and Survey*. Morgan & Claypool Publishers.

Eugenio Martínez-Cámara, M. Teresa Martín-Valdivia, L. Alfonso Ureña-López, and Arturo Montejo-Raéz. 2012. Sentiment analysis in twitter. *Natural Language Engineering*, 1(1):1–28.

Andrés Montoyo, Patricio Martínez-Barco, and Alexandra Balahur. 2012. Subjectivity and sentiment analysis: An overview of the current state of the area and envisaged developments. *Decision Support Systems*, 53(4):675–679.

Brendan O'Connor, Ramnath Balasubramanyan, Bryan R. Routledge, and Noah A. Smith. 2010a. From tweets to polls: Linking text sentiment to public opinion time series. In William W. Cohen and Samuel Gosling, editors, *ICWSM*. The AAAI Press.

Brendan O'Connor, Michel Krieger, and David Ahn. 2010b. Tweetmotif: Exploratory search and topic summarization for twitter. In William W. Cohen and Samuel Gosling, editors, *Proceedings of the Fourth International Conference on Weblogs and Social Media, ICWSM 2010, Washington, DC, USA, May 23-26, 2010*. The AAAI Press.

Lluís Padró and Evgeny Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May. ELRA.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *IN PROCEEDINGS OF EMNLP*, pages 79–86.

Marco Pennacchiotti and Ana-Maria Popescu. 2011. A machine learning approach to twitter user classification. In Lada A. Adamic, Ricardo A. Baeza-Yates, and Scott Counts, editors, *ICWSM*. The AAAI Press.

Veronica Perez-Rosas, Carmen Banea, and Rada Mihalcea. 2012. Learning sentiment lexicons in spanish. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying latent user attributes in twitter. In *Proceedings of the 2Nd International Workshop on Search and Mining User-generated Contents*, SMUC '10, pages 37–44, New York, NY, USA. ACM.

Xabier Saralegi and Iñaki San Vicente. 2013. Elhuyar at tass 2013. In *Proceedings of the TASS workshop at SEPLN 2013*. IV Congreso Español de Informática.

Bernhard Schölkopf, Alex J. Smola, Robert C. Williamson, and Peter L. Bartlett. 2000. New support vector algorithms. *Neural Comput.*, 12(5):1207–1245, May.

Peter D. Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *ACL*, pages 417–424.

Julio Villena-Román and Janine García-Morera. 2013. Workshop on sentiment analysis at sepln 2013: An over view. In *Proceedings of the TASS workshop at SEPLN 2013*. IV Congreso Español de Informática.

G Vinodhini and RM Chandrasekaran. 2012. Sentiment analysis and opinion mining: A survey. *International Journal*, 2(6).

Theresa Wilson, Paul Hoffmann, Swapna Somasundaran, Jason Kessler, Janyce Wiebe, Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005. Opinionfinder: A system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, pages 34–35. Association for Computational Linguistics.

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Sara Rosenthal, Veselin Stoyanov, and Alan Ritter. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. *Proceedings of the International Workshop on Semantic Evaluation, SemEval*, 13.

Felix Ming Fai Wong, Chee Wei Tan, Soumya Sen, and Mung Chiang. 2013. Quantifying political leaning from tweets and retweets. In Emre Kiciman, Nicole B. Ellison, Bernie Hogan, Paul Resnick, and Ian Soboroff, editors, *ICWSM*. The AAAI Press.

A Study of using Syntactic and Semantic Structures for Concept Segmentation and Labeling

Iman Saleh[*], **Shafiq Joty, Lluís Màrquez, Alessandro Moschitti, Preslav Nakov**
ALT Research Group
Qatar Computing Research Institute
{sjoty,lmarquez,amoschitti,pnakov}@qf.org.qa

Scott Cyphers, Jim Glass
MIT CSAIL
Cambridge, Massachusetts 02139
USA
{cyphers,glass}@mit.edu

Abstract

This paper presents an empirical study on using syntactic and semantic information for Concept Segmentation and Labeling (CSL), a well-known component in spoken language understanding. Our approach is based on reranking N-best outputs from a state-of-the-art CSL parser. We perform extensive experimentation by comparing different tree-based kernels with a variety of representations of the available linguistic information, including semantic concepts, words, POS tags, shallow and full syntax, and discourse trees. The results show that the structured representation with the semantic concepts yields significant improvement over the base CSL parser, much larger compared to learning with an explicit feature vector representation. We also show that shallow syntax helps improve the results and that discourse relations can be partially beneficial.

1 Introduction

Spoken Language Understanding aims to interpret user utterances and to convert them to logical forms, or, equivalently, database queries, which can then be used to satisfy the user's information needs. This process is known as Concept Segmentation and Labeling (CSL): it maps utterances into meaning representations based on semantic constituents. The latter are basically sequences of semantic entities, often referred to as concepts, attributes or semantic tags. Traditionally, grammar-based methods have been used for CSL, but more recently machine learning approaches to semantic structure computation have been shown to yield higher accuracy. However, most previous work did not exploit syntactic/semantic structures of the utterances, and the state-of-the-art is represented by conditional models for sequence labeling, such as Conditional Random Fields (Lafferty et al., 2001) trained with simple morphological and lexical features. In our study, we measure the impact of syntactic and discourse structures by also combining them with innovative features. In the following subsections, we present the application context for our CSL task and then we outline the challenges and the findings of our research.

1.1 Semantic parsing for the "restaurant" domain

We experiment with the dataset of McGraw et al. (2012), containing spoken and typed questions about restaurants, which are to be answered using a database of free text such as reviews, categorical data such as names and locations, and semi-categorical data such as user-reported cuisines and amenities.

Semantic parsing, in the form of sequential segmentation and labeling, makes it easy to convert spoken and typed questions such as "cheap lebanese restaurants in doha with take out" into database queries. First, a language-specific semantic parser tokenizes, segments and labels the question:

[*Price* cheap] [*Cuisine* lebanese] [*Other* restaurants in] [*City* doha] [*Other* with] [*Amenity* take out]

Then, label-specific normalizers are applied to the segments, with the option to possibly relabel mislabeled segments; at this point, discourse history may be incorporated as well.

[*Price* low] [*Cuisine* lebanese] [*City* doha] [*Amenity* carry out]

[*] Iman Saleh (iman.saleh@fci-cu.edu.eg) is affiliated to Faculty of Computers and Information, Cairo University.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 193–202, Dublin, Ireland, August 23-29 2014.

Finally, a database query is formed from the list of labels and values, and is then executed against the database, e.g., MongoDB; a backoff mechanism may be used if the query does not succeed.

```
{$and [{cuisine:"lebanese"}, {city:"doha"},  {price:"low"},  {amenity:"carry out"}]}
```

1.2 Related work on CSL

Pieraccini et al. (1991) used Hidden Markov Models (HMMs) for CSL, where the observations were word sequences and the hidden states were meaning units, i.e, *concepts*. In subsequent work (Rubinstein and Hastie, 1997; Santafé et al., 2007; Raymond and Riccardi, 2007; De Mori et al., 2008), other generative models were applied, which model the joint probability of a word sequence and a concept sequence, as well as discriminative models, which directly model a conditional probability over the concepts in the input text.

Seneff (1989) and Miller et al. (1994) used stochastic grammars for CSL. In particular, they applied stochastic Finite State Transducers (FST) for recognizing constituent annotations. FSTs describe local syntactic structures with a sequence of words, e.g., noun phrases or even constituents. Papineni et al. (1998) proposed and evaluated exponential models, but, nowadays, Conditional Random Fields (Lafferty et al., 2001) are considered to be the state-of-the-art. More recently, Wang et al. (2009) illustrated an approach for CSL that is specific to query understanding for web applications. A general survey of CSL approaches can be found in (De Mori et al., 2008). CSL is also connected to a large body of work on shallow semantic parsing; see (Gildea and Jurafsky, 2002; Màrquez et al., 2008) for an overview.

Another relevant line of research with a considerable body of work is reranking in NLP. Tree kernels for reranking syntactic parse trees were first proposed in (Collins and Duffy, 2002). Some variants used explicit spaces (Kudo et al., 2005), and feature vector approaches were proposed in (Koo and Collins, 2005). Other reranking work using tree kernels regards predicate argument structures (Moschitti et al., 2006) and named entities (Nguyen and Moschitti, 2012). In (Dinarelli et al., 2011), we rerank CSL hypotheses using structures built on top of concepts, words and features that are simpler than those studied in this paper. The work of Ge and Mooney (2006) and Kate and Mooney (2006) is also similar to ours, as it models the extraction of semantics as a reranking task using string kernels.

1.3 Syntactic and semantic structures for CSL

The related work has highlighted that automatic CSL is mostly based on powerful machine learning algorithms and simple feature representations based on word and tag n-grams. In this paper, we study the impact of more advanced linguistic processing on CSL, such as shallow and full syntactic parsing and discourse structure. We use a reranking approach to select the best hypothesis annotated with concepts derived by a local model, where the hypotheses are represented as trees enriched with semantic concepts similarly to (Dinarelli et al., 2011). These tree-based structures can capture dependencies between sentence constituents and concepts. However, extracting features from them is rather difficult as their number is exponentially large. Thus, we rely on structural kernels (e.g., see (Moschitti, 2006)) for automatically encoding tree fragments, which represent syntactic and semantic dependencies from words and concepts, and we train the reranking functions with Support Vector Machines (e.g., see (Joachims, 1999)). Additionally, we experiment with several types of kernels and newly designed feature vectors.

We test our models on the above-mentioned *Restaurant* domain. The results show that *(i)* the basic CRF model, in fact semi-CRF (see below), is very accurate, achieving more than 83% in F_1-score, which indicates that improving over the semi-CRF approach is very hard; *(ii)* the upper-bound performance of the reranking approach is very high as well, i.e., the correct annotation is generated in the first 100 hypotheses in 98.72% of the cases; *(iii)* our feature vectors show improvement only when all feature groups are used together; otherwise, we only observe marginal improvement; *(iv)* structural kernels yield a 10% relative error reduction from the semi-CRF baseline, which is more than double the feature vector result; *(v)* syntactic information significantly improves on the best model, but only when using shallow syntax; and finally, *(vi)* although, discourse structures provide good improvement over the semi-CRF model, they perform lower than shallow syntax (thus, a valuable use of discourse features is still an open problem that we plan to pursue in future work).

2 CSL reranking

Reranking is based on a list of N annotation hypotheses, which are generated and sorted by probability using local classifiers. Then a reranker, typically a meta-classifier, tries to select the best hypothesis from the list. The reranker can exploit global information, and, specifically, the dependencies between the different concepts that are made available by the local model. We use semi-CRF as our local model since it yields the highest accuracy in CSL (when using a single model), and preference reranking with kernel machines to rerank the N hypotheses generated by the semi-CRF.

2.1 Basic parser using semi-CRF

We use a semi-Markov CRF (Sarawagi and Cohen, 2004), or semi-CRF, a variation of a linear-chain CRF (Lafferty et al., 2001), to produce the N-best list of labeled segment hypotheses that serve as the input to reranking. In a linear-chain CRF, with a sequence of tokens x and labels y, we approximate $p(y|x)$ as a product of factors of the form $p(y_i|y_{i-1}, x)$, which corresponds to features of the form $f_j(y_{i-1}, y_i, i, x)$, where i iterates over the token/label positions. This supports a Viterbi search for the approximate N best values of y. With M label values, if for each label y_m we know the best N sequences of labels $y_1, y_2, \ldots, y_{i-1} = y_m$, then we can use $p(y_i|y_{i-1}, x)$ to get the probability for extending each path by each possible label $y_i = y'_m$. Then for each label y'_m, we will have MN paths and scores, one from each of the paths of length $i-1$ ending with y_m. For each y'_m, we pick the N best extended paths.

With semi-CRF, we want a labeled segmentation s rather than a sequence of labels. Each segment $s_i = (y_i, t_i, u_i)$ has a label y_i as well as a starting and ending token position for the segment, t_i and u_i respectively, where $u_i + 1 = t_{i+1}$. We approximate $p(s|x)$, with factors of the form $p(s_i|s_{i-1}, x)$, which we simplify to $p(y_i, u_i|y_{i-1}, t_i)$, so features take the form $f_j(y_{i-1}, y_i, t_i, u_i)$, i.e., they can use the previous segment's label and the current segment's label and endpoints. The Viterbi search is extended to search for a pair of label and segment end. Whereas for M labels we kept track of MN paths, we must keep track of MLN paths, where L is the maximum segment length.

We use token n-gram features relative to the segment boundaries, n-grams within the segment, token regular expression and lexicon features within a segment. Each of these features also includes the labels of the previous and current segment, and the segment length.

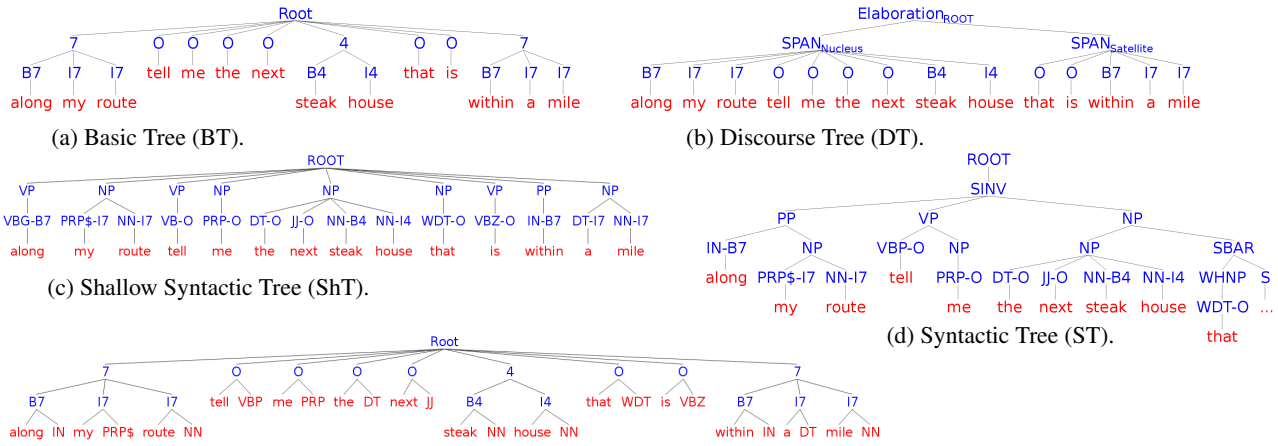
2.2 Preference reranking with kernel machines

Preference reranking (PR) uses a classifier $\mathcal{C}$ of pairs of hypotheses $\langle H_i, H_j \rangle$, which decides if H_i is better than H_j. Given each training question Q, positive and negative examples are generated for training the classifier. We adopt the following approach for example generation: the pairs $\langle H_1, H_i \rangle$ constitute positive examples, where H_1 has the lowest error rate with respect to the gold standard among the hypotheses for Q, and vice versa, $\langle H_i, H_1 \rangle$ are considered as negative examples. At testing time, given a new question Q′, $\mathcal{C}$ classifies all pairs $\langle H_i, H_j \rangle$ generated from the annotation hypotheses of Q′: a positive classification is a vote for H_i, otherwise the vote is for H_j. Also, the classifier score can be used as a weighted vote. H_k are then ranked according to the number (sum) of the (weighted) votes they get.

We build our reranker with kernel machines. The latter, e.g., SVMs, classify an input object o using the following function: $\mathcal{C}(o) = \sum_i \alpha_i y_i K(o, o_i)$, where α_i are model parameters estimated from the training data, o_i are support objects and y_i are the labels of the support objects. $K(\cdot, \cdot)$ is a kernel function, which computes the scalar product between the two objects in an implicit vector space. In the case of the reranker, the objects o are $\langle H_i, H_j \rangle$, and the kernel is defined as follow:

$K(\langle H_1, H_2 \rangle, \langle H'_1, H'_2 \rangle) = S(H_1, H'_1) + S(H_2, H'_2) - S(H_1, H'_2) - S(H_2, H'_1)$.

Our reranker also includes traditional feature vectors in addition to the trees. Therefore, we define each hypothesis H as a tuple $\langle T, \vec{v} \rangle$ composed of a tree T and a feature vector $\vec{v}$. We then define a structural kernel (similarity) between two hypotheses H and H' as follows: $S(H, H') = S_{\text{TK}}(T, T') + S_{\text{v}}(\vec{v}, \vec{v}')$, where S_{TK} is one of the tree kernel functions defined in Section 3.1, and S_{v} is a kernel over feature vectors (see Section 3.3), e.g., linear, polynomial, gaussian, etc.



(a) Basic Tree (BT).

(b) Discourse Tree (DT).

(c) Shallow Syntactic Tree (ShT).

(d) Syntactic Tree (ST).

(e) BT with POS (BTP).

Figure 1: Syntactic/semantic trees. The numeric semantic tagset is defined in Table 7.

3 Structural kernels for semantic parsing

In this section, we briefly describe the kernels we use in $S(H, H')$ for preference reranking. We engineer them by combining three aspects: *(i)* different types of existing tree kernels, *(ii)* new syntactic/semantic structures for representing CSL, and *(iii)* new feature vectors.

3.1 Tree kernels

Structural kernels, e.g., tree and sequence kernels, measure the similarity between two structures in terms of their shared substructures. One interesting aspect is that these kernels correspond to a scalar product in the fragment space, where each substructure is a feature. Therefore, they can be used in the training and testing algorithms of kernel machines (see Section 2.2). Below, we briefly describe different types of kernels we tested in our study, which are made available in the SVM-Light-TK toolkit (Moschitti, 2006).

Subtree Kernel (K0) is one of the simplest tree kernels, as it only generates complete subtrees, i.e., tree fragments that, given any arbitrary starting node, necessarily include all its descendants.

Syntactic Tree Kernel (K1), also known as a subset tree kernel (Collins and Duffy, 2002), maps objects in the space of all possible tree fragments constrained by the rule that the sibling nodes cannot be separated from their parents. In other words, substructures are composed of atomic building blocks corresponding to nodes, along with all of their direct children. In the case of a syntactic parse tree, these are complete production rules for the associated parser grammar.

Syntactic Tree Kernel + BOW (K2) extends ST by allowing leaf nodes to be part of the feature space. The leaves of the trees correspond to words, i.e., we allow bag-of-words (BOW).

Partial Tree Kernel (K3) can be effectively applied to both constituency and dependency parse trees. It generates all possible connected tree fragments, e.g., sibling nodes can be also separated and be part of different tree fragments. In other words, a fragment is any possible tree path from whose nodes other tree paths can depart. Thus, it can generate a very rich feature space.

Sequence Kernel (K4) is the traditional string kernel applied to the words of a sentence. In our case, we apply it to the sequence of concepts.

3.2 Semantic/syntactic structures

As mentioned before, tree kernels allow us to compute structural similarities between two trees without explicitly representing them as feature vectors. For the CSL task, we experimented with a number of tree representations that incorporate different levels of syntactic and semantic information.

To capture the structural dependencies between the semantic tags, we use a basic tree (Figure 1a) where the words of a sentence are tagged with their semantic tags. More specifically, the words in the sentence constitute the leaves of the tree, which are in turn connected to the pre-terminals containing the semantic tags in BIO notation ('B'=*begin*, 'I'=*inside*, 'O'=*outside*). The BIO tags are then generalized in the upper level, and so on. The basic tree does not include any syntactic information.

However, part-of-speech (POS) and phrasal information could be informative for both segmentation and labeling in semantic parsing. To incorporate this information, we use two extensions of the basic tree: one that includes the POS tags of the words (Figure 1e), and another one that includes both POS tags and syntactic chunks (Figure 1c). The POS tags are children of the semantic tags, whereas the chunks (i.e., phrasal information) are included as parents of the semantic tags.

We also experiment with full syntactic trees (Figure 1d) to see the impact of deep syntactic information. The semantic tags are attached to the pre-terminals (i.e., POS tags) in the syntactic tree. We use the Stanford POS tagger and syntactic parser and the Twitter NLP tool[1] for the shallow trees.

A sentence containing multiple clauses exhibits a coherence structure. For instance, in our example, the first clause *"along my route tell me the next steak house"* is *elaborated* by the second clause *"that is within a mile"*. The relations by which clauses in a text are linked are called *coherence* relations (e.g., *Elaboration, Contrast*). Discourse structures capture this coherence structure of text and provide additional semantic information that could be useful for the CSL task (Stede, 2011). To build the discourse structure of a sentence, we use a state-of-the-art discourse parser (Joty et al., 2012) which generates discourse trees in accordance with the Rhetorical Structure Theory of discourse (Mann and Thompson, 1988), as exemplified in Figure 1b. Notice that a text span linked by a coherence relation can be either a *nucleus* (i.e., the core part) or a *satellite* (i.e., a supportive one) depending on how central the claim is.

3.3 New features

In order to compare to the structured representation, we also devoted significant effort towards engineering a set of features to be used in a flat feature-vector representation; they can be used in isolation or in combination with the kernel-based approach (as a composite kernel using a linear combination):

CRF-based: these include the basic features used to train the initial semi-CRF model (cf. Section 2.1).

n-gram based: we collected 3- and 4-grams of the output label sequence at the level of concepts, with artificial tags inserted to identify the start ('S') and end ('E') of the sequence.[2]

Probability-based: two features computing the probability of the label sequence as an average of the probabilities at the word level $p(l_i|w_i)$ (i.e., assuming independence between words). The *unigram* probabilities are estimated by frequency counts using maximum likelihood in two ways: *(i)* from the complete 100-best list of hypotheses; *(ii)* from the training set (according to the gold standard annotation).

DB-based: a single feature encoding the number of results returned from the database when constructing a query using the conjunction of all semantic segments in the hypothesis. Three possible values are considered by using a threshold t: 0 (if the query result is void), 1 (if the number of results is in $[1, t]$), and 2 (if the number of results is greater than t). In our case, t is empirically set to 10,000.

4 Experiments

The experiments aim at investigating which structures, and thus which linguistic models and combination with other models, are the most appropriate for our reranker. We first calculate the *oracle* accuracy in order to compute an upper bound of the reranker. Then we present experiments with the feature vectors, tree kernels, and representations of linguistic information introduced in the previous sections.

4.1 Experimental setup

In our experiments, we use questions annotated with semantic tags in the restaurant domain,[3] which were collected by McGraw et al. (2012) through crowdsourcing on Amazon Mechanical Turk.[4] We split the dataset into training, development and test sets. Table 1 shows statistics about the dataset and about the size of the parts we used for training, development and testing (see the semi-CRF line).

We subsequently split the training data randomly into ten folds. We generated the N-best lists on the training set in a cross-validation fashion, i.e., iteratively training on nine folds and annotating the remaining fold. We computed the 100-best hypotheses for each example.

[1] Available from *http://nlp.stanford.edu/software/index.shtml* and *https://github.com/aritter/twitter_nlp*, respectively.

[2] For instance, if the output sequence is *Other-Rating-Other-Amenity* the 3-gram patterns would be: S-*Other-Rating, Other-Rating-Other, Rating-Other-Amenity*, and *Other-Amenity*-E.

[3] *http://www.sls.csail.mit.edu/downloads/restaurant*

[4] We could not use the datasets used by Dinarelli et al. (2011), because they use French and Italian corpora for which there are no reliable syntactic and discourse parsers.

	Train	Devel.	Test	Total
semi-CRF	6,922	739	1,521	9,182
Reranker	28,482	3,695	7,605	39,782

Table 1: Number of instances and pairs used to train the semi-CRF and rerankers, respectively.

N	1	2	**5**	10	100
F_1	83.03	87.76	**92.63**	95.23	98.72

Table 2: Oracle F_1-score for N-best lists of different lengths.

We used the development set to experiment and tune the hyper-parameters of the reranking model. The results on the development set presented in Section 4.2 were obtained by semi-CRF and reranking models learned on the training set. The results on the test set were obtained by models trained on the training plus development sets. Similarly, the N-best lists for the development and test sets were generated using a single semi-CRF model trained on the training set and the training+development sets, respectively.

Each generated hypothesis is represented using a semantic tree and a feature vector (explained in Section 3) and two extra features accounting for *(i)* the semi-CRF probability of the hypothesis, and *(ii)* the hypothesis reciprocal rank in the N-best list. SVM-Light-TK[5] is used to train the reranker with a combination of tree kernels and feature vectors (Moschitti, 2006; Joachims, 1999). Although we tried several parameters on the validation set, we observed that the default values yielded the highest results. Thus, we used the default c (trade-off) and tree kernel parameters and a linear kernel for the feature vectors. Table 1 shows the sizes of the train, the development and the test sets used for the semi-CRF as well as the number of pairs generated for the reranker. As a baseline, we picked the best-scored hypothesis in the list, according to the semi-CRF tagger. The evaluation measure used in all the experiments is the harmonic mean of precision and recall, i.e., the F_1-score (van Rijsbergen, 1979), computed at the token level and micro-averaged over the different semantic types.[6] We used paired t-test to measure the statistical significance of the improvements: we split the test set into 31 equally-sized samples and performed t-tests based on the F_1-scores of different models on the resulting samples.

4.2 Results

Oracle accuracy. Table 2 shows the oracle F_1-score for N-best lists of different lengths, i.e., which can be achieved by picking the best candidate of the N-best list for various values of N. We can see that going to 5-best increases the oracle F_1-score by almost ten points absolute. Going down to 10-best only adds 2.5 extra F_1 points absolute, and a 100-best list adds 3.5 F_1 points more to yield a respectable F_1-score of 98.72. This high result can be explained considering that the size of the complete hypothesis set is smaller than 100 for most questions. Thus, we can conclude that the N-best lists do include many good options and do offer quite a large space for potential improvement. We can further observe that going to 5-best lists offers a good balance between the length of the list and the possibility to improve F_1-score: generally, we do not want too long N-best lists since they slow down computation and also introduce more opportunities to make the wrong choice for a reranker (since there are just more candidates to choose from). In our experiments with larger N, we observed improvements only for 10 and only on the development set; thus, we will focus on 5-best lists in our experiments below.

	K0	K1	K2	K3	K4
Dev	84.21	82.92	83.07	**85.07**	83.78
Test	84.08	83.19	83.20	**84.61**	82.93

Table 3: Results for using different tree kernels on the basic tree (BT) representation.

Choosing the best tree kernel. We first select the most appropriate tree kernel to limit the number of experiment variables. Table 3 shows the results of different tree kernels using the basic tree (BT) representation (see Figure 1a). We can observe that for both the development set and the test set, kernel K3 (see Section 3.1) yields the highest F_1-score.

Impact of feature vectors. Table 4 presents the results for the feature vector experiments in terms of F_1-scores and relative error reductions (row RER). The first column shows the baseline, when no reranking is used; the following four columns contain the results when using vectors including different

[5] http://disi.unitn.it/moschitti/Tree-Kernel.htm

[6] 'Other' is not considered a semantic type, thus 'Other' tokens are not included in the F_1 calculation.

	Baseline	n-grams	CRF features	Count	DB	ProbBased	AllFeat
Dev	83.86	83.79	**83.96**	83.80	83.86	83.87	**84.49**
	RER	-0.4	**0.6**	-0.4	0.0	0.0	**3.9**
Test	83.03	82.90	**83.44**	82.90	83.01	83.09	**83.86**
	RER	-0.7	**2.4**	-0.7	-0.1	0.3	**4.8**

Table 4: Feature vector experiments: F_1 score and relative error reduction (in %).

	Baseline	BT	BTP	ShT	ST	AllFeat	Combining AllFeat and +BT +ShT	+ShT	+BT
Dev	83.86	85.07	**85.41**	**85.06**	84.30	84.49	85.57	**85.58**	85.33
	RER	7.5	**9.6**	**7.4**	2.8	3.9	10.6	**10.7**	9.1
Test	83.03	84.61	**84.63**	**84.07**	83.81	83.86	84.67	**84.79**	84.76
	RER	9.3	**9.4**	**6.1**	4.5	4.8	9.6	**10.2**	10.2
	p.v.	0.00049	0.0002	0.012	0.032	0.00018	0.00028	0.00004	0.000023

Table 5: Tree kernel experiments: F_1-score, relative error reduction (in %), and p-values.

kinds of features: *(i)* n-gram features, *(ii)* all features used by the semi-CRF, *(iii)* count features, and *(iv)* database (DB) features. In each case, we include two additional features: the semi-CRF score (i.e., the probability) and the reciprocal rank of the hypothesis in the N-best list. Among *(i)–(iv)*, only the semi-CRF features seem to help; the rest either show no improvements or degrade the performance. However, putting all these features together (AllFeat) yields sizable gains in terms of F_1-score and a relative error reduction of 4-5%; the improvement is statistically significant, and it is slightly larger on the test dataset compared to the development dataset.

Impact of structural kernels and combinations. Table 5 shows the results when experimenting with various tree structures (see columns 2-5): *(i)* the basic tree (BT), *(ii)* the basic tree augmented with part-of-speech information (BTP), *(iii)* shallow syntactic tree (ShT), and *(iv)* syntactic tree (ST). We can see that the basic tree works rather well, yielding +1.6 F_1-score on the test dataset, but adding POS information can help a bit more, especially for the tuning dataset. Interestingly, the syntactic tree kernels, ShT and ST, perform worse than BT and BTP, especially on the test dataset. The last three columns in the table show the results when we combine the AllFeat feature vector (see Table 4) with BT and ShT. We can see that combining AllFeat with ShT works better, on both development and test sets, than combining it with BT or with both ShT and BT. Also note the big jump in performance from AllFeat to AllFeat+ShT. Overall, we can conclude that shallow syntax has a lot to offer over AllFeat, and it is preferable over BT in the combination with AllFeat. The improvements reported in Tables 5 and 6 are statistically significant when compared to the semi-CRF baseline as shown by the p.v. (value) row. Moreover, the improvement of AllFeat + ShT over BT is also statistically significant (p.v.$<$0.05).

	Baseline	DS	Combining AllFeat and +DS	+DS +BT	+DS +ShT
Dev	83.86	84.61	85.14	85.43	**85.46**
	RER	4.7	7.9	9.7	**9.9**
Test	83.03	84.38	84.55	84.63	**84.67**
	RER	7.9	8.9	9.4	**9.6**
	p.v.	0.0005	0.0001	0.00066	0.00015

Table 6: Experiments with discourse kernels: F_1 score, relative error reduction (in %), and p-values.

Discourse structure. Finally, Table 6 shows the results for the discourse tree kernel (DS), which we designed and experimented with for the first time in this paper. We see that DS yields sizable improvements over the baseline. We also see that further gains can be achieved by combining DS with AllFeat, and also with BT and ShT, the best combination being AllFeat+DS+ShT (see last column). However, comparing to Table 5, we see that it is better to use just AllFeat+ShT and leave DS out. We would like to note though that the discourse parser produced non-trivial trees for only 30% of the hypotheses (due to the short, simple nature of the questions); in the remaining cases, it probably hurt rather than helped. We conclude that discourse structure has clear potential, but how to make best use of it, especially in the case of short simple questions, remains an open question that deserves further investigation.

Tag ID		Other	Rating	Restaurant	Amenity	Cuisine	Dish	Hours	Location	Price
0	Other	8260	35	43	110	15	19	55	113	9
1	Rating	29	266	0	**14**	3	6	0	0	8
2	Restaurant	72	6	657	**20**	19	15	0	5	0
3	Amenity	117	9	10	841	**27**	**27**	7	12	7
4	Cuisine	36	2	12	**26**	543	**44**	3	1	0
5	Dish	23	0	4	**20**	**33**	324	1	4	0
6	Hours	61	0	1	2	6	1	426	9	1
7	Location	104	1	14	**20**	2	1	1	1457	0
8	Price	22	1	0	7	0	2	0	1	204

Table 7: Confusion matrix for the output of the best performing system.

4.3 Error analysis and discussion

Table 7 shows the confusion matrix for our best-performing model *AllFeat+ShT* (rows = gold standard tags; columns = system predicted tags). Given the good results of the semantic parser, the numbers in the diagonal are clearly dominating the weight of the matrix. The largest errors correspond to missed (first column) and over-generated (first row) entity tokens. Among the proper confusions between semantic types, *Dish* and *Cuisine* tend to mislead each other most. This is due to the fact that these two tags are semantically similar, thus making them hard to distinguish. We can also notice that it is difficult to identify *Amenity* correctly, and the model mistakenly tags many other tags as *Amenity*. We looked into some examples to further investigate the errors. Our findings are as follow:

Inaccuracies and inconsistencies in human annotations. Since the annotations were done in Mechanical Turk, they have many inaccuracies and inconsistencies. For example, the word *good* with exactly the same sense was tagged as both *Other* and *Rating* by the Turkers in the following examples:

Gold: [$_{Other}$ any good] [$_{Price}$ cheap] [$_{Cuisine}$ german] [$_{Other}$ restaurants] [$_{Location}$ nearby]
Model: [$_{Other}$ any] [$_{Rating}$ good] [$_{Price}$ cheap] [$_{Cuisine}$ german] [$_{Other}$ restaurants] [$_{Location}$ nearby]

Gold: [$_{Other}$ any place] [$_{Location}$ along the road] [$_{Other}$ has a] [$_{Rating}$ good] [$_{Dish}$ beer] [$_{Other}$ selection that also serves] ...

Requires lexical semantics and more coverage. In some cases our model fails to generalize well. For instance, it fails to correctly tag *establishments* and *tameles* for the following examples. This suggests that we need to consider other forms of semantic information, e.g., distributional and compositional semantics computed from large corpora and/or using Web resources such as Wikipedia.

Gold: [$_{Other}$ any] [$_{Location}$ dancing establishments] [$_{Other}$ with] [$_{Price}$ reasonable] [$_{Other}$ pricing]
Model: [$_{Other}$ any] [$_{Amenity}$ dancing] [$_{Other}$ establishments] [$_{Other}$ with] [$_{Price}$ reasonable] [$_{Other}$ pricing]

Gold: [$_{Other}$ any] [$_{Cuisine}$ mexican] [$_{Other}$ places have a] [$_{Dish}$ tameles] [$_{Amenity}$ special today]
Model: [$_{Other}$ any] [$_{Cuisine}$ mexican] [$_{Other}$ places have a] [$_{Amenity}$ tameles] [$_{Other}$ special] [$_{Hours}$ today]

5 Conclusions

We have presented a study on the usage of syntactic and semantic structured information for improved Concept Segmentation and Labeling (CSL). Our approach is based on reranking a set of N-best sequences generated by a state-of-the-art semi-CRF model for CSL. The syntactic and semantic information was encoded in tree-based structures, which we used to train a reranker with kernel-based Support Vector Machines. We empirically compared several variants of syntactic/semantic structured representations and kernels, including also a vector of manually engineered features.

The first and foremost conclusion from our study is that structural kernels yield significant improvement over the strong baseline system, with a relative error reduction of $\sim 10\%$. This more than doubles the improvement when using the explicit feature vector. Second, we observed that shallow syntactic information also improves results significantly over the best model. Unfortunately, the results obtained using full syntax and discourse trees are not so clear. This is probably explained by the fact that user queries are rather short and linguistically not very complex. We also observed that the upper bound performance for the reranker still leaves large room for improvement. Thus, it remains to be seen whether some alternative kernel representations can be devised to make better use of discourse and other syntactic/semantic information. Also, we think that some innovative features based on analyzing the results obtained from our database (or the Web) when querying with the segments represented in each hypotheses have the potential to improve the results. All these concerns will be addressed in future work.

Acknowledgments

This research is developed by the Arabic Language Technologies (ALT) group at Qatar Computing Research Institute (QCRI) within the Qatar Foundation in collaboration with MIT. It is part of the Interactive sYstems for Answer Search (Iyas) project.

References

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 263–270, Philadelphia, PA, USA.

Renato De Mori, Dilek Hakkani-Tür, Michael McTear, Giuseppe Riccardi, and Gokhan Tur. 2008. Spoken language understanding: a survey. *IEEE Signal Processing Magazine*, 25:50–58.

Marco Dinarelli, Alessandro Moschitti, and Giuseppe Riccardi. 2011. Discriminative reranking for spoken language understanding. *IEEE Transactions on Audio, Speech and Language Processing*, 20(2):526–539.

Ruifang Ge and Raymond Mooney. 2006. Discriminative reranking for semantic parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, COLING-ACL'06, pages 263–270, Sydney, Australia.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Thorsten Joachims. 1999. Advances in kernel methods. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Making Large-scale Support Vector Machine Learning Practical*, pages 169–184, Cambridge, MA, USA. MIT Press.

Shafiq Joty, Giuseppe Carenini, and Raymond Ng. 2012. A novel discriminative framework for sentence-level discourse analysis. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 904–915, Jeju Island, Korea.

Rohit Kate and Raymond Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, COLING-ACL '06, pages 913–920, Sydney, Australia.

Terry Koo and Michael Collins. 2005. Hidden-variable models for discriminative reranking. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT-EMNLP '05, pages 507–514, Vancouver, British Columbia, Canada.

Taku Kudo, Jun Suzuki, and Hideki Isozaki. 2005. Boosting-based parse reranking with subtree features. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 189–196, Ann Arbor, MI, USA.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, ICML '01, pages 282–289, Williamstown, MA, USA.

William Mann and Sandra Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Lluís Màrquez, Xavier Carreras, Kenneth Litkowski, and Suzanne Stevenson. 2008. Semantic Role Labeling: An Introduction to the Special Issue. *Computational Linguistics*, 34(2):145–159.

Ian McGraw, Scott Cyphers, Panupong Pasupat, Jingjing Liu, and Jim Glass. 2012. Automating crowd-supervised learning for spoken language systems. In *Proceedings of 13th Annual Conference of the International Speech Communication Association*, INTERSPEECH '12, Portland, OR, USA.

Scott Miller, Richard Schwartz, Robert Bobrow, and Robert Ingria. 1994. Statistical language processing using hidden understanding models. In *Proceedings of the workshop on Human Language Technology*, HLT '94, pages 278–282, Morristown, NJ, USA.

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2006. Semantic role labeling via tree kernel joint inference. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 61–68, New York City, June.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of the 17th European Conference on Machine Learning*, ECML'06, pages 318–329, Berlin, Heidelberg. Springer-Verlag.

Truc-Vien T. Nguyen and Alessandro Moschitti. 2012. Structural reranking models for named entity recognition. *Intelligenza Artificiale*, 6(2):177–190.

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

Roberto Pieraccini, Esther Levin, and Chin-Hui Lee. 1991. Stochastic representation of conceptual structure in the ATIS task. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '91, pages 121–124, Pacific Grove, CA, USA.

Christian Raymond and Giuseppe Riccardi. 2007. Generative and discriminative algorithms for spoken language understanding. In *Proceedings of 8th Annual Conference of the International Speech Communication Association*, INTERSPEECH '07, pages 1605–1608, Antwerp, Belgium.

Yigal Dan Rubinstein and Trevor Hastie. 1997. Discriminative vs informative learning. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, KDD '97, pages 49–53, Newport Beach, CA, USA.

Guzmán Santafé, Jose Lozano, and Pedro Larrañaga. 2007. Discriminative vs. generative learning of Bayesian network classifiers. *Lecture Notes in Computer Science*, 4724:453–464.

Sunita Sarawagi and William Cohen. 2004. Semi-Markov conditional random fields for information extraction. In *Proceedings of the 18th Annual Conference on Neural Information Processing Systems*, NIPS '04, pages 1185–1192, Vancouver, British Columbia, Canada.

Stephanie Seneff. 1989. TINA: A probabilistic syntactic parser for speech understanding systems. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '89, pages 168–178, Philadelphia, PA, USA.

Manfred Stede. 2011. *Discourse Processing*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool Publishers.

Cornelis Joost van Rijsbergen. 1979. *Information Retrieval*. Butterworth.

Ye-Yi Wang, Raphael Hoffmann, Xiao Li, and Jakub Szymanski. 2009. Semi-supervised learning of semantic classes for query understanding: from the web and for the web. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 37–46, New York, NY, USA.

Time-aware Personalized Hashtag Recommendation on Social Media

Qi Zhang, Yeyun Gong, Xuyang Sun, Xuanjing Huang
Shanghai Key Laboratory of Intelligent Information Processing
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, P.R.China
{qz, 12110240006, 13210240106, xjhuang}@fudan.edu.cn

Abstract

The task of recommending hashtags for microblogs has been received considerable attention in recent years, and many applications can reap enormous benefits from it. Various approaches have been proposed to study the problem from different aspects. However, the impacts of temporal and personal factors have rarely been considered in the existing methods. In this paper, we propose a novel method that extends the translation based model and incorporates the temporal and personal factors. To overcome the limitation of only being able to recommend hashtags that exist in the training data of the existing methods, the proposed method also incorporates extraction strategies into it. The results of experiments on the data collected from real world microblogging services by crawling demonstrate that the proposed method outperforms state-of-the-art methods that do not consider these aspects. The relative improvement of the proposed method over the method without considering these aspects is around 47.8% in F1-score.

1 Introduction

Over the past few years, social media services have become one of the most important communication channels for people. According to the statistic reported by the Pew Research Center's Internet & American Life Project in Aug 5, 2013, about 72% of adult internet users are also members of at least one social networking site. Hence, microblogs have also been widely used as data sources for public opinion analyses (Bermingham and Smeaton, 2010; Jiang et al., 2011), prediction (Asur and Huberman, 2010; Bollen et al., 2011), reputation management (Pang and Lee, 2008; Otsuka et al., 2012), and many other applications (Sakaki et al., 2010; Becker et al., 2010; Guy et al., 2010; Guy et al., 2013). In addition to the limited number of characters in the content, microblogs also contain a form of metadata tag (hashtag), which is a string of characters preceded by the symbol (#). Hashtags are used to mark the keywords or topics of a microblog. They can occur anywhere in a microblog, at the beginning, middle, or end. Hashtags have been proven to be useful for many applications, including microblog retrieval (Efron, 2010), query expansion (A.Bandyopadhyay et al., 2011), sentiment analysis (Davidov et al., 2010; Wang et al., 2011). However, only a few microblogs contain hashtags provided by their authors. Hence, the task of recommending hashtags for microblogs has become an important research topic and has received considerable attention in recent years.

Existing works have studied discriminative models (Ohkura et al., 2006; Heymann et al., 2008) and generative models (Krestel et al., 2009; Blei and Jordan, 2003; Ding et al., 2013) based on textual information from a single microblog. However, from a dataset containing 282.2 million microblogs crawled from Sina Weibo[1], we observe that different users may have different perspectives when picking hashtags, and the perspectives of users are impacted by their own interests or the global topic trend. Meanwhile,the global topic distribution is likely to change over time. To better understand how the topics vary over time, we aggregate the microblog posts published in a month as a document. Then, we use a Latent Dirichlet Allocation (LDA) to estimate their topics. Figure 1 illustrates an example, where ten active topics are selected. We can observe that the topics distribution varies greatly over time.

[1]http://www.weibo.com. It is one of the most popular microblog services in China.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 203–212, Dublin, Ireland, August 23-29 2014.

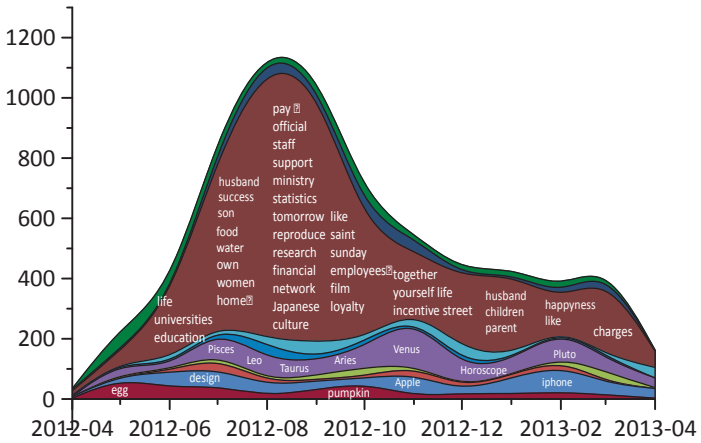


Figure 1: An example of the topics of retweets in each month. Each colored stripe represents a topic, whose height is the number of words assigned to the topic. For each topic, the top words of this topic in each month are placed on the stripe.

Motivated by the methods proposed to handle the vocabulary gap problem for keyphrase extraction (Liu et al., 2012) and hashtag suggestion (Ding et al., 2013), in this work, we also assume that the hashtags and textual content in a microblog are parallel descriptions of the same thing in different languages. To model the document themes, in this paper, we adopt the topical translation model to facilitate the translation process. Topic-specific word triggers are used to bridge the gap between the words and hashtags. Since existing topical translation models can only recommend hashtags learned from the training data, we also incorporate an extraction process into the model.

This work makes three main contributions. First, we incorporate temporal and personal factors into considerations. Most of the existing works on hashtag recommendation tasks have focused on textual information. Second, we adopt a topical translation model to combine extraction and translation methods. This makes it possible to suggest hashtags that are not included in the training data. Third, to evaluate the task, we construct a large collection of microblogs from a real microblogging service. All of the microblogs in the collection contain textual content and hashtags labeled by their authors. This can benefit other researchers investigating the same task or other topics using author-centered data.

The remaining part of this paper is structured as follows: We briefly review existing methods in related domains in Section 2. Section 3 gives an overview of the proposed generation model. Section 4 introduces the dataset construction, experimental results and analyses. In Section 5, we will conclude the paper.

2 Related Works

Due to the usefulness of tag recommendation, many methods have been proposed from different perspectives (Heymann et al., 2008; Krestel et al., 2009; Rendle et al., 2009; Liu et al., 2012; Ding et al., 2013). Heymann et al. (Heymann et al., 2008) investigated the tag recommendation problem using the data collected from social bookmarking system. They introduced an entropy-based metric to capture the generality of a particular tag. In (Song et al., 2008), a Poisson Mixture Model based method is introduced to achieve the tag recommendation task. Krestel et al. (Krestel et al., 2009) introduced a Latent Dirichlet Allocation to elicit a shared topical structure from the collaborative tagging effort of multiple users for recommending tags. Based on the the observation that similar webpages tend to have the same tags, Lu et al. proposed a method taking both tag information and page content into account to achieve the task (Lu et al., 2009). Ding et al. proposed to use translation process to model this task (Ding et al., 2013). They extended the translation based method and introduced a topic-specific translation model to process the various meanings of words in different topics. In (Tariq et al., 2013), discriminative-term-weights were used to establish topic-term relationships, of which users' perception were learned to suggest suitable hashtags for users. To handle the vocabulary problem in keyphrase extraction task, Liu et al. proposed a

topical word trigger model, which treated the keyphrase extraction problem as a translation process with latent topics (Liu et al., 2012).

Most of the works mentioned above are based on textual information. Besides these methods, personalized methods for different recommendation tasks have also been paid lots of attentions (Liang et al., 2007; Shepitsen et al., 2008; Garg and Weber, 2008; Li et al., 2010; Liang et al., 2010; Rendle and Schmidt-Thieme, 2010). Shepitsen et al. (2008) proposed to use hierarchical agglomerative clustering to take into account personalized navigation context in cluster selection. In (Garg and Weber, 2008), the problem of personalized, interactive tag recommendation was also studied based on the statics of the tags co-occurrence. Liang et al. (2010) proposed to the multiple relationships among users, items and tags to find the semantic meaning of each tag for each user individually and used this information for personalized item recommendation.

From the brief descriptions given above, we can observe that most of the previous works on hashtag suggestion focused on textual information. In this work, we propose to incorporate temporal and personal information into the generative methods. Further more, to over the limitation that translation based method can only recommend hashtags learned from the training data, we also propose to incorporate an extraction process into the model.

3 The Proposed Methods

In this section, we firstly introduce the notation and generation process of the proposed method. Then, we describe the method used for learning parameters. Finally, we present the methods of how do we apply the learned model to achieve the hashtag recommendation task.

3.1 The Generation Process

We use D to represent the number of microblogs in the given corpus, and the microblogs have been divided into T epoches. Let $t = 1, 2, ..., T$ be the index of an epoches, θ^t is the topic distribution of the epoch t. Each microblog is generated by a user u_i, where u_i is an index between 1 and U, and U is the total number of users. A microblog is a sequence of N_d words denoted by $w_d = \{w_{d1}, w_{d2}, ..., w_{dN_d}\}$. Each microblog contains a set of hashtags denoted by $h_d = \{h_{d1}, h_{d2}, ..., h_{dM_d}\}$. A word is defined as an item from a vocabulary with W distinct words indexed by $w = \{w_1, w_2, ..., w_W\}$. Each hashtag is from the vocabulary with V distinct hashtags indexed by $h = \{h_1, h_2, ..., h_V\}$. The notations in this paper are summarized in Table 1.

The original LDA assumes that a document is contains a mixture of topics, which is represented by a topic distribution, and each word has a hidden topic label. Although, it is sensible for long document, due to the limitations of the length of characters in a single microblog, it tends to be about a single topic. Hence, we associate a single hidden variable with each microblog to indicate its topic. Similar idea of assigning a single topic to a short sequence of words has also been used for modeling Twitters (Zhao et al., 2011)

The hashtag recommendation task is to discover a list of hashtags for each unlabeled microblog, In our method, we first learn a topical translation model, and then we estimate the latent variables for each microblog, finaly recommending hashtags accord to the learned model.

Fig. 2 shows the graphical representation of the generation process. The generative story for each microblog is as follows:

3.2 Learning

To learn the parameters of our model, we use collapsed Gibbs sampling (Griffiths and Steyvers, 2004) to sample the topics assignment $\mathbf{z}$, latent variables assignment $\mathbf{x}$ and $\mathbf{y}$.

Given the current state of all but the variable x_d and z_d for the dth microblog, we can jointly sample

1. Draw $\pi \sim Beta(\delta)$, $\eta \sim Beta(\lambda)$
2. Draw background word distribution $\phi^B \sim Dirichlet(\beta^w)$
3. Draw global trendy topic distribution $\theta^t \sim Dirichlet(\alpha)$ for each time epoch $t = 1, 2, ..., T$
4. Draw personal topic distribution $\psi^u \sim Dirichlet(\alpha)$ for each user $u = 1, 2, ..., U$
5. Draw word distribution $\phi^z \sim Dirichlet(\beta^w)$ for each topic $z = 1, 2, ..., K$
6. Draw hashtag distribution $\varphi_{z,w} \sim Dirichilet(\beta^h)$ for each topic $z = 1, 2, ..., K$ and each word $w = 1, 2, ..., W$
7. For each microblog $d = 1, 2, ..., D$

   a. Draw $x_d \sim Bernoulli(\eta)$

   b. If $x_d = 0$ then

   Draw a topic $z_d \sim Multinomial(\psi^u)$

   End if

   If $x_d = 1$ then

   Draw a topic $z_d \sim Multinomial(\theta^t)$

   End if

   c. For each word $n = 1, ..., N_d$

   i. Draw $y_{dn} \sim Bernoulli(\pi)$

   ii. If $y_{dn} = 0$ then

   Draw a word $w_{dn} \sim Multinomial(\phi^B)$

   End if

   If $y_{dn} = 1$ then

   Draw a word $w_{dn} \sim Multinomial(\phi^{z_d})$

   End if

   d. For each hashtag $m = 1, ..., M_d$

   i. Draw $h_{dm} \sim P(h_{dm}|w_d, z_d, \varphi_{z_d,w_d})$

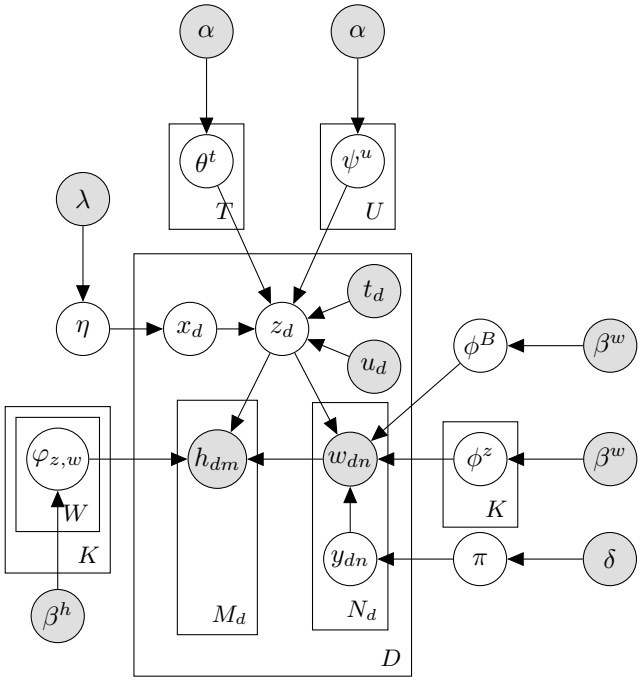


Figure 2: The graphical representation of the proposed model. Shaded circles are observations or constants. Unshaded ones are hidden variables.

Table 1: The notations used in this work.

D	The number of training data set
W	The number of unique word in the corpus
V	The number of unique hashtag in the corpus
K	The number of topics
T	The total number of time epoches
U	The total number of users
N_d	The number of words in the dth microblog
M_d	The number of hashtags in the dth microblog
z_d	The topic of the dth microblog
x_d	The latent variable decided the distribution category of z_d
y_{dn}	The latent variable decided the distribution category of w_{dn}
π	The distribution of latent variable y_{dn}
η	The distribution of latent variable x_d
ϕ^z	The distribution of topic words
ϕ^B	The distribution of background words
θ^t	The distribution of topics for time epoch t
ψ^u	The distribution of topics for user u
t_d	The time epoch for microblog d
u_d	The user of the microblog d
φ	The topic-specific word alignment table between word and hashtag or itself

x_d and z_d, the conditional probability of $x_d = p$,$z_d = k$ is calculated as follows:

$$\begin{aligned} &Pr(x_d = p, z_d = k|\mathbf{z}_{\neg\mathbf{d}}, \mathbf{x}_{\neg\mathbf{d}}, \mathbf{y}, \mathbf{w}, \mathbf{h}) \\ &\propto \frac{N_p^\eta + \lambda}{N_{(.)}^\eta + 2\lambda} \cdot \frac{N_k^l + \alpha}{N_{(.)}^l + K\alpha} \cdot \prod_{n=1}^{N_d} \frac{N_{w_{dn}}^k + \beta^w}{N_{(.)}^k + W\beta^w} \cdot \prod_{m=1}^{M_d} \sum_{n=1}^{N_d} \frac{M_{\neg d,k}^{w_{dn},h_{dm}} + \beta^h}{M_{\neg d,k}^{w_{dn},(.)} + V\beta^h}, \end{aligned} \tag{1}$$

where $l = u_d$ when $p = 0$ and $l = t_d$ when $p = 1$. N_0^η is the number of microblog generated by personal interests, while N_1^η is the number of microblog coming from global topical trends, $N_{(.)}^\eta = N_0^\eta + N_1^\eta$. $N_k^{u_d}$ is the number of microblogs generated by user u_d and under topic k. $N_{(.)}^{u_d}$ is the total number of microblogs generated by user u_d. $N_k^{t_d} = \sum_{t'=1}^{t_d} e^{\frac{-t'}{\rho}} N'^{t-t'}_k$, $N'^{t-t'}_k$ is the number of microblogs assigned to topic k at time epoch $t - t'$, $e^{\frac{-t'}{\rho}}$ is decay factory, and $N_{(.)}^{t_d} = \sum_{k=1}^K N_k^{t_d}$. $N_{w_{dn}}^k$ is the times of word w_{dn} assigned to topic k, $N_{(.)}^k$ is the times of all the word assigned to topic k, $M_{\neg d,k}^{w_{dn},h_{dm}}$ is the number of occurrences that word w_{dn} is translated to hashtag h_{dm} given topic k. All the counters mentioned above are calculated with the dth microblog excluded.

We sample y_{dn} for each word w_{dn} in the dth microblog using the following equation:

$$Pr(y_{dn} = q|\mathbf{z}, \mathbf{x}, \mathbf{y}_{\neg\mathbf{dn}}, \mathbf{w}, \mathbf{h}) \propto \frac{N_q^\pi + \delta}{N_{(.)}^\pi + 2\delta} \cdot \frac{N_{w_{dn}}^l + \beta^w}{N_{(.)}^l + W\beta^w}, \tag{2}$$

where $l = B$ when $q = 0$ and $l = z_d$ when $q = 1$. N_0^π is the number of words assigned to background words and N_1^π is the number of words under any topic respectively. $N_{(.)}^\pi = N_0^\pi + N_1^\pi$, $N_{w_{dn}}^B$ is a count of word w_{dn} occurs as a background word. $N_{w_{dn}}^{z_d}$ is the number of word w_{dn} is assigned to topic z_d, and $N_{(.)}^{z_d}$ is the total number of words assigned to topic z_d. All counters are calculated with taking no account of the current word w_{dn}.

In many cases, hashtag dose not appear in the training data, to solve this problem, we assume that each word in the microblog can translate to a hashtag in the training data or itself. We assume that each word

have aligned σ (we set $\sigma = 1$ in this paper after trying some number) times with itself under the specific topic. After all the hidden variables become stable, we can estimate the alignment probability as follows:

$$\varphi_{h,w,z} = \begin{cases} \frac{N^h_{z,w}+\beta^h}{N^{(.)}_{z,w}+\sigma+(V+1)\beta^h} & \text{if } h \text{ is a hashtag in the training data} \\ \frac{\sigma+\beta^h}{N^{(.)}_{z,w}+\sigma+(V+1)\beta^h} & \text{if } h \text{ is the word itself} \end{cases} \tag{3}$$

where $N^h_{z,w}$ is the number of the hashtag h co-occurs with the word w under topic z in the microblogs.

For the probability alignment φ between hashtag and word, the potential size is $W \cdot V \cdot K$. The data sparsity poses a more serious problem in estimating φ than the topic-free word alignment case. To remedy the problem, we use interpolation smoothing technique for φ. In this paper, we emplogy smoothing as follows:

$$\varphi^*_{h,w,z} = \gamma\varphi_{h,w,z} + (1-\gamma)P(h|w), \tag{4}$$

where $\varphi^*_{h,w,z}$ is the smoothed topical alignment probabilities, $\varphi_{h,w,z}$ is the original topical alignment probabilities. $P(h|w)$ is topic-free word alignment probability. Here we obtain $P(h|w)$ by exploring IBM model-1 (Brown et al., 1993). γ is trade-off of two probabilities ranging from 0.0 to 1.0. When $\gamma = 0.0$, $\varphi^*_{h,w,z}$ will be reduce to topic-free word alignment probability; and when $\gamma = 1.0$, there will be no smoothing in $\varphi^*_{h,w,z}$. For the word itself there are no smoothing, because it is a pseudo-count.

3.3 Hashtag Extraction

We perform hashtag extraction as follows. Suppose given an unlabeled dataset, we perform Gibbs Sampling to iteratively estimate the topic and determine topic/background words for each microblog. The process is the same as described in Section 3.2. After the hidden variables of topic/background words and the topic of each microblog become stable, we can estimate the distribution of topics for the dth microblog in unlabeled data by:$\chi^*_{dk} = \frac{p(k)p(w_{d1}|k)\ldots p(w_{dN_d}|k)}{Z}$ where $p(w_{dn}|k) = \frac{N^\pi_1+\delta}{N^\pi_{(.)}+2\delta} \cdot \frac{N^k_{w_{dn}}+\beta^w}{N^k_{(.)}+W\beta^w}$ and $N^k_{w_{dn}}$ is the number of words w_{dn} that are assigned to topic k in the corpus, and $p(k) = \frac{N^\eta_0+\lambda}{N^\eta_{(.)}+2\lambda} \cdot \frac{N^u_k+\alpha}{N^u_{(.)}+K\alpha} + \frac{N^\eta_1+\lambda}{N^\eta_{(.)}+2\lambda} \cdot \frac{N^t_k+\alpha}{N^t_{(.)}+K\alpha}$ is regarded as a prior for topic distribution, Z is the normalized factor. With topic distribution χ^* and topical alignment table φ^*, we can rank hashtags for the dth microblog in unlabeled data by computing the scores:

$$P(h_{dm}|w_d, \chi^*_d, \varphi^*) \propto \sum_{z_d=1}^{K}\sum_{n=1}^{N_d} P(h_{dm}|z_d, w_{dn}, \varphi^*) \cdot P(z_d|\chi^*_d) \cdot P(w_{dn}|w_d), \tag{5}$$

where h_{dm} can be a hashtag in the training data or a word in the dth microblog, $p(w_{dn}|w_d)$ is the weight of the word w_{dn} in the microblog, which can be estimated by the IDF score of the word. According to the ranking scores, we can suggest the top-ranked hashtags for each microblog to users.

4 Experiments

In this section, we introduce the experimental results and the data collection we constructed for training and evaluation. Firstly, we describe how do we construct the collection and statics of it. Then we introduce the experiment configurations and baseline methods. Finally, the evaluation results and analysis are given.

4.1 Data Collection

We use a dataset collected from Sina Weibo to evaluate the proposed approach and alternative methods. We random select 166,864 microblogs from Aug. 2012 to June 2013. The unique number of hashtags in the corpus is 17,516. We use the microblogs posted from Aug. 2012 to May 2013 as the training data. The other microblogs are used for evaluation. The hashtags marked in the original microblogs are considered as the golden standards.

Figure 3: Precision-recall curves of different methods on this task.

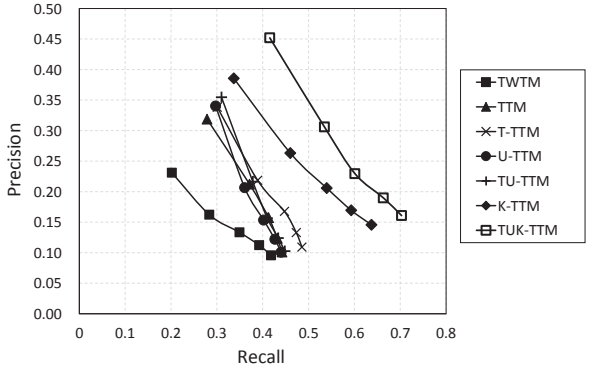


Table 2: Evaluation results of different methods on the evaluation collection.

Methods	Precision	Recall	F_1
TWTM	0.231	0.202	0.215
SVM	0.418	0.366	0.390
TTM	0.319	0.279	0.297
T-TTM	0.338	0.301	0.319
U-TTM	0.341	0.307	0.323
K-TTM	0.386	0.337	0.360
TU-TTM	0.355	0.310	0.331
TUK-TTM	**0.452**	**0.415**	**0.433**

4.2 Experiment Configurations

We use precision (P), recall (R), and F1-score (F_1) to evaluate the performance. Precision is calculated based on the percentage of "hashtags truly assigned" among "hashtags assigned by system". Recall is calculated based on the "hashtags truly assigned" among "hashtags manually assigned". F1-score is the harmonic mean of precision and recall. We do 500 iterations of Gibbs sampling to train the model. For optimize the hyperparmeters of the proposed method and alternative methods, we use 5-fold cross-validation in the training data to do it. The number of topics is set to 70. The other settings of hyperparameters are as follows: $\alpha = 50/K$, $\beta^w = 0.1$, $\beta^h = 0.1$, $\lambda = 0.01$, and $\delta = 0.01$. The smoothing factor γ in Eq.(3) is set to 0.6. For estimating the translation probability without topical information, we use GIZA++ 1.07 to do it (Och and Ney, 2003).

For baselines, we compare the proposed model with the following alternative models.

- **TWTM**: Topical word trigger model (TWTM) was proposed by Liu et al. for keyphrase extraction using only textual information (Liu et al., 2012). We implemented the model and used it to achieve the task.

- **TTM**: Ding et al. (2013) proposed the topical translation model (TTM) for hash tag extraction. We implemented and extended their method for evaluating it on the corpus constructed in this work.

4.3 Experimental Results

Table 2 shows the comparisons of the proposed method with the state-of-the-art methods on the constructed evaluation dataset. "TUK-TTM" denotes the method proposed in this paper. "T-TTM" and "U-TTM" represent the methods incorporating temporal and personal information respectively. "K-TTM" represents the method incorporating the extraction factor. From the results, we can observe that the proposed method is significantly better than other methods at 5% significance level (two-sided). Comparing to results of the TTM, we can observe that the temporal information, personal information and extraction strategy can all benefit the task. Among the three additional factors, the extraction strategy achieves the best result. The limitation of only being able to recommend hashtags that exist in the training data can be overcome in some degree by the proposed method. The relative improvement of proposed TUK-TTM over TTM is around 47.8% in F1-score.

Table 3 shows the comparisons of the proposed method with the method "K-TTM" in two corpus NE-Corpus and E-Corpus. NE-Corpus include microblogs whose hashtags are not contained in the training data. E-Corpus include the microblogs whose hashtags appear in the training data. We can observe that the proposed method significantly better than "K-TTM" in the E-Corpus. Another observation is that the method incorporating the extraction factor achieves better performances on the NE-Corpus than E-Corpus. We think that the reason is that the fewer times hashtag appear, the greater weight it has. Hence, we can extract this kind of hashtags more easier.

Figure 3 shows the precision-recall curves of TWTW, TTM, T-TTM, U-TTM, TU-TTM, K-TTM, and TUK-TTM on the evaluation dataset. Each point of a precision-recall curve represents extracting

Table 3: Evaluation results of two different corpus.

Corpus	Methods	P	R	F
NE-Corpus	K-TTM	0.631	0.553	0.589
	TUK-TTM	0.641	0.561	0.598
E-Corpus	K-TTM	0.172	0.162	0.167
	TUK-TTM	0.288	0.271	0.279

Table 4: The influence of the number of topics K of TUK-TTM.

K	Precision	Recall	$\mathbf{F}_1$
10	0.410	0.382	0.396
30	0.435	0.380	0.406
50	0.448	0.413	0.430
70	**0.452**	**0.415**	**0.433**
100	0.439	0.404	0.421

Table 5: The influence of the smoothing parameter γ of TUK-TTM.

γ	Precision	Recall	$\mathbf{F}_1$
0.0	0.379	0.354	0.366
0.2	0.405	0.372	0.388
0.4	0.433	0.398	0.415
0.6	**0.452**	**0.415**	**0.433**
0.8	0.426	0.386	0.405
1.0	0.423	0.381	0.401

different number of hashtags ranging from 1 to 5 respectively. In the figure, curves which are close to the upper right-hand corner of the graph indicate the better performance. From the results, we can observe that the performance of TUK-TTM is in the upper right-hand corner. It also demonstrates that the proposed method achieves better performances than other methods.

From the description of the proposed model, we can know that there are several hyperparameters in the proposed TUK-TTM. To evaluate the impacts of them, we evaluate two crucial ones, the number of topics K and the smoothing factor γ. Table 4 shows the influence of the number of topics. From the table, we can observe that the proposed model obtains the best performance when K is set to 70. And performance decreases with more number of topics. We think that data sparsity may be one of the main reasons. With much more topic number, the data sparsity problem will be more serious when estimating topic-specific translation probability. Table 5 shows the influence of the translation probability smoothing parameter γ. When γ is set to 0.0, it means that the topical information is omitted. Comparing the results of $\gamma = 0.0$ and other values, we can observe that the topical information can benefit this task. When γ is set to 1.0, it represents the method without smoothing. The results indicate that it is necessary to address the sparsity problem through smoothing.

5 Conclusions

In this paper, we propose a novel method which incorporates temporal and personal factors into the topical translation model for hashtag recommendation task. Since existing translation model based methods for this task can only recommend hashtags that exist in the training data of the topical translation model, we also incorporate extraction strategies into the model. To evaluate the proposed method, we also construct a dataset from real world microblogging services. The results of experiments on the dataset demonstrate that the proposed method outperforms state-of-the-art methods that do not consider these aspects.

6 Acknowledgement

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by 973 Program (2010CB327900), National Natural Science Foundation of China (61003092,61073069), Shanghai Leading Academic Discipline Project (B114) and "Chen Guang" project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation(11CG05).

References

A.Bandyopadhyay, M. Mitra, and P. Majumder. 2011. Query expansion for microblog retrieval. In *Proceedings of The Twentieth Text REtrieval Conference*, TREC 2011.

S. Asur and B.A. Huberman. 2010. Predicting the future with social media. In *WI-IAT'10*, volume 1, pages 492–499.

Hila Becker, Mor Naaman, and Luis Gravano. 2010. Learning similarity metrics for event identification in social media. In *Proceedings of WSDM '10*.

Adam Bermingham and Alan F. Smeaton. 2010. Classifying sentiment in microblogs: is brevity an advantage? In *Proceedings of CIKM'10*.

D.M. Blei and M.I. Jordan. 2003. Modeling annotated data. In *Proceedings of SIGIR*, pages 127–134.

Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8.

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

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of COLING '10*.

Zhuoye Ding, Xipeng Qiu, Qi Zhang, and Xuanjing Huang. 2013. Learning topical translation model for microblog hashtag suggestion. In *Proceedings of IJCAI 2013*.

Miles Efron. 2010. Hashtag retrieval in a microblogging environment. In *Proceedings of SIGIR '10*.

Nikhil Garg and Ingmar Weber. 2008. Personalized, interactive tag recommendation for flickr. In *Proceedings of RecSys '08*.

T. L. Griffiths and M. Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*.

Ido Guy, Naama Zwerdling, Inbal Ronen, David Carmel, and Erel Uziel. 2010. Social media recommendation based on people and tags. In *Proceedings of SIGIR '10*.

Ido Guy, Uri Avraham, David Carmel, Sigalit Ur, Michal Jacovi, and Inbal Ronen. 2013. Mining expertise and interests from social media. In *Proceedings of WWW '13*.

Paul Heymann, Daniel Ramage, and Hector Garcia-Molina. 2008. Social tag prediction. In *Proceedings of SIGIR '08*.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of ACL 2011*, Portland, Oregon, USA.

Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl. 2009. Latent dirichlet allocation for tag recommendation. In *Proceedings of RecSys '09*.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM.

Ting-Peng Liang, Hung-Jen Lai, and Yi-Cheng Ku. 2007. Personalized content recommendation and user satisfaction: Theoretical synthesis and empirical findings. *Journal of Management Information Systems*, 23(3):45–70.

Huizhi Liang, Yue Xu, Yuefeng Li, Richi Nayak, and Xiaohui Tao. 2010. Connecting users and items with weighted tags for personalized item recommendations. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, pages 51–60. ACM.

Zhiyuan Liu, Chen Liang, and Maosong Sun. 2012. Topical word trigger model for keyphrase extraction. In *Proceedings of COLING*.

Yu-Ta Lu, Shoou-I Yu, Tsung-Chieh Chang, and Jane Yung-jen Hsu. 2009. A content-based method to enhance tag recommendation. In *Proceedings of IJCAI'09*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Tsutomu Ohkura, Yoji Kiyota, and Hiroshi Nakagawa. 2006. Browsing system for weblog articles based on automated folksonomy. *Workshop on the Weblogging Ecosystem Aggregation Analysis and Dynamics at WWW*.

Takanobu Otsuka, Takuya Yoshimura, and Takayuki Ito. 2012. Evaluation of the reputation network using realistic distance between facebook data. In *Proceedings of WI-IAT '12*.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January.

Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 81–90. ACM.

Steffen Rendle, Leandro Balby Marinho, Alexandros Nanopoulos, and Lars Schmidt-Thieme. 2009. Learning optimal ranking with tensor factorization for tag recommendation. In *Proceedings of KDD '09*.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of WWW '10*.

Andriy Shepitsen, Jonathan Gemmell, Bamshad Mobasher, and Robin Burke. 2008. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 259–266, New York, NY, USA. ACM.

Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C. Lee Giles. 2008. Real-time automatic tag recommendation. In *Proceedings of SIGIR '08*.

Amara Tariq, Asim Karim, Fernando Gomez, and Hassan Foroosh. 2013. Exploiting topical perceptions over multi-lingual text for hashtag suggestion on twitter. In *FLAIRS Conference*.

Xiaolong Wang, Furu Wei, Xiaohua Liu, Ming Zhou, and Ming Zhang. 2011. Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach. In *Proceedings of CIKM '11*.

Wayne Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li. 2011. Topical keyphrase extraction from twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 379–388. Association for Computational Linguistics.

Sarcasm Detection on Czech and English Twitter

Tomáš Ptáček[†‡], **Ivan Habernal**[†] **and Jun Hong**[‡]
[†] Department of Computer Science and Engineering, Faculty of Applied Sciences,
University of West Bohemia, Univerzitní 8, 306 14 Plzeň, Czech Republic
tigi@kiv.zcu.cz habernal@kiv.zcu.cz

[‡] School of Electronics, Electrical Engineering and Computer Science,
Queen's University Belfast, Belfast BT7 1NN, UK
j.hong@qub.ac.uk

Abstract

This paper presents a machine learning approach to sarcasm detection on Twitter in two languages – English and Czech. Although there has been some research in sarcasm detection in languages other than English (e.g., Dutch, Italian, and Brazilian Portuguese), our work is the first attempt at sarcasm detection in the Czech language. We created a large Czech Twitter corpus consisting of 7,000 manually-labeled tweets and provide it to the community. We evaluate two classifiers with various combinations of features on both the Czech and English datasets. Furthermore, we tackle the issues of rich Czech morphology by examining different preprocessing techniques. Experiments show that our language-independent approach significantly outperforms adapted state-of-the-art methods in English (F-measure 0.947) and also represents a strong baseline for further research in Czech (F-measure 0.582).

1 Introduction

Sentiment analysis on social media has been one of the most targeted research topics in NLP in the past decade, as shown in several recent surveys (Liu and Zhang, 2012; Tsytsarau and Palpanas, 2012). Since the goal of sentiment analysis is to automatically detect the polarity of a document, misinterpreting irony and sarcasm represents a big challenge (Davidov et al., 2010).

As there is only a weak boundary in meaning between irony, sarcasm and satire (Reyes et al., 2012), we will use only the term sarcasm in this paper. Bosco et al. (2013) claim that "even if there is no agreement on a formal definition of irony, psychological experiments have delivered evidence that humans can reliably identify ironic text utterances from an early age in life." We have thus decided to rely on the ability of our human annotators to manually label sarcastic tweets to train our classifiers. Sarcasm generally reverses the polarity of an utterance from positive or negative into its opposite, which deteriorates the results of a given NLP task. Therefore, correct identification of sarcasm can improve the performance.

The issue of automatic sarcasm detection has been addressed mostly in English, although there has been some research in other languages, such as Dutch (Liebrecht et al., 2013), Italian (Bosco et al., 2013), or Brazilian Portuguese (Vanin et al., 2013). To the best of our knowledge, no research has been conducted in Czech or other Slavic languages. These languages are challenging for many NLP tasks because of their rich morphology and syntax. This has motivated us to focus our current research on both English and Czech.

Majority of the existing state-of-the-art techniques are language dependent, which rely on language-specific lexical resources. Since no such resources are available for Czech, we adapt some language-independent methods and also apply various preprocessing steps for sentiment analysis proposed by Habernal et al. (2013).

This paper focuses on document-level sarcasm detection on Czech and English Twitter datasets using supervised machine learning methods. The Czech dataset consists of 7,000 manually labeled tweets,

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 213–223, Dublin, Ireland, August 23-29 2014.

the English dataset consists of a balanced distribution and an imbalanced distribution, each containing 100,000 tweets, where hashtag #sarcasm was used as an indicator of sarcastic tweets. We provide both datasets under Creative Commons BY-NC-SA licence[1] at http://liks.fav.zcu.cz/sarcasm/.

Our research questions were the following: (1) To what extent can the language-independent approach compete with methods based on lexical language-dependent resources? (2) Is it possible to reach good agreement on annotating sarcasm and what typical text properties on Twitter are important for sarcasm detection? (3) What is the best preprocessing pipeline that can boost performance on highly-flective Czech language and what types of features and classifiers yield the best results?

The rest of this article is organized as follows. Section 2 describes the related work. In section 3, we outline our approach to sarcasm detection and describe the selection of features in our approach. Section 4 thoroughly describes the datasets and the annotation process. Section 5 describes and discusses the experimental results. Finally we conclude in Section 6.

2 Related Work

Experiments with semi-supervised sarcasm identification on a Twitter dataset (5.9 million tweets) and on 66,000 product reviews from Amazon were conducted in (Davidov et al., 2010) and (Tsur et al., 2010). They used 5-fold cross validation on their kNN-like classifier and obtained an F-measure of 0.83 on the product reviews dataset and 0.55 on the Twitter dataset. For acquiring the Twitter dataset they used hashtag #sarcasm as an indicator of sarcastic tweets. They further created a balanced evaluation set of 180 tweets using 15 human annotators via Amazon Mechanical Turk[2] and achieved an inter-annotator agreement 0.41 (Fleiss' κ).

González-Ibáñez et al. (2011) experimented with Twitter data divided into three categories (sarcastic, positive sentiment and negative sentiment), each containing 900 tweets. They used the #sarcasm and #sarcastic hashtags to identify sarcastic tweets. They used two classifiers – support vector machine (SVM) with sequential minimal optimization (SMO) and logistic regression. They tried various combinations of unigrams, dictionary-based features and pragmatic factors (positive and negative emoticons and user references), achieving the best result (accuracy 0.65) for sarcastic and non-sarcastic classification with the combination of SVM with SMO and unigrams. They employed 3 human judges to annotate 180 tweets (90 sarcastic and 90 non-sarcastic). The human judges achieved Fleiss' $\kappa = 0.586$, demonstrating the difficulty of sarcasm classification. Another experiment included 50 sarcastic and 50 non-sarcastic (25 positive, 25 negative) tweets with emoticons annotated by two judges. The automatic classification and human judges achieved the accuracy of 0.71 and 0.89 respectively. The inter-annotator agreement (Cohen's κ) was 0.74.

Reyes et al. (2012) proposed features to capture properties of a figurative language such as ambiguity, polarity, unexpectedness and emotional scenarios. Their corpus consists of five categories (humor, irony, politics, technology and general), each containing 10,000 tweets. The best result in the classification of irony and general tweets was F-measure 0.65.

In (Reyes et al., 2013) they explored the representativeness and relevance of conceptual features (signatures, unexpectedness, style and emotional scenarios). These features include punctuation marks, emoticons, quotes, capitalized words, lexicon-based features, character n-grams, skip-grams, (Guthrie et al., 2006), and polarity skip-grams. Their corpus consists of four categories (irony, humor, education and politics), each containing 10,000 tweets. Their evaluation was performed on two distributional scenarios, balanced distribution and imbalanced distribution (25% ironic tweets and 75% tweets from all three non-ironic categories) using the Naive Bayes and decision trees algorithms from the Weka toolkit (Witten and Frank, 2005). The classification by the decision trees achieved an F-measure of 0.72 on the balanced distribution and an F-measure of 0.53 on the imbalanced distribution.

The work of Riloff et al. (2013) identifies one type of sarcasm: contrast between a positive sentiment and negative situation. They used a bootstrapping algorithm to acquire lists of positive sentiment phrases

[1] http://creativecommons.org/licenses/by-nc-sa/3.0/

[2] http://www.mturk.com

and negative situation phrases from sarcastic tweets. They proposed a method which classifies tweets as sarcastic if it contains a positive predicative that precedes a negative situation phrase in close proximity. Their evaluation on a human-annotated dataset[3] of 3000 tweets (23% sarcastic) was done using the SVM classifier with unigrams and bigrams as features, achieving an F-measure of 0.48. The hybrid approach that combines the results of the SVM classifier and their contrast method achieved an F-measure of 0.51.

Sarcasm and nastiness classification in online dialogues was also explored in (Lukin and Walker, 2013) using bootstrapping, syntactic patterns and a high precision classifier. They achieved an F-measure of 0.57 on their sarcasm dataset.

3 Our Approach

This paper presents the first attempt at sarcasm detection in the Czech language, in which we focus on supervised machine learning approaches and evaluate their performance. We selected various n-grams, including unigrams, bigrams, trigrams with frequency greater than three (Liebrecht et al., 2013), and a set of language-independent features, including punctuation marks, emoticons, quotes, capitalized words, character n-grams and skip-grams (Reyes et al., 2013) as our baselines.

3.1 Classification

Our evaluation was performed using the Maximum Entropy (MaxEnt) and Support Vector Machine (SVM) classifiers. We used *Brainy* – a Java framework for machine learning (Konkol, 2014) – with default settings (the linear kernel for SVM). All experiments were conducted in the 5-fold cross validation manner similar to (Davidov et al., 2010; González-Ibáñez et al., 2011). Our motivation to test multiple classifiers stemmed also from related works which mostly test more than one classifier. On the other hand, the choice between state-of-the-art linear classifiers might not be much of importance, as the most important is the feature engineering.

3.2 Features

For our evaluation we used the most promising language-independent features from the related work and POS related features. Feature sets used in our evaluation are described in Table 1.

Group	Features	Description
N-gram	Character n-gram	We used character n-gram features (Blamey et al., 2012). We set the minimum occurrence of a particular character n-gram to either 5 or 50, in order to prune the feature space. Our character feature set contains 3-grams to 6-grams.
	N-gram	We used word unigrams, bigrams and trigrams as binary features. The feature space is pruned by the minimum n-gram occurrence set to 3 (Liebrecht et al., 2013).
	Skip-bigram	Instead of using sequences of adjacent words (n-grams) we used skip-grams (Guthrie et al., 2006), which skip over arbitrary gaps. Reyes et al. (2013) consider skip-bigrams with 2 or 3 word skips and remove skip-grams with a frequency ≤ 20.
Pattern	Pattern	Patterns composed of high frequency words (HFWs)[4] and content words (CWs)[5] used by (Davidov et al., 2010). Pattern must contain at least one high frequency word. The patterns contain 2-6 HFWs and 1-6 CWs. We set the minimum occurrence of a particular pattern to 5.
	Word-shape pattern	We tried to improve pattern features by using word-shape classes for content words. We assign words into one of 24 classes[6] similar to the function specified in (Bikel et al., 1997).
POS	POS characteristics	We implemented various POS features, e.g., the number of nouns, verbs, and adjectives (Ahkter and Soria, 2010), the ratio of nouns to adjectives and verbs to adverbs (Kouloumpis et al., 2011), and number of negative verbs obtained from POS tags.

[3]They used three annotators. Each annotator was given the same 100 tweets with the sarcasm hashtag and 100 tweets without the sarcasm hashtag (the hashtags were removed). On these tweets the pairwise inter-annotator scores were computed (Cohen's Kappa $\kappa_1 = 0.80, \kappa_2 = 0.81$ and $\kappa_3 = 0.82$). Then each annotator labeled additional 1000 tweets.

[4]A word whose corpus frequency is more than 1000 words per million plus all punctuation characters.

[5]A word whose corpus frequency is less than 1000 words per million.

[6]We use edu.stanford.nlp.process.WordShapeClassifier with the WORDSHAPECHRIS1 setting.

	POS word-shape	Unigram feature consisting of POS and word-shape (see Word-shape pattern). The feature space is pruned by the minimum occurrence set to 5.
	POS n-gram	Direct use of POS n-grams has not shown any significant improvement in sentiment analysis but it may improve the results of sarcasm detection. We experimented with 3-grams to 6-grams with the minimum n-gram occurrence set to 5.
Others	Emoticons	We used two lists of positive and negative emoticons (Montejo-Ráez et al., 2012). The feature captures the number of occurrences of each class of emoticons within the text.
	Punctuation-based	We adapted punctuation-based features proposed by (Davidov et al., 2010). This feature set consists of number of words, exclamation marks, question marks, quotation marks and capitalized words normalized by dividing them by the maximal observed value multiplied by the averaged maximal value of the other feature groups.
	Pointedness	Pointedness was used by (Reyes et al., 2013) to distinguish irony. It focuses on explicit marks which should reflect a sharp distinction in the information that is transmitted. The presence of punctuation marks, emoticons, quotes and capitalized words has been considered.
	Extended Pointedness	This feature captures the number of occurrences of punctuation marks and emoticons as well as the number of words, exclamation marks, question marks, quotation marks and capitalized words normalized by maximal observed value.
	Word-case	We implemented various word-case features that include, e.g., the number of upper cased words, number of words with first letter capital normalized by number of words and number of upper cased characters normalized by number of words.

Table 1: Descriptions of used feature sets.

4 Evaluation Datasets

We collected datasets using *Twitter Search API* and *Java Language Detector*[7]. We collected 140,000 Czech and 780,000 English tweets, respectively. Due to lack of support for the Czech language on Twitter, we used the *Twitter Search API* parameter *geocode* to acquire tweets posted near Prague. For the English dataset we also collected tweets with the #sarcasm hashtag. Czech users generally don't use the sarcasm ("`#sarkasmus`") or irony ("`#ironie`") hashtag variants[8] thus we had to annotate the Czech dataset manually. The final label distribution in datasets is shown in Table 4.

4.1 Filtering and Normalization

All user, URL and hashtag references in tweets have been replaced by "user", "link" and "hashtag" respectively. We also removed all tweets starting with "RT" because they refer to previous tweets and tweets containing just combinations of user, link, "RT" and hashtags without any additional words.

Tokenization of tweets requires proper handling of emoticons and other special character sequences typical on Twitter. The *Ark-tweet-nlp tool* (Gimpel et al., 2011) offers precisely that and although it was developed and tested in English, it yields satisfactory results in Czech as well.

Czech is a highly flective language and uses a lot of diacritics. However some Czech users type only the unaccented characters.[9] Preliminary experiments showed that removing diacritics yields better results, thus we removed diacritics from all tweets.

4.2 Czech Dataset Annotation

Firstly we conducted an experiment to determine whether to annotate the original data or the normalized data. We selected two sample sets of 50 tweets containing Czech sarcasm (`#sarkasmus`) and irony (`#ironie`) hashtags and other tweets. One annotator obtained the original data while the other got the normalized data from the first sample set. We then tried to give both annotators the original data from the first sample set and finally we gave them both the normalized data from the second sample set. Table 2 shows the difficulty of sarcasm identification without the knowledge hidden in hashtags, user and links.

[7] `http://code.google.com/p/jlangdetect/`

[8] We found only 10 tweets with sarcasm hashtag ("`#sarkasmus`") and 100 tweets with irony hashtag ("`#ironie`") in 140,000 collected tweets.

[9] Approximately 10% of collected tweets were without any diacritics.

Normalized	Normalized		
	Tag	**n**	**s**
	n	35	10
	s	0	5
Cohen's κ:		0.412	

Original	Normalized		
	Tag	**n**	**s**
	n	19	10
	s	5	16
Cohen's κ:		0.404	

Original	Original		
	Tag	**n**	**s**
	n	25	4
	s	3	18
Cohen's κ:		0.715	

Table 2: Confusion matrices and annotation agreement (Cohen's κ) between two annotators using original or normalized data.

"Basic" pipe	**Pipe 2**	**Pipe 3**
Tokenizing: ArkTweetNLP		
POS tagging: PDT		
–	**Stem:** no (Sn) / light (Sl) / HPS (Sh)	
–	**Stopwords removal**	
–	–	**Phonetic:** eSpeak (Pe)

Table 3: The preprocessing pipes for Czech (top-down). Combinations of methods are denoted using the appropriate labels, e.g. "Sn" means *1. tokenizing, 2. POS-tagging, 3. no stemming and 4. removing stopwords*. eSpeak stands for a phonetic transcription to International Phonetic Alphabet, which should reduce the effects of grammar mistakes and misspellings.

The most promising results come from the annotation of the original data, thus the rest of the data are annotated in this manner.

We randomly selected 7,000 tweets from the collected data for annotation. The annotators were given just simple instructions without an explicit sarcasm definition (see Section 1): *"A tweet is considered sarcastic when its content is intended ironically / sarcastically without anticipating further information. Offensive utterances, jokes and ironic situations are not considered ironic / sarcastic."*

The complete dataset of 7,000 tweets was independently annotated by two annotators. The inter-annotator agreement (Cohen's κ) between the two annotators is 0.54. They disagreed on 403 tweets. To resolve these conflicts we used a third annotator.

The third annotator has been instructed the same way as the other two. The final κ agreement was measured between the first two annotators, thus it was not affected by the third annotator. Kappa agreements measured on the conflicted states (403 tweets) were 0.4 (annotator 1 vs. annotator 3) and 0.6 (annotator 2 vs. annotator 3).

Preprocessing

Preprocessing steps for handling social media texts in Czech were explored in (Habernal et al., 2013). The preprocessing diagram and its variants is depicted in Table 3. Overall, there are various possible preprocessing "pipe" configurations including "Basic" pipeline consisting of tokenizing and POS-tagging only. We adapted all their preprocessing pipelines. However, as the number of combinations would be too large, we report only the settings with better performance.

4.3 English Dataset

We collected 780,000 (130,000 sarcastic and 650,000 non-sarcastic) tweets in English. The `#sarcasm` hashtag was used as an indicator of sarcastic tweets. From this corpus we created two distributional scenarios based on the work of (Reyes et al., 2013). Refer to Table 4 for the final statistics of the dataset. Part of speech tagging was done using the *Ark-tweet-nlp tool* (Gimpel et al., 2011).

5 Results

For each preprocessing pipeline (refer to table 3) we assembled various sets of features and employed two classifiers. Accuracy (micro F-measure) tends to prefer performance on dominant classes in highly

Dataset \ Tweets	Sarcastic	Non-sarcastic
Czech	325	6,675
English Balanced	50,000	50,000
English Imbalanced	25,000	75,000

Table 4: The tweet distributions in datasets.

Feature Set \ Pipeline	Basic	Sh	ShPe	Sl	SlPe	Sn	SnPe
Baseline 1 (B1): n-gram	54.8	55.3	55.2	55.0	55.0	54.4	55.3
B1 + pattern	55.1	54.4	54.7	55.1	54.8	54.2	54.5
B1 + word-shape pattern	54.6	54.8	55.2	54.4	55.0	54.8	55.1
B1 + punctuation-based	54.7	48.8	48.8	48.8	48.8	53.8	55.5
B1 + pointedness	55.0	54.7	54.7	55.0	55.9	54.8	54.9
B1 + extended pointedness	54.5	48.8	48.8	48.8	48.8	54.7	54.6
B1 + POS n-gram	53.4	54.1	54.2	55.3	55.1	54.2	53.9
B1 + POS word-shape	55.0	55.6	55.2	54.8	54.6	55.8	54.4
B1 + skip-bigram	54.2	54.8	54.2	54.7	**56.0**	54.6	54.4
B1 + POS characteristics + emoticons	55.5	54.7	55.6	55.2	55.4	55.2	53.9
B1 + POS characteristics + emoticons + word-case	53.8	**56.4**	55.5	54.6	55.3	55.9	55.3
Character n-gram (3-6, min. occurrence > 5)	53.0	52.7	53.2	53.9	54.7	52.0	53.2
Baseline 2 (B2)	55.0	55.2	55.4	**56.8**	**56.2**	54.7	54.0
B2 + FS1	52.3	48.8	48.8	48.8	48.8	52.0	52.9
B2 + FS1 + FS2	53.0	48.8	48.8	48.8	48.8	52.2	53.6
B2 + pattern	55.3	55.4	55.7	**56.9**	**56.6**	54.4	53.6
B2 + POS word-shape	55.5	55.8	55.4	**57.0**	**56.3**	55.3	54.7
B2 + POS characteristics + emoticons + word-case	**56.1**	55.7	55.7	**56.9**	**56.1**	55.0	54.3

Table 5: Results on the Czech dataset with the MaxEnt classifier. Macro F-measure, 95% confidence interval $\approx \pm 1.2$. Best results are in bold. **B2**: character n-gram (3-5, min. occurrence > 50) + skip-bigram + pointedness; **FS1**: character n-gram (3-6, min. occurrence > 5) + extended pointedness; **FS2**: POS word-shape + pattern + POS characteristics + emoticons + word-case.

unbalanced datasets (Manning et al., 2008), thus we chose macro F-measure as the evaluation metric (Forman and Scholz, 2010), as it allows us to compare classification results on different datasets. For statistical significance testing, we report confidence intervals at α 0.05. Another applicable methods would be, i.e., two-matched-samples t Test or McNemar's test (Japkowicz and Shah, 2011).

5.1 Czech

Tables 5 and 6 show the results on the Czech dataset. The best result (F-measure 0.582) was achieved by the SVM classifier and a feature set enriched with patterns, utilizing stopwords removal and phonetic transcription in the preprocessing step.

The importance of the appropriate preprocessing techniques for Czech is evident from the improvement of results for various feature sets, e.g., the best result for "Basic" pipeline (see line "B2 + pattern"). Both baselines show improvement on most preprocessing pipelines. The most significant difference is visible on the second baseline with the MaxEnt classifier and the "Sl" pipeline where the F-measure is 0.018 higher than the "Basic" pipeline with no additional preprocessing. The n-gram baseline was significantly outperformed by the SVM classifier with feature sets "B1 + POS characteristics + Emoticons + Word-case" and "B1 + extended pointedness" on the "SnPe" pipeline.

Error Analysis

To get a better understanding of the limitations of our approach, we inspected 100 random tweets from the Czech dataset, which were wrongly classified by the SVM classifier with the best feature combination.

Feature Set \ Pipeline	Basic	Sh	ShPe	Sl	SlPe	Sn	SnPe
Baseline 1 (B1): n-gram	55.8	54.6	54.5	54.6	55.5	56.0	53.9
B1 + pattern	55.6	54.0	54.3	54.6	55.7	55.4	55.6
B1 + word-shape pattern	54.9	55.0	53.8	55.2	55.1	55.4	55.3
B1 + punctuation-based	55.8	48.8	48.8	48.8	48.8	55.7	53.7
B1 + pointedness	55.9	54.5	53.1	54.6	54.3	55.4	54.6
B1 + extended pointedness	56.5	48.8	48.8	48.8	48.8	55.8	56.9
B1 + POS n-gram	54.0	54.1	54.0	54.7	53.4	54.5	53.9
B1 + POS word-shape	55.2	56.4	55.9	55.1	56.0	56.1	55.0
B1 + skip-bigram	55.9	55.3	54.8	55.4	55.0	56.1	55.2
B1 + POS characteristics + emoticons	55.9	54.5	54.1	54.6	54.2	56.7	55.8
B1 + POS characteristics + emoticons + word-case	55.6	54.5	54.3	55.1	55.5	56.3	56.4
Character n-gram (3-6, min. occurrence > 5)	54.6	53.6	53.3	55.2	53.6	53.4	54.9
Baseline 2 (B2)	55.9	56.4	56.3	**57.0**	56.2	**57.1**	55.8
B2 + FS1	52.2	48.8	48.8	48.8	48.8	53.1	52.7
B2 + FS1 + FS2	54.0	48.8	48.8	48.8	48.8	54.4	54.3
B2 + pattern	56.8	**57.0**	56.7	56.5	**57.5**	**57.1**	**58.2**
B2 + POS word-shape	56.5	56.3	**57.2**	56.4	56.1	56.3	**57.8**
B2 + POS characteristics + emoticons + word-case	56.2	55.7	55.8	56.0	56.0	**57.0**	56.0

Table 6: Results on the Czech dataset with the SVM classifier. Macro F-measure, 95% confidence interval $\approx \pm 1.2$. Best results are in bold. **B2**: character n-gram (3-5, min. occurrence > 50) + skip-bigram + pointedness; **FS1**: character n-gram (3-6, min. occurrence > 5) + extended pointedness; **FS2**: POS word-shape + pattern + POS characteristics + emoticons + word-case.

We found 48 false positives and 52 false negatives. The annotators disagreed upon 10% of these tweets.

Non-sarcastic tweets were often about news, reviews, general information and user status updates. In most of the difficult cases of true negatives, the tweet contains a question, insult, opinion or wordplay.

Understanding sarcasm in some tweets was often bound with broader common knowledge (e.g., about news or celebrities), the context known only to the author or authors opinion. Another difficulty poses subtle or sophisticated expression of sarcasm such as *"I'm not sure whether you didn't overdo a bit the first part of the renovation - the demolition. :)"*[10] or *"Conservatism, once something is in the school rules, it must be followed, forever, otherwise anarchy will break out and traditional values will die."*[11]

5.2 English

The results on both balanced and imbalanced English datasets are presented in Table 7. In most cases the MaxEnt classifier significantly outperforms the SVM classifier. The combination of majority of features ("B2 + FS1 + FS2") with the MaxEnt classifier yields the best results for both balanced and imbalanced dataset distributions. This suggests that these features are coherent. While no single feature captures the essence of sarcasm, all features together provide useful linguistic information for detecting sarcasm at textual level.

Balanced distribution Both baselines were surpassed by various combinations of feature sets with the MaxEnt classifier, although in some cases very narrowly ("B1 + punctuation-based" and "B1 + pointedness" feature sets). Although the SVM classifier has slightly worse results, it still performs reasonably, and we even recorded significant improvement over the baseline for "B1 + POS word-shape". The best results were achieved using the MaxEnt classifier with "B2 + FS1 + FS2" (F-measure 0.947) and "B1 + word-shape pattern " (F-measure 0.943) feature sets.

[10] "Jestli jste tu první část rekonstrukce - demolici - trochu nepřehnali . :)"

[11] "Konzervatismus , když je to jednou ve školním řádu , tak se to musí dodržovat , a to navždy , jinak vypukne anarchie a tradiční hodnoty zemřou ."

Dataset	Balanced				Imbalanced			
Classifier	MaxEnt		SVM		MaxEnt		SVM	
Feature set \ Results	Fm	CI	Fm	CI	Fm	CI	Fm	CI
Baseline 1 (B1): n-gram	**93.28**	0.16	92.86	0.16	90.76	0.18	90.44	0.18
B1 + pattern	**94.25**	0.14	**93.13**	0.16	91.86	0.17	90.22	0.18
B1 + word-shape pattern	**94.33**	0.14	**93.17**	0.16	**92.01**	0.17	90.35	0.18
B1 + punctuation-based	**93.32**	0.15	92.84	0.16	90.72	0.18	90.43	0.18
B1 + pointedness	**93.29**	0.16	92.99	0.16	91.00	0.18	90.07	0.19
B1 + extended pointedness	**93.68**	0.15	92.61	0.16	91.07	0.18	89.89	0.19
B1 + POS n-gram	**93.66**	0.15	92.64	0.16	91.20	0.18	89.85	0.19
B1 + POS word-shape	**93.96**	0.15	**93.19**	0.16	91.41	0.17	90.51	0.18
B1 + skip-bigram	**93.63**	0.15	**93.17**	0.16	90.99	0.18	90.48	0.18
B1 + POS characteristics + emoticons	**93.97**	0.15	91.66	0.17	91.69	0.17	89.39	0.19
B1 + POS characteristics + emoticons + word-case	**93.96**	0.15	91.54	0.17	91.61	0.17	88.89	0.19
Character n-gram: (3-6, min. occurrence > 5)	**93.01**	0.16	91.73	0.17	90.36	0.18	88.81	0.20
Baseline 2 (B2)	92.81	0.16	91.67	0.17	90.65	0.18	88.70	0.20
B2 + FS1	**93.82**	0.15	91.56	0.17	91.21	0.18	88.73	0.20
B2 + FS1 + FS2	**94.66**	0.14	91.39	0.17	**92.37**	0.16	88.62	0.20
B2 + pattern	**93.60**	0.15	91.66	0.17	90.86	0.18	88.82	0.20
B2 + POS word-shape	**93.20**	0.16	91.65	0.17	90.82	0.18	88.74	0.20
B2 + POS characteristics + emoticons + word-case	**93.21**	0.16	91.07	0.18	89.98	0.19	88.40	0.20

Table 7: Results on the English dataset with the MaxEnt and SVM classifiers. Macro F-measure (Fm) and 95% confidence interval (CI) are in %. Best results are in bold. **B2**: character n-gram (3-5, min. occurrence > 50) + skip-bigram + pointedness; **FS1**: character n-gram (3-6, min. occurrence > 5) + extended pointedness; **FS2**: POS word-shape + pattern + POS characteristics + emoticons + word-case.

Imbalanced distribution However, data in the real world do not necessarily resemble the balanced distribution. Therefore we have also performed the evaluation on an imbalanced distribution. The MaxEnt classifier clearly achieves the best results. This experiment indicates that combinations of features "B2 + FS1 + FS2" (F-measure 0.924) and "B1, word-shape pattern" (F-measure 0.920) yields the best results for both balanced and imbalanced dataset distribution.

5.3 Discussion

To explain the huge difference in the performance between English and Czech, we conducted an additional experiment in English. We sampled the "big-data" English corpus (100k Tweets) to obtain the same distribution as on the "small-data" Czech corpus (325 sarcastic and 6,675 non-sarcastic Tweets). Feature combination "B2 + FS1 + FS2" achieves an F-measure of 0.734 ± 0.01 (MaxEnt classifier) and 0.729 ± 0.01 (SVM). This performance drop shows that the amount of training data plays a key role (≈ 0.92 on "big-data" vs. ≈ 0.73 on "small-data"). However, these results are still significantly better than in Czech (≈ 0.58). This demonstrates that Czech is a challenging language in sarcasm detection, as in other NLP tasks.

In addition, we also experimented with the Naive Bayes classifier and with delta TF-IDF feature variants (Martineau and Finin, 2009; Paltoglou and Thelwall, 2010) in both languages. However, the performance was not satisfactory in comparison with the reported results.

6 Conclusions

We investigated supervised machine learning methods for sarcasm detection on Twitter. As a pilot study for sarcasm detection in the Czech language, we provide a large human-annotated Czech Twitter dataset containing 7,000 tweets with inter-annotator agreement $\kappa = 0.54$. The novel contributions of our work include the extensive evaluation of two classifiers with various combinations of feature sets on both the Czech and English datasets as well as a comparison of different preprocessing techniques for the

Czech dataset. Our approaches significantly outperformed both baselines adapted from related work[12] in English and achieved F-measure of 0.947 and 0.924 on the balanced and imbalanced datasets, respectively.[13] The best result on the Czech dataset was achieved by the SVM classifier with the feature set enriched with patterns yielding F-measure 0.582. The whole project is available to the community under GPL license at `http://liks.fav.zcu.cz/sarcasm/`. We believe that our findings will contribute to the research outside the mainstream languages and may be applied to sarcasm detection in other Slavic languages, such as Slovak or Polish.

6.1 Future work

We approached the problem mainly from the data-driven perspective (annotation, feature engineering, error analysis). However, we feel that elaborating deep linguistic insights would be helpful to better understand the phenomena of sarcasm on social media (Averbeck, 2013; Averbeck and Hample, 2008; Ivanko et al., 2004; Jorgensen, 1996).

There are also possible extensions to the lexical/morphological features – either in the direction of semi-supervised learning and adding for example features based on latent semantics, topic models, or graphical models popular in the sentiment analysis field (Habernal and Brychcín, 2013; Brychcín and Habernal, 2013), or the direction of deeper linguistic processing in terms of, e.g., syntax/dependecy parsing (but this has limitation given the nature of Twitter data as well as unavailability of such tools for Czech). These deserve further investigation and are planned in future work.

Acknowledgements

Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum, provided under the programme "Projects of Large Infrastructure for Research, Development, and Innovations" (LM2010005), is greatly appreciated. Access to the CERIT-SC computing and storage facilities provided under the programme Center CERIT Scientific Cloud, part of the Operational Program Research and Development for Innovations, reg. no. CZ. 1.05/3.2.00/08.0144, is greatly appreciated.

References

Julie Kane Ahkter and Steven Soria. 2010. Sentiment analysis: Facebook status messages. Technical report, Stanford University. Final Project CS224N.

Joshua M Averbeck and Dale Hample. 2008. Ironic message production: How and why we produce ironic messages. *Communication Monographs*, 75(4):396–410.

Joshua M Averbeck. 2013. Comparisons of ironic and sarcastic arguments in terms of appropriateness and effectiveness in personal relationships. *Argumentation & Advocacy*, 50(1).

Daniel M Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, pages 194–201. Association for Computational Linguistics.

Ben Blamey, Tom Crick, and Giles Oatley. 2012. R U : -) or : -(? character- vs. word-gram feature selection for sentiment classification of OSN corpora. In *Proceedings of AI-2012, The Thirty-second SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 207–212. Springer.

Cristina Bosco, Viviana Patti, and Andrea Bolioli. 2013. Developing corpora for sentiment analysis: The case of irony and senti-tut. *IEEE Intelligent Systems*, 28(2):55–63.

Tomáš Brychcín and Ivan Habernal. 2013. Unsupervised improving of sentiment analysis using global target context. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 122–128, Hissar, Bulgaria, September. INCOMA Ltd. Shoumen, BULGARIA.

[12] Word unigrams, bigrams, trigrams (Liebrecht et al., 2013) and a set of language-independent features (punctuation marks, emoticons, quotes, capitalized words, character n-grams and skip-grams.) (Reyes et al., 2013)

[13] Note that the best result (F-measure 0.715 on the balanced distribution and F-measure 0.533 on the imbalanced distribution) from the related work was achieved by (Reyes et al., 2013) using decision trees classifier.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 107–116, Stroudsburg, PA, USA. Association for Computational Linguistics.

George Forman and Martin Scholz. 2010. Apples-to-apples in cross-validation studies: Pitfalls in classifier performance measurement. *SIGKDD Explor. Newsl.*, 12(1):49–57, November.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 42–47, Stroudsburg, PA, USA. Association for Computational Linguistics.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 581–586, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Guthrie, Ben Allison, Wei Liu, Louise Guthrie, and Yorick Wilks. 2006. A closer look at skip-gram modelling. In *Proceedings of the 5th international Conference on Language Resources and Evaluation (LREC-2006)*, pages 1–4.

Ivan Habernal and Tomáš Brychcín. 2013. Semantic spaces for sentiment analysis. In Ivan Habernal and Václav Matoušek, editors, *Text, Speech, and Dialogue*, volume 8082 of *Lecture Notes in Computer Science*, pages 484–491. Springer Berlin Heidelberg.

Ivan Habernal, Tomáš Ptáček, and Josef Steinberger. 2013. Sentiment analysis in czech social media using supervised machine learning. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 65–74, Atlanta, Georgia, June. Association for Computational Linguistics.

Stacey L Ivanko, Penny M Pexman, and Kara M Olineck. 2004. How sarcastic are you? individual differences and verbal irony. *Journal of language and social psychology*, 23(3):244–271.

Nathalie Japkowicz and Mohak Shah. 2011. *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press.

Julia Jorgensen. 1996. The functions of sarcastic irony in speech. *Journal of Pragmatics*, 26(5):613 – 634.

Michal Konkol. 2014. Brainy: A machine learning library. In Leszek Rutkowski, Marcin Korytkowski, Rafal Scherer, Ryszard Tadeusiewicz, Lotfi Zadeh, and Jacek Zurada, editors, *Artificial Intelligence and Soft Computing*, volume 8468 of *Lecture Notes in Computer Science*, pages 490–499. Springer International Publishing.

Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. 2011. Twitter sentiment analysis: The good the bad and the OMG! In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*. The AAAI Press.

Christine Liebrecht, Florian Kunneman, and Antal Van den Bosch. 2013. The perfect solution for detecting sarcasm in tweets #not. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 29–37, Atlanta, Georgia, June. Association for Computational Linguistics.

Bing Liu and Lei Zhang. 2012. A survey of opinion mining and sentiment analysis. In *Mining Text Data*, pages 415–463. Springer.

Stephanie Lukin and Marilyn Walker. 2013. Really? Well. Apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. In *Proceedings of the Workshop on Language Analysis in Social Media*, pages 30–40, Atlanta, Georgia, June. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Justin Martineau and Tim Finin. 2009. Delta TFIDF: An improved feature space for sentiment analysis. In *Proceedings of the Third International Conference on Weblogs and Social Media, ICWSM 2009, San Jose, California, USA*. The AAAI Press.

A. Montejo-Ráez, E. Martínez-Cámara, M. T. Martín-Valdivia, and L. A. Ureña López. 2012. Random walk weighting over sentiwordnet for sentiment polarity detection on twitter. In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, WASSA '12, pages 3–10, Stroudsburg, PA, USA. Association for Computational Linguistics.

Georgios Paltoglou and Mike Thelwall. 2010. A study of information retrieval weighting schemes for sentiment analysis. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1386–1395, Stroudsburg, PA, USA. Association for Computational Linguistics.

Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From humor recognition to irony detection: The figurative language of social media. *Data Knowl. Eng.*, 74:1–12, April.

Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Language Resources and Evaluation*, 47(1):239–268.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 704–714, Seattle, Washington, USA, October. Association for Computational Linguistics.

Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. ICWSM - a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In William W. Cohen and Samuel Gosling, editors, *Proceedings of the Fourth International Conference on Weblogs and Social Media, ICWSM 2010, Washington, DC, USA, May 23-26, 2010*. The AAAI Press.

Mikalai Tsytsarau and Themis Palpanas. 2012. Survey on mining subjective data on the web. *Data Mining and Knowledge Discovery*, 24(3):478–514, May.

Aline A Vanin, Larissa A Freitas, Renata Vieira, and Marco Bochernitsan. 2013. Some clues on irony detection in tweets. In *Proceedings of the 22nd international conference on World Wide Web companion*, pages 635–636. International World Wide Web Conferences Steering Committee.

Ian H Witten and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

A Three-Step Transition-Based System for Non-Projective Dependency Parsing

Ophélie Lacroix and **Denis Béchet**
LINA - University of Nantes
2 Rue de la Houssinière
44322 Nantes Cedex 3
{ophelie.lacroix,denis.bechet}@univ-nantes.fr

Abstract

This paper presents a non-projective dependency parsing system that is transition-based and operates in three steps. The three steps include one classical method for projective dependency parsing and two inverse methods predicting separately the right and left non-projective dependencies. Splitting the parsing allows to increase the scores on both projective and non-projective dependencies compared to state-of-the-art non-projective dependency parsing. Moreover, each step is performed in linear time.

1 Introduction

Dependency parsing is a particularly studied task and could be a significant step in various natural language processes. That is why dependency parsers should tend to get speed and precision. In recent years, various methods for dependency parsing were proposed (Kübler et al., 2009). Among these methods, transition-based systems are particularly suitable.

The first methods developed for transition-based parsers proposed to produce projective dependency structures (including no crossing dependencies). Then, extended methods were developed to handle the non-projective cases. The non-projective dependency structures admit non-projective dependencies (a dependency is non-projective if at least one word located between the head and the dependent of the dependency does not depend directly or inderectly on the head, see Figure 1 for example). Handling the non-projective cases has been the foundation of the first work concerning the dependency representations (Tesnière, 1959; Melcuk, 1988). Moreover, it is important to successfully parse the non-projective sentences which can be very helpful in processes such as question-answering.

The transition-based parsers achieve interesting overall results for both projective and non-projective analyses. But, in practice, the non-projective methods achieve far lower and variable scores on non-projective dependencies than on projective dependencies. Finding these dependencies is more difficult because the non-projective dependencies are often distant ones. It is then essential to achieve descent scores on non-projective dependencies as well as on projective ones because some languages contain a high rate of non-projective dependencies.

Here we propose to predict separately the projective dependencies from the non-projective ones. Using a mixed dependency representation including both projective and non-projective dependency annotations in one representation, we aim at predicting the projective dependencies in a first step. Taking advantage of the good results of projective dependency parsing, we aim at predicting the non-projective dependencies in a second step.

The formal dependency representation on which we base our work results from the formalism of categorial dependency grammars (CDG) (Dekhtyar and Dikovsky, 2008). It allows to handle the discontinuities of the natural languages. The dependency representation induced is mixed: it associates projective and non-projective dependencies to represent complementary syntactic information in one dependency

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 224–232, Dublin, Ireland, August 23-29 2014.

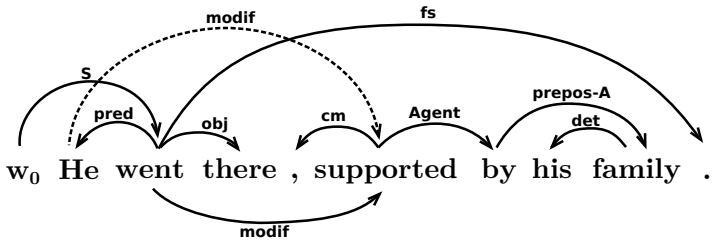


Figure 1: Dependency structure of the sentence "He went there, supported by his family." Anchors are shown below the sentence. Non-projective dependencies appear using a dash line. The other dependencies are plain projective dependencies.

structure. In this representation, each non-projective dependency is paired with a projective one called an anchor. From any dependency structure a projective tree[1] can be extracted.

Our approach is to predict the projective dependency trees first, using a standard and efficient method for projective dependency parsing. In a second step, we use the information (the projective/anchor labelled dependencies) given by the projective parsing to predict the non-projective dependencies. This second step is split into two inverse methods which predict independently the right and left non-projective dependencies. The advantage of the splitting is to perform the parsing in linear time and achieve better scores on non-projective dependencies.

Finally, in order to evaluate the efficiency of our method, we apply it on data annotated according to the formalism of the categorial dependency grammar. The data consists on a treebank containing both projective and non-projective trees associated with sentences of French.

2 Related Work

Our approach is similar to a post-processing method for retrieving the non-projective dependencies. In a way, our work is then analogous to the work of Hall and Novák (2005) who apply a post-processing method after converting constituency trees into dependency ones since the conversion can not automatically recover the non-projective relations.

Moreover, taking advantage of the efficiency of projective dependency methods to predict the non-projective dependencies is a technique used by Nivre and Nilsson (2005) in their pseudo-projective method. They projectivize the dependency trees before parsing in order to apply a projective method first and apply an inverse transformation to retrieve the non-projective dependencies. For our method, we do not need to projectivize the trees since the dependency representation we use includes both projective and non-projective annotations in one representation. But we can employ the projectivization method to build such data adding the generated projective dependencies to the non-projective structure as if they were artificial anchors. Consequently, our approach can be applied on treebank containing standard non-projective trees.

The advantage of our method is that the information that is useful for retrieving the non-projective dependencies is not predicted during the projective parsing which makes the projective and non-projective steps completely independent from each other. Moreover, the non-projective steps are data-driven and remain linear.

3 Representation and Formalism

Our work is based on dependency structures combining projective and non-projective annotations in one representation. In such a representation the projective dependencies bring both local and syntactic information while the non-projective dependencies bring only syntactic information (i.e. the relation shared by the dependents). Thus, each non-projective dependency is paired with a projective relation (called anchor) determining the position of the dependent in the sentence. Figure 1 presents a non-projective dependency structure of a sentence which illustrates the use of a projective relation (anchor)

[1] Composed of projective dependencies and anchors of non-projective dependencies, see Section 3.

and a non-projective dependency to represent a discontinuous relation: "supported" is a modifier for the pronoun "he".

The dependency representation is induced by a particular formalism: the class of the categorial dependency grammars (CDG). The categories of the grammars correspond to the dependency labels. The rules $\mathbf{L}^1$, $\mathbf{I}^1$ and $\mathbf{\Omega}^1$, presented in Table 1, are the classical left elimination rules of categorial grammars. Only the left rules are shown but there are symmetrical right rules. These rules allow to define the projective dependencies and anchors. Moreover, CDGs are classical categorial grammars in which the notion of polarized valencies was added. Each of the three first rules includes the concatenation of potentials (such as P, P_1, P_2) which are lists of polarized valencies. The polarized valencies are label names associated with a polarity (south-west $\swarrow$, north-west $\nwarrow$, north-east $\nearrow$ and south-east $\searrow$). They represent the ends of the non-projective dependencies. The south polarities indicate an incoming non-projective dependency and the north valencies indicate an outgoing non-projective dependency. The rule $\mathbf{D}^1$ allows the elimination of dual pairs of polarized valencies, following the **FA** principle.

First Available (FA) principle: the closest dual polarized valencies with the same name are paired.

Thus, the elimination of the dual pairs $(\swarrow C)$ $(\nwarrow C)$ and $(\nearrow C)$ $(\searrow C)$ defines respectively left and right non-projective dependencies labelled by C.

$\mathbf{L}^1$	$C^{P_1}\,[C\backslash\beta]^{P_2} \vdash [\beta]^{P_1P_2}$
$\mathbf{I}^1$	$C^{P_1}\,[C^*\backslash\beta]^{P_2} \vdash [C^*\backslash\beta]^{P_1P_2}$
$\mathbf{\Omega^1}$	$[C^*\backslash\beta]^{P} \vdash [\beta]^{P}$
$\mathbf{D}^1$	$\alpha^{P_1(\swarrow C)P(\nwarrow C)P_2} \vdash \alpha^{P_1PP_2}$, if $(\swarrow C)(\nwarrow C)$ satisfies the **FA** principle

Table 1: (Left) Rules of the categorial dependency grammars.

4 Method

We conduct a three-step transition-based parsing. We choose the arc-eager method of Nivre (2008) to perform the first step. Note that any projective method for dependency parsing would also be appropriate to perform this step. The second and third steps are methods which go through the sentence (respectively from left to right and from right to left) in order to find the non-projective dependencies.

4.1 Projective Dependency Parsing

The arc-eager method is an efficient transition-based method for projective dependency parsing. A transition system is composed of a set of configurations (states), a set of transitions (operations on the configurations), an initial configuration and a set of terminal configurations. The transition-based parsing consists in applying a sequence of transitions to configurations in order to build a dependency structure. For the arc-eager method, a configuration is a triplet $\langle \sigma, \beta, A \rangle$ where:

- σ is a stack of partially treated words;
- β is a buffer of non-treated words;
- A is a set of dependencies (the partially built dependency structure).

The dependencies are described by triplets such as (k, l, i) where k is the position of the head, l is the label of the dependency and i is the position of the dependent. The set of transitions includes three transitions which are evolutions of the standard transitions of the system of Yamada and Matsumoto (2003) plus the Reduce transition which allows to delete the first word of the stack when this one shares no dependency with the first word of the buffer. The standard Right-Arc and Left-Arc are renamed respectively as Local-Right and Local-Left since these transitions only add local dependencies (whitout distinction between projective ones and anchors). The Shift transition pops the first word from the buffer

Transition	Application	Condition
Local-Left(l)	$(\sigma \mid w_i, w_j \mid \beta, A) \Rightarrow (\sigma, w_j \mid \beta, A \cup \{(j, l, i)\})$	$i \neq 0 \wedge \neg\exists k \exists l'(k, l', i) \in A$
Local-Right(l)	$(\sigma \mid w_i, w_j \mid \beta, A) \Rightarrow (\sigma \mid w_i w_j, \beta, A \cup \{(i, l, j)\})$	$\neg\exists k \exists l'(k, l', j) \in A$
Reduce	$(\sigma \mid w_i, \beta, A) \Rightarrow (\sigma, \beta, A)$	$\exists k \exists l(k, l, i) \in A$
Shift	$(\sigma, w_i \mid \beta, A) \Rightarrow (\sigma \mid w_i, \beta, A)$	

Table 2: Transitions of the arc-eager method.

and pushes it into the stack. The Reduce transition pops the first word from the stack. The effects of the transitions on configurations are detailed in Table 2.

For a given sentence $W = w_1...w_n$, the initial configuration of the transition-based system is defined as follows: $([w_0], [w_1, ..., w_n], \emptyset)$ where w_0 is the root of the structure. And any terminal configuration is of the form: $([w_0], [], A')$ where A' contains the fully projective dependency/anchor structure for the sentence W[2].

This step should produce the projective dependency structure of Figure 2 for the sentence "Il y est allé, soutenu par sa famille" (french equivalent of the sentence seen in Figure 1).

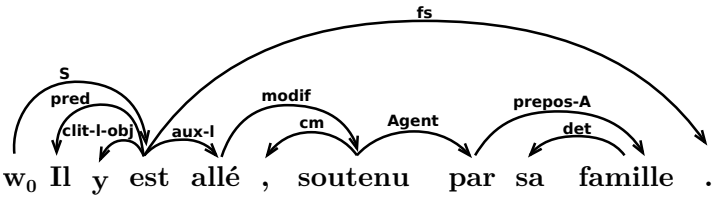


Figure 2: Projective dependency structure of the sentence "Il y est allé, soutenu par sa famille".

4.2 Adding Non-Projective Dependencies

With the aim of retrieving non-projective dependencies we propose two inverse methods also inspired by transition-based systems. For these methods, the configuration is a quadruplet $\langle \sigma, \beta, \theta, A \rangle$ where σ, β and A are the same stack, buffer and set of arcs as those defined for projective parsing in the previous subsection and θ is a list of polarized valencies. The valencies have the same role here as in the formalism of the categorial dependency grammars (detailed in section 3). They define the ends of the non-projective dependencies. Therefore, our idea is to go through the sentence in order to predict, for each word, whether a non-projective dependency could end on the word (by adding valency $\swarrow l$ or $\searrow l$ in the list θ) or should start from it (by adding valency $\nwarrow l$ or $\nearrow l$ in the list θ). As soon as dual valencies are collected in θ, they are removed from it (according to the **FA** principle) and the corresponding non-projective dependency is added to the set of dependencies.

In the second step, the valencies associated with the left dependencies are computed, i.e. the valencies of the form $\swarrow l$ and $\nwarrow l$. The sentence is linearly covered from left to right, as in the previous projective step. Details of the transitions are presented in Table 3. The Shift transition is the same as during the previous step and allows to cover the sentence classically from left to right. The PutValency transition makes possible to predict, for the first word of the buffer, exactly one southwest valency $\swarrow l$, which means that a left dependency labelled l can end on this word. In addition, the valency is concatenated at the end

Transition	Application	Condition
PutValency($\swarrow l$)	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma \mid w_i, \beta, \theta \swarrow l^i, A)$	$\swarrow l^i \notin \theta$
Dist-Left($\nwarrow l$)	$(\sigma, w_j \mid \beta, \theta_1 \swarrow l^i \theta_2, A) \Rightarrow (\sigma, w_j \mid \beta, \theta'_1\theta'_2, A \cup \{(j, l, i)\})$	$\swarrow l \notin \theta_2 \wedge \forall k \swarrow k^i \notin \theta'_1\theta'_2$
Shift	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma \mid w_i, \beta, \theta, A)$	

Table 3: Transitions of the left non-projective method.

[2]The words which were not attached during the parsing are automatically attached to the root node w_0.

of θ. The transition Dist-Left is applied when the first word of the buffer receives the dual valency (i.e. a valency of the form $\nwarrow l$). If at least one valency $\swarrow l$ belongs to θ then the last one is removed from θ and the non-projective dependency corresponding to the pair of dual valencies $\swarrow l \nwarrow l$ (left non-projective labelled l) is added to A.

Therefore, for a given sentence, the initial configuration of this system is $([w_0], [w_1, ..., w_n], (), A')$ where A' is the projective dependency structure predicted by the arc-eager method. And the terminal configuration is a quadruplet of the form $([w_0, ..., w_n], [], \theta', A'')$ where θ' could contain southwest valencies which did not match with their dual and A'' is a partially non-projective dependency structure.

The third step uses the inverse method of the previous step and allows to predict right non-projective dependencies. In this method, the sentence is linearly covered from right to left. The initial configuration $([w_0, .., w_{n-1}], [w_n], (), A'')$ contains the partial dependency structure A'' produced by the last method and the terminal configuration $([w_0], [w_1, ..., w_n], \theta'', A''')$ contains the fully non-projective dependency structure A'''. The transitions used here are presented in Table 4. This time, the PutValency transition adds only southeast valencies ($\searrow l$) at the beginning of θ and pops the first word of σ to push it into β. The Dist-Right transition adds a right non-projective dependency in the set of arcs by predicting a dual valency of the form $\nearrow l$. Finally, the RShift transition pops the first word of σ to push it in β.

Transition	**Application**	**Condition**
PutValency($\searrow l$)	$(\sigma \mid w_i, \beta, \theta, A) \Rightarrow (\sigma, w_i \mid \beta, \searrow l^i \theta, A)$	$\searrow l^i \notin \theta$
Dist-Right($\nearrow l$)	$(\sigma \mid w_j, \beta, \theta_1 \searrow l^i \theta_2, A) \Rightarrow (\sigma \mid w_j, \beta, \theta'_1 \theta'_2, A \cup \{(j, l, i)\})$	$\searrow l \notin \theta_1 \wedge \forall k \searrow k^i \notin \theta'_1 \theta'_2$
RShift	$(\sigma \mid w_i, \beta, \theta, A) \Rightarrow (\sigma, w_i \mid \beta, \theta, A)$	

Table 4: Transitions of the right non-projective method.

The splitting of the non-projective dependencies prediction on two different methods is essential to find the right non-projective dependencies as well as the left ones. Practically, finding the head (i.e. the $\nearrow l$ and $\nwarrow l$ valencies) of a non-projective dependency is easier once the dependent (i.e. the $\searrow l$ and $\swarrow l$ valencies) has been previously predicted. Indeed, the prediction system benefits of information about the presence of the head valency in θ to predict the dual valency. Moreover, the heads are predicted more efficiently whether the projective dependency associated with the word was predicted with the right label during the first parsing step. The next section presents the prediction system and the features needed to proceed good transition predictions.

The application of these two steps on the sentence seen in Figure 2 are shown on Table 5. The

Transition	**Configuration**			
	([w$_0$],	[Il,...,.],	(),	A)
Shift	$\Rightarrow$ ([w$_0$,Il],	[y,...,.],	(),	A)
PutValency($\swarrow$clit-l-obj)	$\Rightarrow$ ([w$_0$,Il],	[y,...,.],	($\swarrow$clit-l-obj),	A)
Shift	$\Rightarrow$ ([w$_0$,...,y],	[est,...,.],	($\swarrow$clit-l-obj),	A)
Shift	$\Rightarrow$ ([w$_0$,...,est],	[alle,...,.],	($\swarrow$clit-l-obj),	A)
DistLeft($\swarrow$clit-l-obj)	$\Rightarrow$ ([w$_0$,...,est],	[alle,...,.],	(),	$A_1 = A \cup \{(4,\text{clit-l-obj}, 2)\}$)
Shift **(x6)**	$\Rightarrow$ ([w$_0$,...,.],	[],	(),	A_1)
	([w$_0$,... ,famille],	[.],	(),	A_1)
RShift	$\Rightarrow$ ([w$_0$,...,],	[famille,.],	(),	A_1)
RShift **(x3)**	$\Rightarrow$ ([w$_0$,...,.],	[soutenu,...,.],	(),	A_1)
PutValency($\swarrow$modif)	$\Rightarrow$ ([w$_0$,...,.],	[soutenu,...,.],	($\swarrow$modif),	A_1)
RShift **(x5)**	$\Rightarrow$ ([w$_0$],	[il,...,.],	($\swarrow$modif),	A_1)
DistLeft($\swarrow$modif)	$\Rightarrow$ ([w$_0$],	[il,...,.],	(),	$A_2 = A_1 \cup \{(1,\text{modif}, 6)\}$)

Table 5: Transition sequences of the left and right non-projective steps on the sentence in Figure 2.

projective structure built during the first step (Figure 2) is substituted to the set of arcs A in the initial configuration of the left non-projective step. The non-projective dependency structure A_2 provided at the end of the right (final) non-projective step is presented in Figure 3.

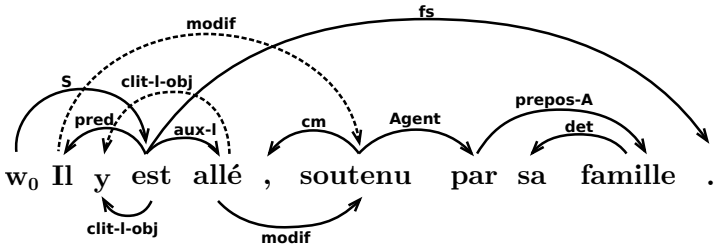


Figure 3: Non-projective dependency structure of the sentence in Figure 2.

4.3 Oracle

The transition-based systems are particularly interesting for deterministic data-driven parsing. Associated with a statistical method, such as a probabilistic graphical model or a linear classifier, and suitable features, the prediction of the transitions is very efficient. It ensures a deterministic parsing in linear time for both the projective arc-eager method and our two non-projective post-processing methods.

Previous work such as (Yamada and Matsumoto, 2003) shows that support vector machines (SVM) allow to achieve good scores on dependency parsing when associated with a transition-based system. Therefore, we chose to use this classifier to predict the transitions of our two post-processing methods. Moreover, the arc-eager method (i.e. *nivreeager*) being already successfully implemented and optimized, we decided to use the MaltParser (Nivre et al., 2007) to perform the projective dependency parsing.

For this projective step, the features are composed of classical features such as the word forms, POS-tags and dependency labels of the current words (the first elements of the stack and the buffer), their neighbors and their attached dependents. For the two non-projective steps the feature pattern includes in addition some features on the projective head of the first word of the buffer and the list of the valencies remaining in θ. The feature pattern is presented in Table 6. Nevertheless, the SVM model bears only numerical features. And each feature must be converted into a binary feature determining its absence or presence. For the valencies, the features denotes the absence or presence of each possible valency label in θ.

Feature Pattern	
• Word forms:	• POS-tags:
$w_{\{i-1,i+1\}}$	$t_{\{i-2,i+2\}}$
w_j	t_j
• Labels:	• Valencies:
l_j (projective dependency label)	$(v_0, ..., v_k)$ (the list of valencies in θ)
$(l_{j_1}, ..., l_{j_n})$ (the list of dependency labels)	

Table 6: Features for the prediction of transition in the two inverse methods. i is the position of the first word in β, j is the position of the head of w_i, the list of dependency labels is the list of labels of the right or left dependents of the head (depending on the right or left method).

5 Evaluation

In order to evaluate the efficiency of our approach, we decided to experiment on a dependency treebank for which the data were annotated following the formalism of the categorial dependency grammars[3]. We call this treebank the CDG Treebank 1. Moreover, in order to evaluate the adaptation of our method

[3]The treebank is not yet publicly available. But the authors have made it available to us.

to standard treebanks we would like to perform the method on data for which the anchors would have been artificially created. Therefore, we build a second treebank from the first one, which we call the CDG Treebank 2, in which the original anchors are replaced by artificial anchors generated by the projectivization step of the pseudo-projective method of Nivre and Nilsson (2005).

5.1 Non-Projective Dependency Treebank

The CDG Treebank 1 contains 3030 sentences of French, each paired with a dependency structure. The dependency structures are composed of both projective and non-projective dependencies. Out of the 37580 dependencies (excluding the anchor ones), 3.8% are non-projective. Hence, 41% of the dependency structures of the treebank contain at least one non-projective dependency.

The data were annotated semi-automatically using the CDG Lab (Alfared et al., 2011), a development environment dedicated to large scale grammar and treebank development. Thus, the annotations followed the formalism proposed by the categorial dependency grammar of French (Dikovsky, 2011). The labels of the dependencies are the 117 categories used by the grammar. Most of the dependency labels (89) are exclusively associated with projective dependencies. 23 labels can be associated both with projective and non-projective dependencies. Among these ones the most frequent are clitics, negatives, objects, reflexives and copredicates. In most of the cases, clitics, negatives and reflexives are associated with short dependencies (generally, one or two words separate the head from the dependent) whereas copredicates or apposition are often associated with distant dependencies (the heads and dependents can be located at the opposite ends of the sentence). Four dependency labels are exclusively associated with non-projective dependencies, they are particular cases of aggregation, copula, comparison and negation.

The grammar and the treebank were developed simultaneously. Consequently, a large part of the sentences were used to develop the grammar and were chosen to cover as much as possible the syntactic phenomenon of French. The treebank contains sentences from newspaper, 19^{th} and 20^{th} century literary works and plain language.

To build the CDG Treebank 2, we removed the anchors of the dependency structures of the CDG Treebank 1 and added the projective dependencies generated by projectivization[4]. Note that, 90.9% of the anchors are the same between the two CDG treebanks.

5.2 Experimental Settings

We evaluate our method through a 10-fold cross-validation on the non-projective dependency treebank. First, we train the prediction models (the MaltParser training model and the SVM model) on each training set containing 90% sentences of the treebank. Second, each fold of our testing data sets is tagged with Part-Of-Speech tags using Melt (Denis and Sagot, 2009), a POS-tagger that achieves high score on French. Then the sentences are parsed.

In order to estimate the benefit of our method, our results are compared with those obtained by the methods proposed by the MaltParser. The table shows the results of the methods that give the best results among the non-projective ones and the best results among the projective ones (associated with the pseudo-projective method (Nivre and Nilsson, 2005)):

- the *covnonproj* (non-projective) method inspired by Covington (2001);
- the *nivreeager* (projective) method associated with the pseudo-projective method.

For a fair comparison, the scores are computed on the same data for each experiments, i.e. on the non-projective structures minus the anchors and the dependencies combined with punctuations.

Moreover, in order to demonstrate that our method can be applied successfully on standard treebanks, the experiments are performed on the CDG Treebank 1 an 2. The comparison scores that are used in these experiments are:

[4]The labels of the artificial anchors do not contain additional encoded information. They are identical to the labels of the non-projective dependencies.

- the label accuracy (LA), i.e. the percentage of words for which the correct label is assigned;
- the unlabelled attachment score (UAS), i.e. the percentage of words for which the correct dependency is assigned;
- the labelled attachment score (LAS), i.e. the percentage of words for which the correct labelled dependency is assigned.

5.3 Experimental Results

The results of the experiments are presented in Table 7. First, we notice that the scores relating to projective dependencies of our method, both for CDG Treebank 1 (3) and CDG Treebank 2 (4), are better than those obtained by the covnonproj method (1) and equivalent to the pseudo-projective method (2). We assume that finding non-projective dependencies at the same time as the projective ones is more difficult than finding projective dependencies only. Moreover, the scores on non-projective dependencies

	All dependencies			**Projective Dep.**			**Non-projective Dep.**		
	LA	UAS	LAS	LA	UAS	LAS	LA	UAS	LAS
(1) covnonproj	82.2	85.5	78.0	82.8	86.2	78.7	68.7	68.7	62.7
(2) pseudoproj [5]	83.6	85.9	78.7	**84.1**	**87.0**	**79.7**	73.5	56.9	53.5
(3) non-projLR (CDGTbk1)	**83.7**	**86.3**	**79.1**	**84.1**	86.9	79.6	**75.5**	70.2	66.3
(4) non-projLR (CDGTbk2)	**83.7**	86.2	79.0	**84.1**	86.9	79.5	**75.5**	**70.5**	**66.7**

Table 7: Results of the non-projective dependency parsing comparing the MaltParser methods (1) and (2) with ours (3).

are particularly interesting. Our method achieves far better scores on non-projective dependencies than the other two. The label accuracy (LA) achieves significantly better scores (+6.8) than the covnonproj method. Indeed, the projective step allows to find the anchors which are a kind of projective dependencies, so there are easier to predict than the non-projective dependencies. Thus, the label accuracy of the non-projective dependencies takes advantage of the good results of the anchors which were not paired with a non-projective dependency during the second and third parsing steps. Concerning the attachment scores, our method still outperforms the two others. Globally, our method allows to recover the head of the non-projective dependencies more successfully.

The non-projective dependencies can be also compared depending on their direction. The left non-projective dependencies achieve far better scores (75.0% LAS) than the right non-projective dependencies (42.7% LAS). We know that the non-projective step performed from right to left is essential to recover the right non-projective dependencies. In fact, finding the right non-projective dependencies by performing the non-projective step from left to right seems almost infeasible because it is essential to find the dependent first. Therefore, the problem comes essentially from the bad prediction of the anchors during the projective step. Indeed, only 51.4% of the words associated with a right non-projective dependency receive the correct label (LA), compared with 84.2% for those associated with left non-projective dependencies. The under-representation of the right non-projective dependencies (25% of the non-projective dependencies) in the treebank is a first explanation. But, even the more frequent labels (associated with right non-projective dependencies) achieve low scores. Moreover, we noticed that even the right projective dependencies always achieve lower scores than the left projective dependencies. This problem may suggest that the use of a left-to-right projective method is not appropriate to predict the right dependencies.

Furthermore, we note that our method achieve equivalent scores on CDG Treebank 1 and CDG Treebank 2, and even slightly better for non-projective dependencies with the use of artificial anchors. This suggest that our method could be succesfully applied to standard treebanks in which artificial anchors would have been added.

[5] The pseudo-projective method were applied with the option "path" for projectivization and deprojectivization.

6 Conclusion

We propose a three-step method retrieving separately the projective dependencies and anchors, the left non-projective dependencies and the right non-projective dependencies through the use of a mixed dependency representation. The projective step and the two non-projective steps are performed in linear time and allow to outperform state-of-the-art transition-based scores on non-projective dependencies. The method needs a learning corpus that associate to each non-projective dependency a projective anchor. Thus the method is well adapted to CDG treebanks. But we showed that the method can be applied to standard treebanks by adding artificial anchors with the use of a method of projectivization.

One of the advantages of our method is a significant improvement on the label accuracy for the non-projective dependencies. The efficiency of the two non-projective methods depends on the good results of the projective parsing. Moreover, performing the non-projective parsing from left-to-right and from right-to-left raises interesting questions on how to recover the right and left dependencies for both projective and non-projective methods.

Acknowledgement

We want to thank Danièle Beauquier and Alexander Dikovsky for giving us the CDG Treebank on which we experimented our system. Moreover, we want to thank all our reviewers : the anonymous reviewers of Coling for their accurate reviews, and the members of the team TALN of the University of Nantes (Colin de la Higuera, Florian Boudin and the master students) who reviewed our work with a fresh eye.

References

Ramadan Alfared, Denis Béchet, and Alexander Dikovsky. 2011. CDG Lab: a Toolbox for Dependency Grammars and Dependency Treebanks Development. In *Proceedings of the International Conference on Dependency Linguistics*, DEPLING 2011, pages 272–281, Barcelona, Spain, September.

Michael A. Covington. 2001. A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102.

Michael Dekhtyar and Alexander Dikovsky. 2008. Generalized categorial dependency grammars. In *Trakhtenbrot/Festschrift*, LNCS 4800, pages 230–255. Springer.

Pascal Denis and Benoît Sagot. 2009. Coupling an Annotated Corpus and a Morphosyntactic Lexicon for State-of-the-Art POS Tagging with Less Human Effort. In *Proceedings of the Pacific Asia Conference on Language, Information and Computation*, PACLIC 2009, Hong Kong, China.

Alexander Dikovsky. 2011. Categorial Dependency Grammars: from Theory to Large Scale Grammars. In *Proceedings of the International Conference on Dependency Linguistics*, DEPLING 2011, September.

Keith Hall and Václav Novák. 2005. Corrective modeling for non-projective dependency parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, IWPT 2005, pages 42–52.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. 2009. Dependency Parsing. *Synthesis Lectures on Human Language Technologies*, 1(1):1–127.

Igor Melcuk. 1988. *Dependency syntax : Theory and Practice*. State University of New York Press.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective Dependency Parsing. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 99–106, Ann Arbor, Michigan.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Glsen Eryigit, Sandra Kbler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A Language-Independent System for Data-Driven Dependency Parsing. *Natural Language Engineering*, 13:95–135, 6.

Joakim Nivre. 2008. Algorithms for Deterministic Incremental Dependency Parsing. *Comput. Linguist.*, 34(4):513–553, December.

Lucien Tesnière. 1959. *Éléments de syntaxe structurale*. Klincksieck.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of the International Conference on Parsing Technologies*, IWPT 2003, pages 195–206.

Collaborative Topic Regression with Multiple Graphs Factorization for Recommendation in Social Media

Qing Zhang
Key Laboratory of Computational Linguistics (Peking University)
Ministry of Education, China
zqicl@pku.edu.cn

Houfeng Wang*
Key Laboratory of Computational Linguistics (Peking University)
Ministry of Education, China
wanghf@pku.edu.cn

Abstract

With a large amount of complex network data available from multiple data sources, how to effectively combine these available data with existing auxiliary information such as item content into the same recommendation framework for more accurately modeling user preference is an interesting and significant research topic for various recommender systems. In this paper, we propose a novel hierarchical Bayesian model to integrate multiple social network structures and content information for item recommendation. The key idea is to formulate a joint optimization framework to learn latent user and item representations, with simultaneously learned social factors and latent topic variables. The main challenge is how to exploit the shared information among multiple social graphs in a probabilistic framework. To tackle this challenge, we incorporate multiple graphs probabilistic factorization with two alternatively designed combination strategies into collaborative topic regression (CTR). Experimental results on real dataset demonstrate the effectiveness of our approach.

1 Introduction

Many real-life data have representations in the form of multiple views (Liu et al., 2013). For example, web pages usually consist of both text content and hyperlink information; images on the web have relevant tags associated with their content. In addition, it is also common that in real networks comprising multiple types of nodes are connected by multiple types of links, forming heterogeneous information networks (HIN) (Huang et al., 2012). For example, in scientific community, various types of links are formed for different types of objects, i.e., author writes paper, venue publishes paper, reader labels tag, and so on. Therefore, with a large amount of complex network data available from multiple data sources, how to effectively combine this kind of rich structure with other auxiliary information such as content information into the same recommendation framework is an interesting and significant research topic for various recommender systems. This paper aims to model multiple social graphs into a principled hierarchy Bayesian framework to improve recommending performance.

The basic idea in this paper is inspired by multi-view learning approach (Liu et al., 2013), i.e., leveraging the redundancy and consistency among distinct views (Kumar et al., 2011) to strengthen the overall performance. We extend this idea (Liu et al., 2013) originally for clustering problem to deal with rating scarcity problem when modeling user preference for recommendation. Just as in general multi-view learning, each view of objective function is assumed to be capable of correctly classifying labeled examples separately. Then, they are smoothed with respect to similarity structures in all views. Similarly, in this paper, we assume that our individual views of multiple user social relations are similar and complementary with a shared latent structure.

However, different from multi-view clustering problem, our goal is to recover a sparse rating matrix with a large number of missing user-item pairs rather than merely exploiting cluster structure with full task information. Thus, the straightforward multi-view representation of the objective (rating matrix) is

*Corresponding author

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 233–244, Dublin, Ireland, August 23-29 2014.

not available. Instead, we use side information (user social graphs) to exploit multi-view learning for improving collaborative filtering (CF). As a result, collaborative topic regression (CTR) (Wang and Blei, 2011) is employed as our basic learning framework with side information. Recently, CTR has gained considerable attention due to its well-defined mathematical framework and strong performance on user behavior prediction for various real-world applications, such as document recommendation (Li et al., 2013), tag recommendation (Wang et al., 2013), music recommendation (Purushotham et al., 2012), celebrity recommendation (Ding et al., 2013) and vote prediction (Kang and Lerman, 2013). However, all the extensions above merely focus on a single view of user or item relation. In reality, a large amount of diverse social graphs data are widely existed and particularly valuable for mutually reinforcing each other. Therefore, it should be well considered. Taking this into consideration, we extend CTR with multiple social graphs factorization for recommender systems.

The main challenge of incorporating multiple relations into CTR is how to exploit the shared information among multiple social networks and how to further deal with it to recover sparse rating matrix in a probabilistic framework. Previous efforts, purely to address the first issue for clustering problem, are usually to seek a weak consensus (Liu et al., 2013) learnd from data jointly with clustering process. Intuitively, consensus can be seen as a latent cluster structure shared by different views. Thus, it means that learning from different views should be softly regularized towards a common latent structure. However, it is not easy to directly formulate it in a probabilistic framework, because weak consensus modeling can not be separated from a joint higher task, i.e., recovering spare rating matrix, in our case.

To tackle this challenge, we propose a novel hierarchical Bayesian model with multiple social graphs factorization. We exploit two ways of modeling shared information for multi-view based recommendation. One is for heterogeneous network by directly modeling different view specific latent structures with consensus for user representation. The other is for homogeneous case, which can be used as a transformed version of heterogeneous relations. In contrast with the first strategy, we model the latter using a shared latent social structure for all views but with different user representations. Thus, we can relax strong consensus assumption in our heterogeneous case, through linear combination of each sub-latent user with maintained sharing mechanism. The multiple graphs factorization process in the proposed model can be seen as a regularization approach on each latent user for better uncovering user-item latent structures. Although, regularization technique for modeling multiple heterogeneous networks is a hot research topic, in clustering study from an algebra view (Liu et al., 2013; Kumar et al., 2011), not much is known on using it for collaborative recommendation problems in a more complex probabilistic setting.

The following sections will discuss those in details and we use the terms network and graph interchangeably throughout this paper.

2 Preliminaries

In this section, we briefly review collaborative topic regression (CTR) (Wang and Blei, 2011), as the foundation of our proposed model. Figure 1 (left) shows the graphical representation of CTR, which combines the merits of traditional collaborative filtering and probabilistic topic modeling. Specifically, the key mechanism of CTR is that using topic vectors learned from LDA (Blei et al., 2003) jointly controls the prior distribution of latent items in original matrix factorization process of CF. The generative process is described as follows:

- For each user i,
  - draw user latent vector $u_i \sim N(0, \lambda_u^{-1} I)$, multivariate Gauss distribution with zero mean.
- For each item j,
  - draw topic proportions $\theta_j \sim Dirichlet(\alpha)$, Dirichlet distribution.
  - draw item latent offset vector $\epsilon_j \sim N(0, \lambda_v^{-1} I)$, and set the item latent vector as $v_j = \epsilon_j + \theta_j$.
  - For each word w_{jn}
    - draw topic assignment $z_{jn} \sim Mult(\theta)$, Multinomial distribution.
    - draw word $w_{jn} \sim Mult(\beta_{z_{jn}})$, Multinomial distribution.
- For each user-item pair (i, j),

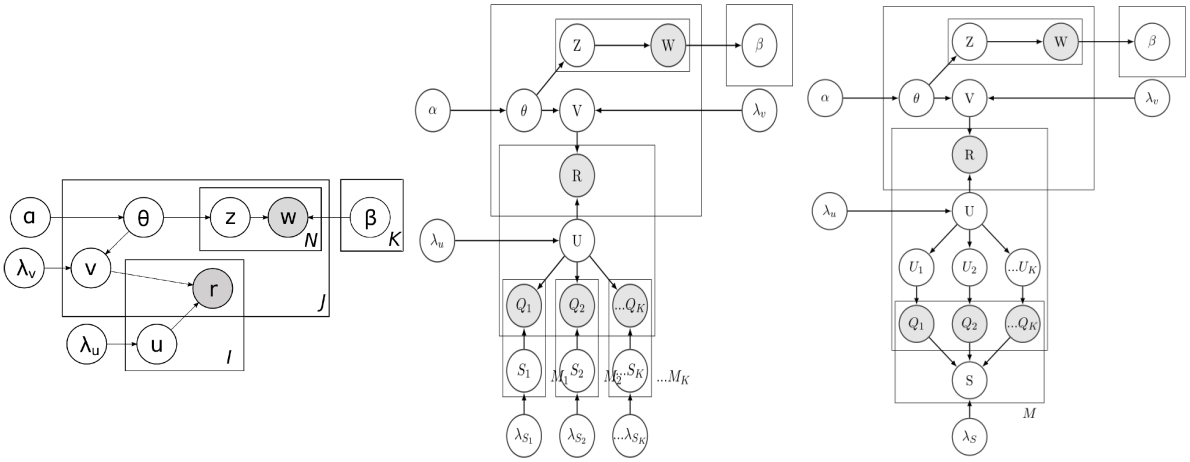


Figure 1: CTR (left), heterogeneous CTR-MGF (middle), homogeneous CTR-MGF (right).

– draw the response $r_{ij} \sim N(u_i^T v_j, c_{ij}^{-1})$, univariate Gauss distribution, where c_{ij} is a confidence parameter for rating r_{ij}, $a > b$. $c_{ij} = a$ (higher confidence), if $r_{ij} = 1$, and $c_{ij} = b$, if $r_{ij} = 0$.

However, CTR does not take the complex social network information, which is available and crucial in many real-world applications, into consideration.

3 CTR-MGF: Collaborative Topic Regression with Multiple Graphs Factorization

In this section, we discuss our proposed method, called CTR with multiple graphs factorization (CTR-MGF). Our model is a generalized hierarchical Bayesian model which jointly learns latent user, item and multiple latent social factor spaces. Different from previous approaches, our method extends CTR to multiple complex networks setting instead of one particular type of relation for user or item. Moreover, we consider two real general contexts in various practical applications. One is the context of heterogeneous networks. The other is the context of homogeneous networks. It is noted that, for relative simplicity, in this paper we only consider user oriented complex network. The graphical representation of our models in Figure 1 (middle and right) takes $K = 3$ networks as illustration, which can be arbitrary in our derivation. It is also easy to see that Purushotham et al. (2012) is a special case of our proposed model, which is not equipped with graph sharing mechanism.

3.1 Model Notations

Each social matrix Q corresponds to a social network structure $G = \{V, E\}$, where users and their social relations are represented as vertex set V and edge set E in network structure G, respectively. The element q_{im} in Q denotes the binary relation between user 'i' and graph specific feature 'm' in heterogeneous network or the relation between two users 'i' and 'm' in homogeneous network.

3.2 CTR-MGF for Heterogeneous Networks

Heterogeneous network is formed by multiple types of nodes being connected by multiple types of links. The key characteristic of heterogeneous network is that the sizes of feature dimensions are different among multiple social graphs. For example, in a social music sharing system such as LastFM, each user has multiple heterogeneous relations associated with the interested music, i.e., user-artist, user-tag, and so on. Our model can handle all these relations in the proposed framework, CTR-MGF. Specifically, the generative process of CTR-MGF for heterogeneous networks is listed as follows:

- For each item j,
  - draw topic proportions $\theta_j \sim Dirichlet(\alpha)$, Dirichlet distribution.
  - draw item latent offset $\epsilon_j \sim N(0, \lambda_v^{-1} I)$, multivariate Gauss distribution and set the item latent vector as $v_j = \epsilon_j + \theta_j$.

- For each word w_{jn}
  - $*$ draw topic assignment $z_{jn} \sim Mult(\theta)$.
  - $*$ draw word $w_{jn} \sim Mult(\beta_{z_{jn}})$.

- For each heterogeneous social graph k,
  - For each social graph specific feature m
    - $*$ draw graph factor-specific latent feature vector $s_m^k \sim N(0, \lambda_{s_{km}}^{-1} I)$.
- For each user i,
  - draw the shared latent user vector among multiple social graphs $u_i \sim N(0, \lambda_{u_i}^{-1} I)$.
  - For each heterogeneous social graph k
    - $*$ For each social graph specific feature m
      - $\cdot$ draw graph specific user heterogeneous relation pair $q_{im}^k \sim N(u_i^{\mathrm{T}} s_m^k, c_{k,q_{im}}^{-1})$.
- For each user-item pair (i, j),
  - draw the response $r_{ij} \sim N(u_i^{\mathrm{T}} v_j, c_{ij}^{-1})$.

In the above generative process, the joint likelihood of data, i.e., $R, Q_{k=1,\ldots,K}, W$, and the latent factors $U, V, S_{k=1,\ldots,K}$ under the full model is:

$$\begin{aligned} & p(R, U, V, S_{k=1,\ldots,K}, Q_{k=1,\ldots,K}, W, \theta | \lambda_\bullet) \\ & = p(R|U,V) \cdot p(W, \theta|\beta) \cdot p(U|\lambda_U) \cdot p(V|\lambda_V) \cdot \prod_k^K p(Q_k|U, S_k, \lambda_{Q_k}) \cdot \prod_k^K p(S_k|\lambda_{S_k}) \end{aligned} \tag{1}$$

For learning the parameters, we develop an EM-style algorithm similar to CTR. In our model, finding the MAP is equivalent to maximizing the following log likelihood obtained by substituting univariate and multivariate Gaussian pdfs in Eq. 1:

$$\begin{aligned} L = & \sum_j \sum_n log(\sum_z \theta_{jz} \beta_{z_{jn}}) - \sum_{k=1}^K \frac{\lambda_{S_k}}{2} \sum_m (s_m^k)^{\mathrm{T}} s_m^k - \sum_{k=1}^K \sum_i \sum_m \frac{c_{k,q_{im}}}{2} (q_{im}^k - u_i^{\mathrm{T}} s_m^k)^2 \\ & - \frac{\lambda_U}{2} \sum_i u_i^{\mathrm{T}} u_i - \frac{\lambda_V}{2} \sum_j (v_j - \theta_j)^{\mathrm{T}} (v_j - \theta_j) - \sum_i \sum_j \frac{c_{ij}}{2} (r_{ij} - u_i^{\mathrm{T}} v_j)^2 \end{aligned} \tag{2}$$

We employ coordinate ascent (CA) approach alternatively optimizing latent factor variables u_i, v_j, $s_m^{k=1,2,\ldots,K}$ and the simplex variables θ_j as topic proportions. Specifically, the following update rules in CA are obtained by setting the derivative of L with respect to u_i, v_j, and $s_m^{k=1,2,\ldots,K}$ to zero.

$$u_i = (\lambda_U I + V^{\mathrm{T}} D_{c_i} V + \sum_k^K S_k^{\mathrm{T}} D_{q_i}^k S_k)^{-1} \cdot (V^{\mathrm{T}} D_{c_i} R_i + \sum_k^K S_k^{\mathrm{T}} D_{q_i}^k Q_i^k) \tag{3}$$

$$v_j = (\lambda_V I + U^{\mathrm{T}} D_{c_j} U)^{-1} \cdot (\lambda_V \theta_j + U^{\mathrm{T}} D_{c_j} R_j) \tag{4}$$

$$s_m^{k=1,2,\ldots,K} = (\lambda_{S_k} I + U^{\mathrm{T}} D_{q_m}^k U)^{-1} \cdot (U^{\mathrm{T}} D_{q_m}^k Q_m^k) \tag{5}$$

where K is the total number of graphs. I is an identity matrix of the same dimension as that of latent space. U and V are matrices with rows as latent users and latent items, respectively. S_k is a matrix with rows as social factor-specific latent feature vectors for graph k. R_i is a column vector with values $[r_{i1}, \ldots, r_{iJ}]^{\mathrm{T}}$. Similarly, $R_j = [r_{1j}, \ldots, r_{Ij}]^{\mathrm{T}}$. For graph k, $Q_i^k = [q_{i1}^k, \ldots, q_{iM}^k]^{\mathrm{T}}$ and $Q_m^k = [q_{1m}^k, \ldots, q_{Im}^k]^{\mathrm{T}}$ respectively. Likewise, $D_{q_i}^k$, and $D_{q_m}^k$ are similarly defined with diagonal elements $c_{k,q_{i\cdot}}$ and $c_{k,q\cdot m}$, respectively. D_{c_i} is a diagonal matrix with values $diag(c_{i1}, \ldots, c_{iJ})$. $D_{c_j} = diag(c_{1j}, \ldots, c_{Ij})$. In addition, c_{ij} and $c_{k,q_{im}}$ are also seen as the confidence parameters for r_{ij} and q_{im}^k, respectively. The high confidence value a is set to the observed interactive pairs and the low confidence value b is set to the unobserved interactive pairs, where $a > b > 0$.

For our brevity, the remaining update rules for θ and β, can be obtained using the same way as described in CTR (Wang and Blei, 2011). Please see that for details.

It is worth noting that through our assumption and the derivation above, we have theoretically proved that our modeling in this case is equivalent to first concatenating features of different views together and then applying Purushotham et al. (2012) for recommendation.

3.3 CTR-MGF for Homogeneous Networks

In this section, we further extend the basic CTR to the context of homogeneous networks. In fact, any user specific homogeneous networks can be obtained through transforming corresponding heterogeneous networks. For example, in LastFM, we can construct two user-user homogeneous networks by computing the similarities of user-tag and user-artist from original heterogeneous networks. The goal of this transformation is to further exploit weak consensus modeling scheme based on Section 3.2. Different from the graph sharing mechanism presented in last section, we relax the restriction that all users have the same representation. Specifically, we assume each latent user has multiple sub-graph specific representations.

However, it is nontrivial to model the relaxed assumption directly from original perspective. To achieve this more weaker sharing mechanism, we are towards its transformed perspective, i.e., sacrificing heterogeneous characteristic, because we need to exploit shared information from latent graph specific feature perspective. Thus, we require equal dimensions of different graphs, which motivates us to investigate the homogeneous case.

The key differences between our model in this section and that in last section are the strategies of latent user modeling and its social factor modeling. More specifically, we model each latent user as a linear combination of all sub-latent users associated with multiple homogeneous networks. All these sub-latent homogeneous users are associated with a shared social factor feature space. Thus, the shared information among multiple graphs can be exploited and it is more flexible to adjust the contribution of each sub-latent user to the final latent user representation. The generative process of CTR-MGF for homogeneous networks is listed as follows:

- For each item j,
  - draw topic proportions $\theta_j \sim Dirichlet(\alpha)$, Dirichlet distribution.
  - draw item latent offset $\epsilon_j \sim N(0, \lambda_v^{-1} I)$, multivariate Gauss distribution and set the item latent vector as $v_j = \epsilon_j + \theta_j$.
  - For each word w_{jn}
    - draw topic assignment $z_{jn} \sim Mult(\theta)$.
    - draw word $w_{jn} \sim Mult(\beta_{z_{jn}})$.
- For each social graph specific feature m, regarding to all related homogeneous social graphs
  - draw a shared factor-specific latent feature vector across multiple graphs $s_m \sim N(0, \lambda_{s_m}^{-1} I)$.
- For each user i,
  - For each homogeneous social graph k
    - draw a social graph specific latent user $u_i^k \sim N(0, (\lambda_{u_i}^k)^{-1} I)$.
    - For each social graph specific feature m
      - draw graph specific user homogeneous relation pair $q_{im}^k \sim N((u_i^k)^{\mathrm{T}} s_m, c_{k,q_{im}}^{-1})$.
  - draw a final latent user $u_i \sim N(\sum_{k=1}^K T_k u_i^k, \lambda_{u_i}^{-1} I)$.
- For each user-item pair (i, j),
  - draw the response $r_{ij} \sim N(u_i^{\mathrm{T}} v_j, c_{ij}^{-1})$.

In the above generative process, the joint likelihood of data, i.e. R, $Q_{k=1,\ldots,K}$ and W, and the latent factors $U, U_{k=1,\ldots,K}, V$ and S under the full model is:

$$p(R, U, V, S, U_{k=1,\ldots,K}, Q_{k=1,\ldots,K}, W, \theta | \lambda_\bullet) = p(R|U, V) \cdot p(W, \theta|\beta) \cdot$$
$$(\prod_{k=1}^K p(Q_k|U_k, S, \lambda_{Q_k})) \cdot p(S|\lambda_S) \cdot (\prod_{k=1}^K p(U_k|\lambda_{U_k})) \cdot p(V|\lambda_V) \cdot p(U|\lambda_U) \cdot p(U|U_{k=1,\ldots,K}, \lambda_C) \tag{6}$$

Similarly to last section, we develop an EM-style algorithm to find the MAP solutions, which is equivalent to maximizing the following log likelihood by substituting univariate and multivariate Gaussian pdfs in Eq. 6:

$$
\begin{aligned}
L = & \sum_j \sum_n log(\sum_z \theta_{jz} \beta_{z_{jn}}) - \sum_{k=1}^{K} \frac{\lambda_{U_k}}{2} \sum_i (u_i^k)^{\mathrm{T}} u_i^k - \frac{\lambda_S}{2} \sum_m s_m^{\mathrm{T}} s_m \\
& - \sum_{k=1}^{K} \sum_i \sum_m \frac{c_{k,q_{im}}}{2} (q_{im}^k - (u_i^k)^{\mathrm{T}} s_m)^2 - \frac{\lambda_V}{2} \sum_j (v_j - \theta_j)^{\mathrm{T}} (v_j - \theta_j) \\
& - \sum_i \sum_j \frac{c_{ij}}{2} (r_{ij} - u_i^{\mathrm{T}} v_j)^2 - \frac{\lambda_C}{2} \sum_i (u_i - \sum_{k=1}^{K} T_k u_i^k)^{\mathrm{T}} (u_i - \sum_{k=1}^{K} T_k u_i^k) - \frac{\lambda_U}{2} \sum_i u_i^{\mathrm{T}} u_i
\end{aligned} \tag{7}
$$

We employ coordinate ascent (CA) approach as previous section alternatively optimizing latent factor variables and simplex variables as topic proportions. Then we acquire the update rules by setting the derivative of L with respect to the following variables to zero.

$$
u_i^{k=1,2,\ldots,K} = (\lambda_{U_k} I + \lambda_C T_k^2 I + S^{\mathrm{T}} D_{q_i}^k S)^{-1} \cdot (\lambda_C u_i T_k - (\sum_{p \neq k}^{K} T_p u_i^p) \lambda_C T_k + S^{\mathrm{T}} D_{q_i}^k Q_i^k) \tag{8}
$$

$$
u_i = (\lambda_U I + \lambda_C I + V^{\mathrm{T}} D_{c_i} V)^{-1} \cdot (V^{\mathrm{T}} D_{c_i} R_i + \lambda_C \sum_{k}^{K} T_k u_i^k) \tag{9}
$$

$$
v_j = (\lambda_V I + U^{\mathrm{T}} D_{c_j} U)^{-1} \cdot (U^{\mathrm{T}} D_{c_j} R_j + \lambda_V \theta_j) \tag{10}
$$

$$
s_m = (\lambda_S I + \sum_{k=1}^{K} U_k^{\mathrm{T}} D_{q_m}^k U_k)^{-1} \cdot (\sum_{k=1}^{K} U_k^{\mathrm{T}} D_{q_m}^k Q_m^k) \tag{11}
$$

where K is the total number of graphs. I is an identity matrix of the same dimension as that of latent space. U and V are matrices with rows as latent users and latent items, respectively. S is a matrix with rows as the shared social factor-specific latent feature vectors for all graphs. T is the graph selection weight, $\sum_{k=1}^{K} T_k = 1, T_k >= 0$. $R_i = [r_{i1}, \ldots, r_{iJ}]^{\mathrm{T}}$ and $R_j = [r_{1j}, \ldots, r_{Ij}]^{\mathrm{T}}$. For graph k, $Q_i^k = [q_{i1}^k, \ldots, q_{iM}^k]^{\mathrm{T}}$, $Q_m^k = [q_{1m}^k, \ldots, q_{Im}^k]^{\mathrm{T}}$ and U_k is a matrix with rows as the social graph k specific latent user vectors. Likewise, $D_{q_i}^k$, and $D_{q_m}^k$ are similarly defined with diagonal elements $c_{k,q_{i\cdot}}$ and $c_{k,q_{\cdot m}}$, respectively. $D_{c_i} = diag(c_{i1}, \ldots, c_{iJ})$ and $D_{c_j} = diag(c_{1j}, \ldots, c_{Ij})$. In addition, c_{ij} and $c_{k,q_{im}}$ are also seen as the confidence parameters for r_{ij} and q_{im}^k, respectively. The high confidence value a is set to the observed interactive pairs and the low confidence value b is set to the unobserved interactive pairs, where $a > b > 0$.

For our brevity, the remaining update rules for θ and β, can be obtained using the same way as described in CTR (Wang and Blei, 2011). Please see that for details.

3.4 Prediction

Using the learned parameters above, we can make in-matrix and out-of-matrix predictions defined in Wang and Blei (2011). For in-matrix prediction, it refers to the case where those items have been rated by at least one user in the system. To compute predicted rating, we use

$$
r_{ij}^* \approx (u_i^*)^T v_j^*. \tag{12}
$$

For out-of-matrix prediction, it refers to the case where those items have never been rated by any user in the system. To compute predicted rating, we use

$$
r_{ij}^* \approx (u_i^*)^T \theta_j^*, \tag{13}
$$

where the corresponding θ_j^* is defined as topic proportion in Section 3.2 and 3.3.

3.5 Computational Issue

To reduce computational costs when updating u_i, v_j and other variables with similar structure in update rule, we adopt the same strategy of matrix operation shown in Hu et al. (2008). Specifically, directly computing $V^{\mathrm{T}} D_{c_i} V$ and $U^{\mathrm{T}} D_{c_j} U$ requires time $O(L^2 J)$ and $O(L^2 I)$ for each user and item, where J and I are the total number of items and users respectively, L is the dimension of latent representation space. Instead, we rewrite $U^{\mathrm{T}} D_{c_j} U = U^{\mathrm{T}}(D_{c_j} - bI)U + bU^{\mathrm{T}}U$. Then, $bU^{\mathrm{T}}U$ can be pre-computed and $D_{c_j} - bI$ has only I_r non-zeros elements, where I_r refers to the number of users who rated item j and empirically $I_r \ll I$. For other similar structures, i.e., $V^{\mathrm{T}} D_{c_i} V$, $S^{\mathrm{T}} D_{q_i}^k S$, and so on, they are similar. Therefore, we can significantly speed up computation by this sparsity property.

4 Experiments

4.1 Data

We evaluate our proposed method on real life dataset [1] from LastFm. LastFm[2] is an online music catalogue, powered by social music discovery service for personalized recommendation. This dataset (Cantador et al., 2011) is challenging. Though it contains 92,834 pairs of observed ratings with 1892 users and 17,632 items, the sparseness is quite low, i.e., merely 0.2783% , which is much lower than that of the well-known Movielens dataset with the sparseness 4.25%. On average, each user has 44.21 items in the play list, ranging from 0 to 50, and each item appears in 4.95 users libraries, ranging from 0 to 611. For each item, the tag information is used as bag-of-word representation. After text processing, 11,946 distinct words are remained in the corpus. In addition, we further remove noisy users which have no items. We also construct two additional social graphs for our experiments. One is user-tag network extracted from user-tag-item relations in original dataset. The other is user-user network through transforming the constructed user-tag network. The relation in all graphs is binary, i.e., the available denoted as 1 and the unavailable denoted as 0.

Table 1: Original dataset description

Dataset	users	items	tags	user-user relations	user-tags-items	user-items relations
LastFm	1892	17632	11946	25434	186479	92834

4.2 Metrics

Two metrics for evaluating the recommendation performance are employed, i.e., Recall and NDCG. Measure for plain relevance:

$$Recall@k = \frac{\sharp relevance}{k}, \tag{14}$$

where $\sharp relevance$ denotes the total relevant papers in returned top-*k* result. Measure for ranking-based relevance:

$$NDCG@k = \frac{\sum_{i=1}^{k} \frac{2^{rel_i}-1}{log_2(1+i)}}{IDCG}, \tag{15}$$

where rel_i denotes the relevant degree which is binary in our task and $IDCG$ is the optimal score computed using the same form in numerator but with optimal ranking known in advance.

4.3 Experimental Design

In this paper, we expect the proposed model 'Our-Homo' in Section 3.2 and 'Our-Heter' in Section 3.3 can jointly provide a general and systematic solution to handling the following cases of using multiple graphs for recommendation:

- **Case 1**:(Heterogeneous networks with noise) Network data or the extraction process is usually imprecise or noisy in practice. Transform it into homogeneous case and then use 'Our-Homo'.

[1] Data available at http://grouplens.org/datasets/hetrec-2011/

[2] http://www.last.fm/

- **Case 2**:(Homogeneous networks) 'Our-Homo' can be directly employed as the tool for case 1.
- **Case 3**:(Heterogeneous networks with high quality) 'Our-Heter' might be directly employed. It is not needed to be further transformed into Homogeneous case.

The detail experiments in the following sections are presented to justify the effectiveness of our methods for the three cases above.

4.4 Experiments for Case 1 and Case 2

In this section, we mainly focus on the most complex and common case 1 with case 2 in practice.

4.4.1 Baselines

We compare our proposed two models, the model in Section 3.2 denoted as Our-Heter and the model in Section 3.3 denoted as Our-Homo, with some state-of-the-art algorithms.

- **CTR**: This method, described in Wang and Blei (2011), combines both item content information and user-item ratings for CF.
- **PMF**: This method, described in Salakhutdinov and Mnih (2007), is a well-known matrix factorization method for CF, only using interactive rating information.
- **SMF-1**: This method, described in Purushotham et al. (2012), exploits single user's social network structure combined with item's content information for CF. SMF-1 denotes using our extracted user-tag relation.
- **SMF-2**: The same SMF method, described in (Purushotham et al., 2012). SMF-2 denotes using original user-user relation.
- **Our-Heter**: Our model for heterogeneous networks, proposed in Section 3.2, uses our extracted user-tag network and original user-user network.
- **Our-Homo**: Our model for homogeneous networks, proposed in Section 3.3, uses two homogeneous networks, i.e., 1) the transformed user-user network through our extracted user-tag relation, and 2) original user-user network.

4.4.2 Settings

For a fair comparison, we use the similar settings as prior work in Purushotham et al. (2012). Specifically, to well judge the influence of multiple social network structures, we fix the effects of content information to the same level that is optimal in SMF, $\lambda_v = 0.1$. We randomly split the dataset into two parts, training (90%) and test datasets (10%), with constraint that users in test dataset have more than half of the average number of rated items, i.e., 20. This expands the range of performance analysis for our evaluation compared with Purushotham et al. (2012). The optimal parameters are obtained on a small held-out dataset. For PMF, we set $\lambda_v = 100, \lambda_u = 0.01$. For all CTR-based methods, we set $a = 1, b = 0.01, \lambda_v = 0.1$. Specifically, for CTR, we set $\lambda_u = 0.01$. For SMF-1 and SMF-2, we set $\lambda_u = 0.01$. For Our-Homo, we set $\lambda_u = 0.01, \lambda_{u1} = \lambda_{u2} = \lambda_s = 100, \lambda_c = 0.01$. For Our-Heter, we set $\lambda_u = 0.01, \lambda_{s1} = \lambda_{s2} = 100$. The remaining paramters are varied for experiment analysis.

It is noted that the task of out-of-matrix prediction is originally designed for evaluating item content modeling in CTR rather than user social graphs as in CTR-smf. Thus, we followed the same setting in baseline CTR-smf (Purushotham et al., 2012), not considering this task.

4.4.3 Performance Comparison with State-of-the-Art Methods

Figure 2 shows the recall and NDCG results of all the methods when the number of latent factor is fixed to 200 (optimal for the baselines). The proposed model 'Our-Homo' consistently outperforms the baselines and 'Our-Heter' model under both recall and NDCG measures. This finding demonstrates that (1) using multiple graphs for CTR is a necessary for improving recommendation performance from both ranking and plain accuracy perspectives. (2) strong consensus for modeling shared information undermines the performance for multiple graphs factorization as designed in Our-Heter. (3) For heterogeneous case, we address that through simply transforming the heterogeneous network to homogeneous one and then use

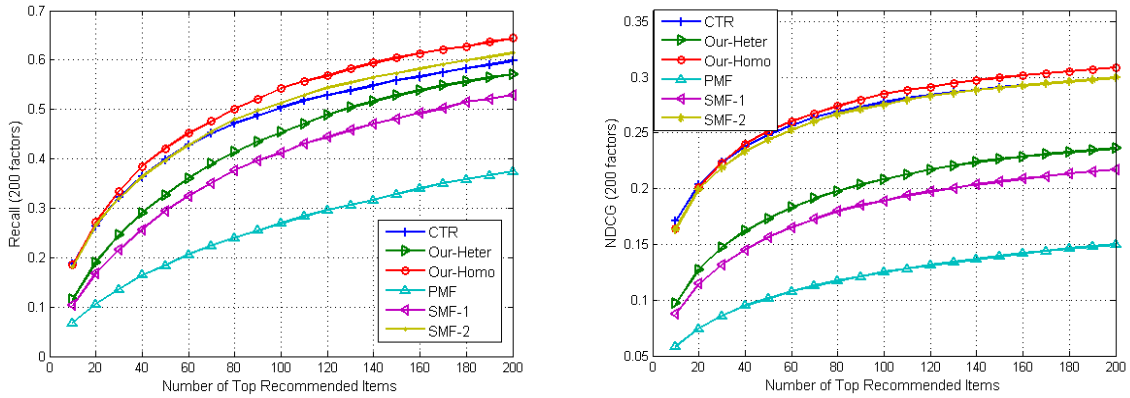


Figure 2: Our model comparison with the state-of-the-art methods, for Recall and NDCG.

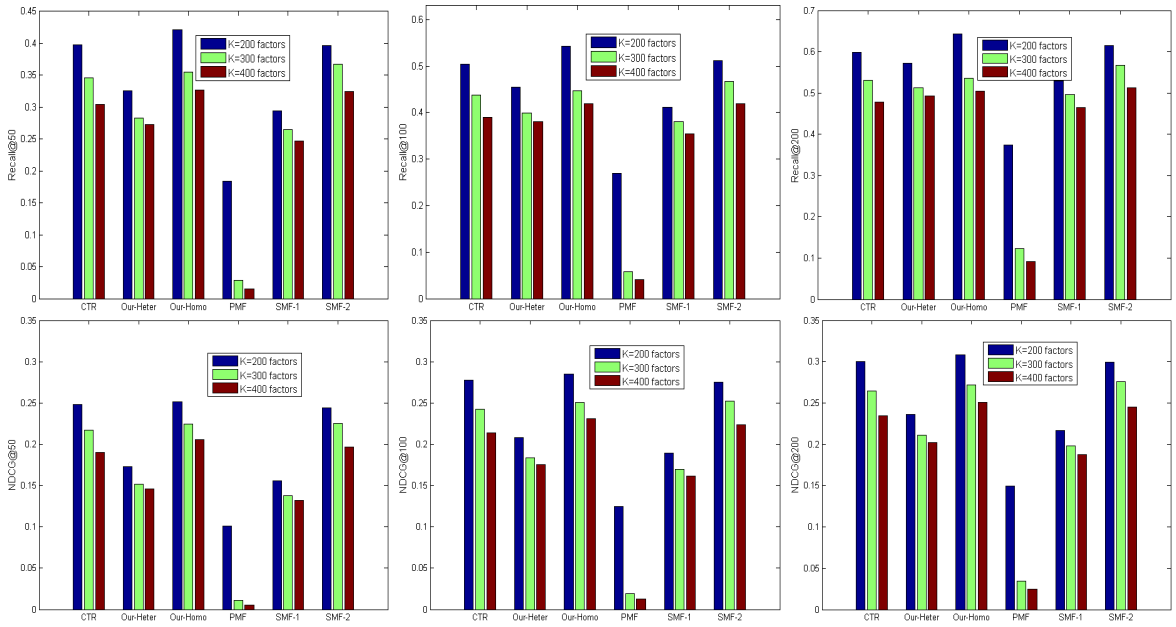


Figure 3: Performance comparison for different latent factors (K=200,300,400) @ top (50,100,200).

'Our-Homo'. This is natural but the opposite is hard. Thus, our solution 'Our-Homo' for modeling weak consensus is effective for both homogeneous and heterogeneous cases.

In addition, we can see that CTR-smf (Purushotham et al., 2012) is sensitive to the quality of graph (SMF-1 with low quality and SMF-2 with high quality as shown in Figure 2). In contrast, we can use the low quality noisy graph (SMF-1) to improve the overall performance by this transformation process. In fact, why Our-Heter does not perform well is mainly due to the noisy graph-1. 'Transformation' can be seen as a 'denoising' process.

4.4.4 Performance Analysis with Different Latent Factors

Figure 3 shows the results of the compared algorithms, with different number of latent factors for varied top recommended item. It shows that $K = 200$ factors is optimal for all baselines compared with other choices of the number of latent factors. This justifies our fire choice of 200 latent factors reported in Figure 3 (Other factor choices are omitted here due to page constraint, which is not optimal for our baselines) and suggests that the choice of latent factor number is crucial for all algorithms especially for PMF. In contrast, the proposed 'Our-Homo' is more stable compared with PMF and outperforms the other baselines in an overall performance as reported in Figure 2.

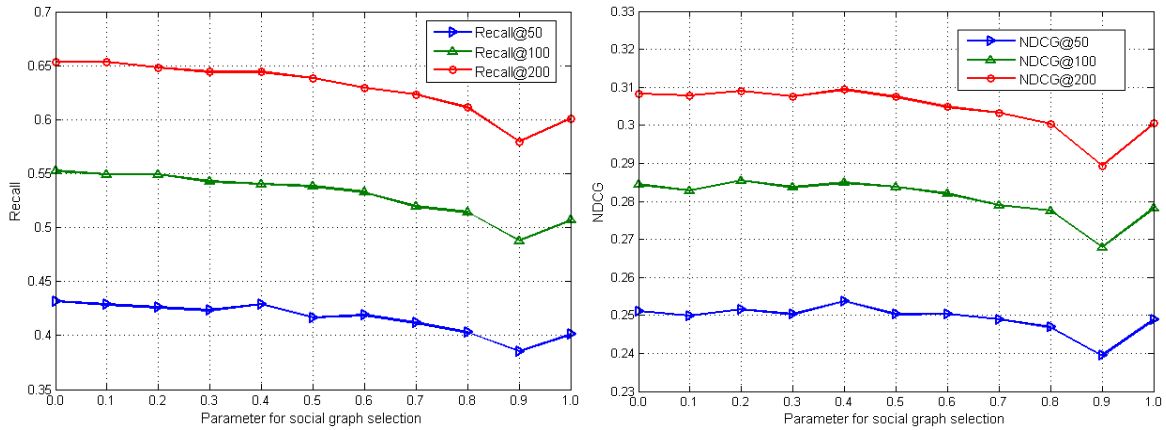


Figure 4: Parameter analysis of graph selection for our weak consensus modeling, i.e., Our-Homo.

4.4.5 Impact of the Parameter for Graph Selection

Next, we examine how our algorithm 'Our-Homo' is influenced by the graph-selection weights. In Figure 4, the horizontal axis shows the graph proportion weight T_1 for graph-1 (user-tag network). In our case, two social graphs are considered. The weight for second graph is $T_2 = 1 - T_1$, which is not shown in Figure 4. Figure 4 clearly proves the effectiveness of our weak consensus modeling. Specifically, we can see that 0.0 seemingly means that the first graph is not selected due to the weight of the first graph $T_1 = 0.0$ and that of second graph $T_2 = 1$ (fully selected). However, due to our weak consensus modeling scheme, though the graph-1 weight is 0.0, it does not mean that the first graph is removed. In fact, the effect of graph-1 is also active through the shared variable 's' in Figure 1 (right). This further can be investigated from the Equation 9. Apparently, the case of 0.0 weight for graph-1, u is only relevant to u_1, but from Equation 8, we can see that u_1 is also influenced by u_2 via the shared social graph factors s. Thus, this also explains why 0.0 weight for graph-1 is not equal to the result of SMF-2 as shown in Figure 2, which only uses graph-2, original user-user graph in SMF-2.

In addition, the 'valley' in Figure 4 might be explained that in the extreme cases (0.0 and 1.0), the denoising effect of weak consensus is slightly strengthened because only one specific graph (higher quality smf-2 or lower quality smf-1 shown in Fig.2) is directly associated with final latent user combined with shared variable. Therefore, the extreme case (1.0) is towards a relative higher performance.

4.5 Experiments for Case 3

Though the proposed 'Our-Homo' is more effective than CTR and CTR-smf, it does not mean that the proposed another method 'Our-Heter' is useless. In this section, we show how the case 3 will be justified.

4.5.1 Baselines

- **Our-Heter(N)**: Our model for heterogeneous networks, proposed in Section 3.2, uses our modified high quality user-tag network as described in Section 4.5.2 and original user-user social network as SMF-2 in Section 4.4.1.
- **SMF-1(N)**: The same SMF method with single social network, described in Purushotham et al. (2012). SMF-1 (N) denotes using our modified high quality user-tag network as described in Section 4.5.2.
- **Our-Homo(O)**: The result of this method is the same as that reported in Section 4.4.3.
- **CTR(O)**: The result of this method (Wang and Blei, 2011) is the same as that reported in Section 4.4.3.

4.5.2 Settings

We want to investigate whether Our-Heter will outperform Our-Homo in the case where the heterogeneous networks with less noise are available, compared with previous results in Figure 2. The settings

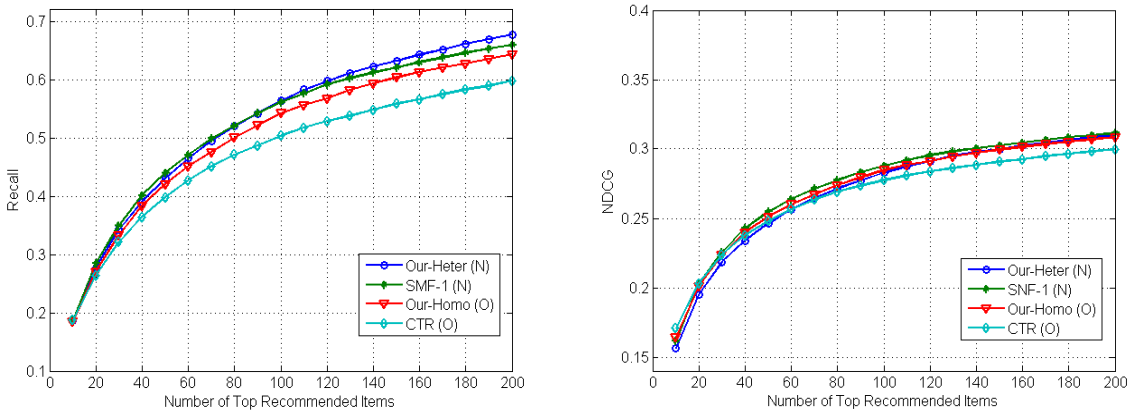


Figure 5: Our model comparison with the state-of-the-art methods (case 3), for Recall and NDCG.

in case 3 are the same as that in Section 4.4.2 except for the refined user-tag graph. In this case, we construct a less noisy user-tag network by selecting top-10% tags according to $tf * idf$ value. The optimal latent factor number is set to 100 for Our-Heter (N) through a small held-out dataset. The remaining parameters are kept as the same values in Section 4.4.2. For notations in baselines Section, 'O' denotes old setting and 'N' denotes new setting with updated user-tag network presented in this section.

4.5.3 Performance Comparison with State-of-the-Art Methods

Figure 5 shows Our-Heter (N) can achieve improved performance compared with baselines without transformation, for the case where high quality graphs are available. Specifically, for recall measure, Our-Heter (N) produces the best result with the increasing number of top recommended items. In addition, we observe that modeling multiple graphs is necessary to further improve recommending performance, while multiple high quality heterogeneous graphs are available.

For NDCG measure, Our-Heter (N) is comparable to our baselines. Since recall measure is only considered for several reasons in previous work (Wang and Blei, 2011; Purushotham et al., 2012), NDCG is introduced as a plus compared with primarily focused recall. Therefore, Our-Heter (N) is also competitive in overall performance in case 3.

In fact, as discussed in Kang and Lerman (2013), CTR-smf (Purushotham et al., 2012) is not always superior to CTR (Wang and Blei, 2011) and vice versa due to different contexts. Likewise, our model is under the multi-view assumption as discussed in Section 1 that should be checked in practice.

5 Conclusions

In this paper, we propose a general recommendation framework with multiple data sources based on CTR. It is a principled hierarchy Bayesian framework with multiple social graphs factorization for recommender systems. In this framework, two ways of consensus modeling are exploited. Specifically, the proposed models Our-Homo and Our-Heter can jointly provide a general and systematic solution to handling three real cases of using multiple graphs with item content information for recommendation: case 1) Heterogeneous networks with noise; case 2) Homogeneous networks; case 3) Heterogeneous networks with high quality. Experimental results on real dataset demonstrate the effectiveness of our approach. While this framework is used for modeling multiple user social graphs, it can be easily extended to exploiting other side information such as multiple complex relations for items in various applications.

Acknowledgements

We thank the anonymous reviewers for their valuable suggestions. This research was partly supported by National Natural Science Foundation of China (No.61370117,61333018), Major National Social Science Fund of China (No.12&ZD227), and National High Technology Research and Development Program of China (863 Program) (No.2012AA011101).

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2011. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM Conference on Recommender Systems*, RecSys 2011, New York, NY, USA. ACM.

Xuetao Ding, Xiaoming Jin, Yujia Li, and Lianghao Li. 2013. Celebrity recommendation with collaborative social topic regression. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI*, pages 2612–2618. AAAI Press.

Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Eighth IEEE International Conference on Data Mining, ICDM*, pages 263–272. IEEE.

Hongzhao Huang, Arkaitz Zubiaga, Heng Ji, Hongbo Deng, Dong Wang, Hieu Khac Le, Tarek F Abdelzaher, Jiawei Han, Alice Leung, John Hancock, et al. 2012. Tweet ranking based on heterogeneous networks. In *Proceedings of International Conference on Computational Linguistics, COLING*, pages 1239–1256.

Jeon-Hyung Kang and Kristina Lerman. 2013. La-ctr: A limited attention collaborative topic regression for social media. In *Proceedings of AAAI Conference on Artificial Intelligence, AAAI*, pages 119–125.

Abhishek Kumar, Piyush Rai, and Hal Daumé III. 2011. Co-regularized multi-view spectral clustering. In *Advances in Neural Information Processing Systems, NIPS*, pages 1413–1421.

Yingming Li, Ming Yang, and Zhongfei Mark Zhang. 2013. Scientific articles recommendation. In *Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, CIKM*, pages 1147–1156. ACM.

Jialu Liu, Chi Wang, Jing Gao, and Jiawei Han. 2013. Multi-view clustering via joint nonnegative matrix factorization. In *Proceedings of SIAM Data Mining Conference, SDM*. SIAM.

Sanjay Purushotham, Yan Liu, and C-c J Kuo. 2012. Collaborative topic regression with social matrix factorization for recommendation systems. In *Proceedings of the 29th International Conference on Machine Learning, ICML*, pages 759–766.

Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems, NIPS*, pages 1257–1264.

Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 448–456. ACM.

Hao Wang, Binyi Chen, and Wu-Jun Li. 2013. Collaborative topic regression with social regularization for tag recommendation. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI*, pages 2719–2725. AAAI Press.

High Performance Word Sense Alignment by Joint Modeling of Sense Distance and Gloss Similarity

Michael Matuschek [‡] **and Iryna Gurevych** [†‡]
† Ubiquitous Knowledge Processing Lab (UKP-DIPF),
German Institute for Educational Research and Educational Information
Schloßstr. 29, 60486 Frankfurt, Germany
[‡]Ubiquitous Knowledge Processing Lab (UKP-TUDA),
Department of Computer Science, Technische Universität Darmstadt
Hochschulstr. 10, 64289 Darmstadt, Germany
http://www.ukp.tu-darmstadt.de

Abstract

In this paper, we present a machine learning approach for word sense alignment (WSA) which combines distances between senses in the graph representations of lexical-semantic resources with gloss similarities. In this way, we significantly outperform the state of the art on each of the four datasets we consider. Moreover, we present two novel datasets for WSA between Wiktionary and Wikipedia in English and German. The latter dataset in not only of unprecedented size, but also created by the large community of Wiktionary editors instead of expert annotators, making it an interesting subject of study in its own right as the first crowdsourced WSA dataset. We will make both datasets freely available along with our computed alignments.

1 Introduction

Lexical-semantic resources (LSRs) are an important foundation for numerous natural language processing (NLP) tasks such as word sense disambiguation (WSD) or information extraction (IE). However, large-scale LSRs are only available for a few languages. The Princeton WordNet (Fellbaum, 1998) is commonly used for English, but for most languages such resources are small or missing altogether. Another problem is that, even for English, there is no single LSR which is suitable for all different application scenarios, because the resources contain different words, senses or even information types. Recently, it has been argued that collaboratively constructed resources (e.g. Wiktionary (Meyer and Gurevych, 2012))) are a viable alternative, especially for smaller languages (Matuschek et al., 2013), but there are still considerable drawbacks in coverage which make their usage challenging.

These observations have led to the insight that word sense alignment (WSA), i.e. linking at the level of word senses, is key for the efficient exploitation of LSRs, and it was shown that the usage of linked resources can indeed yield performance improvements. Examples include WSD using aligned WordNet and Wikipedia (Navigli and Ponzetto, 2012a), semantic role labeling using PropBank, VerbNet and FrameNet (Palmer, 2009), the construction of a semantic parser using FrameNet, WordNet, and VerbNet (Shi and Mihalcea, 2005) and IE using WordNet and Wikipedia (Moro et al., 2013). Cholakov et al. (2014) address the special task of verb sense disambiguation. They use the large-scale resource UBY (Gurevych et al., 2012) which contains nine resources in two languages, mapped to a uniform representation using the LMF standard for interoperability (Eckle-Kohler et al., 2012), and also (among others) sense alignments between WordNet, FrameNet, VerbNet and Wiktionary which are exploited in their approach.

However, WSA is challenging because of word ambiguities, different sense granularities and information types (Navigli, 2006), so that past efforts mostly focused on specific resources or applications, where expert-built resources such as WordNet played a central role in most cases. Approaches which aim at being more generic (i.e. applicable to a wider range of LSRs) usually focused on only one information source for the alignment (e.g. glosses or graph structures) without combining them in an elaborate way.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 245–256, Dublin, Ireland, August 23-29 2014.

In this paper, we want to go beyond this previous work in two ways: i) For the first time, we present an alignment between the large-scale collaboratively constructed resources Wiktionary and Wikipedia. While both LSRs have been extensively used in NLP and especially WSA (see Section 2), no attempt has been made to combine them, although Wiktionary was explicitly designed to complement the encyclopedic knowledge in Wikipedia with linguistic knowledge. Apart from already established tasks like WSD, the strong multilingual focus of both resources makes their combination especially promising for applications such as knowledge-based machine translation or computer-assisted translation where additional background knowledge and translation options can be crucial (Matuschek et al., 2013). To fill this gap in the body of research, we present two new evaluation datasets for English and German, where the latter is not only of remarkable size, but also directly extracted from Wiktionary in a novel approach, making it the first crowdsourced WSA dataset. ii) Also for the first time, we jointly model different aspects of sense similarity by applying machine learning techniques to WSA. However, unlike previous approaches, we do not engineer our features towards a specific resource pair, rendering the approach powerful but proprietary. Instead, we aim to combine generic features which are applicable to a variety of resources, and we show that combining them leads to state-of-the-art WSA performance. In particular, we employ distances calculated with Dijkstra-WSA (Matuschek and Gurevych, 2013), an algorithm which works on graph representations of resources, as well as gloss similarity values. This lets us take advantage of both (orthogonal) ways of identifying equivalent senses and yields a very robust and flexible WSA framework.

The rest of this paper is structured as follows: In Section 2 we discuss related work, in Section 3 we describe our approach and introduce the resources and datasets we use in our experiments, in Section 4 we evaluate our results, and we conclude in Section 5 with some directions for future work.

2 Related Work

There are two main approaches to WSA which have been applied: Similarity-based and graph-based ones. To our knowledge, there exists no previous work which effectively combines both approaches in a unified framework, and only few works which combine both kinds of features for different purposes.

2.1 Similarity-based Approaches

WordNet was aligned to Wikipedia (Niemann and Gurevych, 2011) and Wiktionary (Meyer and Gurevych, 2011) using a framework based on gloss similarity, in spirit of the earliest work in WSD presented by Lesk (1986). In both cases, cosine and personalized PageRank (PPR) similarity (Agirre and Soroa, 2009) were calculated, and a simple machine learning approach was used to classify each pair of senses (see Section 3.3). This idea was also applied to cross-lingual alignment between WordNet and the German part of OmegaWiki (Gurevych et al., 2012), using machine translation as an intermediate component. Henrich et al. (2011) use a similar approach for aligning GermaNet and Wiktionary, but with word overlap as the similarity measure. De Melo and Weikum (2010) report an alignment of WordNet synsets to Wikipedia articles which is also based on word overlap. We later report results based on gloss similarity as one of our baselines (Tables 2 and 3).

2.2 Graph-based Approaches

In one of the earliest structure-based works, Daudé et al. (2003) map different versions of WordNet based on the synset hierarchy. Navigli (2009) disambiguates WordNet glosses, i.e. sense markers are assigned to all non-stopwords in each WordNet gloss. The approach is based on finding circles in the WordNet relation graph to identify disambiguations. In later work, this idea was applied to the disambiguation of translations in a bilingual dictionary (Flati and Navigli, 2012). While this "alignment" of dictionary entries is related to our problem, it was not discussed how this idea could be applied to word sense alignment of two resources. Laparra et al. (2010) use a shortest path algorithm (SSI-Dijkstra+) to align FrameNet lexical units (LUs) with WordNet synsets. They align monosemous LUs first and then search for the closest synset in WordNet for the other LUs in the same frame. The LUs are, however, considered as mere texts to be disambiguated; there is no attempt made to exploit the graph structure of FrameNet.

Ponzetto and Navigli (2009) use a graph-based method for aligning WordNet synsets and Wikipedia categories. Using semantic relations, they build subgraphs of WordNet for each category and then align senses to categories based on the structural features. In our own previous work, we presented Dijkstra-WSA, a graph-based approach working with shortest paths (Matuschek and Gurevych, 2013). It achieves state-of-the-art precision, but recall is an issue if the graphs are sparse (i.e. in case of only few semantic relations). As Dijkstra-WSA distances are one of the features we use for our machine learning approach, we will present this approach in more detail in section 3.2.2 and also report results for Dijkstra-WSA on our evaluation datasets for comparison.

2.3 Hybrid Approaches

In later work, Navigli and Ponzetto (2012a) also align WordNet with the full Wikipedia. Besides using bag-of-words overlap to compute gloss similarity, they also build a graph structure for the senses in both resources by using WordNet semantic relations. The goal is to determine which WordNet sense is closest to the Wikipedia sense to be aligned. However, the graph structure of Wikipedia is disregarded, as is the global structure of WordNet, as just a locally restricted subset of WordNet relations is used. In the same context of BabelNet, Navigli and Ponzetto (2012b) also present BabelRelate, an approach which relies on translations to compute cross-lingual semantic similarity; however, they do not apply it to WSA. Dijkstra-WSA was enhanced by using a backoff, by means of performing a graph-based alignment first, and in cases where no alignment target sense can be found, a decision is made based on the similarity of glosses (Matuschek and Gurevych, 2013). While this simple two-step approach increases recall substantially, it comes at the expense of lower precision. However, the overall F-measure achieved state-of-the-art performance on every considered dataset (0.65–0.87). We also report the results for this hybrid approach as a baseline (Tables 2 and 3). De Melo and Weikum (2008) use a machine learning approach with a combination of structural and content-based features of WordNet, but for building new wordnets in other languages, not aligning existing ones.

In summary, the different approaches to compute similarity have mostly been used in isolation, or combined in a shallow or restricted way. More complex approaches usually require resource-specific feature engineering, which makes their transferability to other resources or languages difficult. Thus, we present a framework which combines different similarity measures in a generic and flexible way and enables state-of-the-art WSA performance on a variety of resources with modest effort.

3 The Alignment Procedure

The basic steps of our alignment algorithm are:

1. For each sense in one resource, all possible candidates in the other resource are retrieved. Candidates are senses which have the same attached lemma and part of speech. For instance, for the *programming* sense of *Java* in one resource, their might exist senses for *programming*, *island* or *coffee* in the other one which are all possible alignment targets.

2. For each candidate pair, we calculate a set of features describing their similarity in different ways.

3. For a set of word senses (the gold standard), the alignment decision is made by human annotators.

4. A machine learning classifier is trained on this gold standard, and an alignment decision is made for the remainder of the candidate pairs to produce a complete alignment of the resources. In our setup, we use 10-fold cross validation to train the classifier.

The different datasets and steps of the algorithm are explained in more detail in the following sections.

3.1 Resources and Datasets

We use four different WSA evaluation datasets, two of which are presented for the first time. To ensure compatibility with previous work, we use the same versions of the resources as reported in (Gurevych et al., 2012) and (Matuschek and Gurevych, 2013).

Pair	Pos.	Neg.	Polysemy	One cand.	F_1	A_0	Composition
WordNet-OmegaWiki	210	473	1.50	75.2%	0.84	0.85	random
WordNet-Wiktionary	313	2 110	4.76	18.6%	0.78	0.93	manual
Wiktionary-Wikipedia (En)	75	292	1.27	87.6%	0.79	0.95	automatic
Wiktionary-Wikipedia (De)	21 855	9 953	1.47	77.6%	0.85	0.89	crowd

Table 1: Characteristics of the gold standards used in the evaluation. The degree of polysemy (i.e. the number of possible alignment targets per sense) hints towards the difficulty of the task, as does the number of senses with only one alignment candidate. WordNet-Wiktionary stands out as it was manually composed and is not representative of the full alignment (Meyer and Gurevych, 2011). The inter-annotator agreements A_0 and F_1 can be considered as upper bounds for automatic alignment accuracy and F-measure. Note that for the Wiktionary-Wikipedia datasets, due to the nature of their creation, the agreement was originally not available; we estimated it by manually re-annotating a sample of 100 examples with two annotators.

3.1.1 Resources

WordNet (Fellbaum, 1998) is a computational lexicon for English created at Princeton University. It is organized in sets of synonyms (synsets), each expressing a distinct concept. Synsets are represented by textual definitions (so-called glosses). A hierarchical organization is encoded via semantic relations such as hyponymy.

Wikipedia is a collaboratively created online encyclopedia available in almost 300 languages. The current English version contains around 4 400 000 articles, and the German one around 1 700 000 articles, each usually describing a particular concept. Due to its encyclopedic nature, Wikipedia mostly covers nouns, while the other LSRs discussed also cover verbs, adjectives, etc. Articles are connected via hyperlinks in the article text (implying a graph structure), and the first paragraph usually gives a short summary of the topic, serving as a gloss for our purposes. Articles are also linked to the equivalent articles in other languages.

Wiktionary is a dictionary "side project" of Wikipedia, available in over 500 languages. Currently, the English Wiktionary contains over 500 000 lexical entry pages, while the German one contains around 350 000 ones. For a word, multiple senses can be encoded, and these are usually represented by glosses. Wiktionary also contains hyperlinks to synonyms, hypernyms, etc. and translations into other languages.

OmegaWiki is a freely editable online dictionary like Wiktionary. However, instead of distinct language editions, OmegaWiki contains language-independent concepts ("Defined Meanings") which carry lexicalizations in different languages. These concepts are connected via semantic relations. OmegaWiki contains over 46 000 concepts and lexicalizations in almost 500 languages.

3.1.2 Datasets

WordNet–OmegaWiki: The first alignment between these LSRs based on the German part of OmegaWiki was reported in (Gurevych et al., 2012). As OmegaWiki Defined Meanings are multilingual, we used the same dataset for monolingual WSA in later work (Matuschek and Gurevych, 2013). Table 1 presents details about this and the other evaluation datasets.

WordNet–Wiktionary: Meyer and Gurevych (2011) originally used this dataset for similarity-based alignment. While we could not improve upon this using Dijkstra-WSA on its own (Matuschek and Gurevych, 2013), the backoff approach yielded a significant improvement. This dataset was manually composed according to specific criteria, hence it differs from the others and is not fully representative of the full alignment.

Wiktionary–Wikipedia (English): No evaluation dataset (let alone a full alignment) has been reported for this resource pair yet. However, as the datasets for WordNet-Wiktionary (Meyer and Gurevych, 2011) and WordNet-Wikipedia (Niemann and Gurevych, 2011) are lexically overlapping, we were able to automatically create a gold standard for Wiktionary-Wikipedia by exploiting the transitivity of the alignment relation, i.e. by using WordNet as a pivot. Note that, unlike Wiktionary, Word-

Net synsets have multiple lexicalizations for the same meaning, introducing alignment candidates from Wikipedia which might not be applicable to a particular Wiktionary sense. Hence, we decided to filter the examples where the lexeme of the Wiktionary sense and the Wikipedia article title did not match. An effect of this process was that words not contained in all three resources were filtered out, and many examples were left with few or only one candidate, leading to a low polysemy. We also manually checked the derived gold standard and corrected a small number of wrong annotations introduced through the automatic process. The resulting dataset is thus considerably smaller than the others, but it still turned out to be sufficient for machine learning experiments.

Wiktionary–Wikipedia (German): Same as for the English editions, neither a gold standard nor an alignment was previously reported for this pair. We were able to create a gold standard in a novel way by exploiting the fact that many German Wiktionary senses contain links to the corresponding Wikipedia articles, inducing a sense alignment between the two LSRs manually validated by the Wiktionary community. However, we were unable to extract such an alignment for English, as Wikipedia articles are attached to the lexical entry page in this version and not to a specific sense.

In the German Wiktionary, a large portion of the senses is linked in this way, and even after aggressively filtering out invalid link targets (e.g. disambiguation pages or pages with a non-matching title), we retained over 20 000 alignments between Wiktionary senses and Wikipedia pages, a sample of which we manually confirmed to be correct. Of course, this only yields positive examples; to also include cases of non-alignment, we extracted the other candidate (i.e. lexically matching) Wikipedia articles for each aligned Wiktionary sense, assuming that Wiktionary editors also considered and discarded them before eventually creating a link. Interestingly, the number of negative examples derived in this way is relatively low in comparison to the other datasets. An analysis revealed that a large fraction of the linked Wiktionary senses are either scientific terms (e.g. from biology) or named entities such as cities. Both types of senses tend to have few alternative candidates in Wikipedia due to their specificity, and it seems logical that Wiktionary users predominantly link these senses to the explanatory Wikipedia articles which are not familiar to the majority of users.

In the end, this process yielded a WSA dataset with unprecedented characteristics: It was not only created and validated by a crowd of editors rather than a handful of annotators, but it is also an order of magnitude larger than previously reported datasets (Table 1). This enables us to assess the performance of our WSA approach in a scenario which is close in size to a full alignment task, allowing a more well-grounded statement about its effectiveness.

3.2 Feature Engineering

The selection of features for our machine learning approach was driven by the premise to keep the framework as generic and resource-agnostic as possible, in order to ensure applicability to many different LSRs without additional engineering effort. Thorough analysis of existing resources and approaches revealed that two types of information are available for the vast majority of LSRs: i) Glosses, or more general, textual descriptions of concepts, and ii) Relationships between concepts inducing a graph, given through semantic relations, links, or other means. We also evaluated some features which are specific to a smaller subset of resources (see Section 3.2.3).

3.2.1 Gloss Similarity

Cosine similarity (COS) calculates the cosine of the angle between a vector representation of two senses s_1 and s_2. For the vector representation of a sense, we use a bag-of-words approach, i.e., a vector $\mathrm{BoW}(s)$ contains the term frequencies of all words in the description of s. In this work, we only rely on the textual definition of a sense to keep the approach as generic as possible, while the usage of example sentences, related words, synonyms etc. would also be possible.

Personalized PageRank similarity (PPR) (Agirre and Soroa, 2009) measures the semantic relatedness between two word senses s_1 and s_2 by comparing semantic vectors which can be derived in different ways; we utilize the variant introduced by Niemann and Gurevych (2011). The idea is to identify senses of words in a sense's gloss which are central for describing its meaning. These senses (represented in a graph derived from an LSR such as WordNet) should have a high PageRank score (i.e. a high centrality).

3.2.2 Dijkstra-WSA Distance

Dijkstra-WSA (Matuschek and Gurevych, 2013) is the graph-based WSA algorithm we use to calculate a distance-based similarity measure between word senses. We will briefly explain its two steps.

Graph construction: The *resource graph* is comprised of a set of nodes V which represents the senses of an LSR and a set of edges $E \subseteq V \times V$ which expresses semantic relatedness between them. One can use semantic relations, hyperlinks, or other relatedness indicators. For sparse LSRs, it is advisable to add edges between senses s_1 and s_2 if a monosemous term t with sense s_2 is included in the gloss of s_1. For example, one can link a sense of *Java* to *programming language* if the latter term is included in the former's definition text. This *monosemous linking* enhances the graph density (and hence, the recall) significantly.

Computing sense alignments: First, trivial alignments between the two resource graphs A and B are created. Alignments are trivial if two senses have the same attached lexeme in A and B and this lexeme is also unique in either resource. Intuitively, these alignments serve as "bridges" between highly related regions of A and B. Next, for each remaining sense $s \in A$, the set of possible target senses $T \subset B$ is retrieved in a similar fashion as for our approach, and for each of them the shortest path is computed using Dijkstra's algorithm (Dijkstra, 1959). While Dijkstra-WSA then goes on to directly align the sense which is closest to the source sense, we save the distance for each candidate sense and directly use it as a feature, expressing semantic relatedness based on the structure of both underlying resources. When no distance can be computed (in case of a disconnected graph), we assume infinite distance.

3.2.3 Other Features

We also experimented with other features which were accessible directly from the resources, i.e. without the need for external knowledge or extensive computational effort; these were usually not available for every resource pair. Features we tried were the part of speech (Wiktionary, OmegaWiki, WordNet), the sense index, i.e. the position in the sense list for a lexeme (WordNet, Wiktionary), similarity of example sentences (WordNet, Wiktionary), overlap of translations into other languages (Wikipedia, Wiktionary, cf. (Bond and Foster, 2013)) and overlap of domain labels (Wikipedia, Wiktionary, WordNet, OmegaWiki). However, for none of these features we could observe any significant[1] impact on the results, mostly due to sparsity of the respective features. Thus, we do not report them, but on the other hand we consider this an indicator that gloss similarity and distance in the resource graph already sufficiently capture the similarity between senses.

3.3 Machine Learning Classifiers

We experimented with different machine learning classifiers using WEKA (Hall et al., 2009). While a detailed discussion of these classifiers is beyond the scope of this work, we will at least give a short description of the ones we eventually used. For more details, please refer to textbooks such as (Murphy, 2012). We used WEKA's standard configuration in every case.

Threshold-based classifiers work by simply trying to learn a numeric boundary value which separates positive examples from negative ones. Although this approach is rather naive, it has been successfully used in previous WSA efforts (Meyer and Gurevych, 2011; Niemann and Gurevych, 2011).

A **Naive Bayes** classifier assumes that features are independent (i.e. the value of one feature is unrelated to any other feature), and is thus able to learn reliable classification probabilities on relatively small training sets. While the independence assumption can be considered an oversimplification, the algorithm is widely used due to its efficiency and good precision.

Bayesian Networks (or *belief networks*) also classify based on probabilities learned from training data, however, they offer the advantage of modeling dependencies between features, hence allowing a more accurate representation of the data. Technically, such a network is a directed acyclic graph modeling the conditional dependencies between variables.

A **Perceptron** is a classifier which maps a real-valued input vector to a binary output, by means of an artificial neural network. It is commonly used for pattern recognition, also in NLP (Collins, 2002).

[1] All significance claims in this paper are based on McNemar's test at a confidence level of 1%.

Support Vector Machines (SVMs) construct a hyperplane in a multi-dimensional space which yields a good separation between positive and negative training examples, represented as data points.

Decision Trees are built from training input by iteratively splitting the set of samples based on attribute values so that the resulting subset is as homogeneous as possible with regard to the class label. Unseen examples can be classified by testing the attribute values and following different branches of the tree. One of the main advantages (e.g. in comparison to SVMs) is that this approach is easily interpretable.

4 Experimental Results and Analysis

Baselines For reference, we report six different baselines: i) *Random*: A random sense from the set of candidates is chosen in each case, ii) *1:1*: An alignment is always made if and only if there is exactly one candidate, iii) *1st*: The first of the candidate senses is always selected[2], iv) *SIM*: A similarity threshold is learned for gloss similarity values as suggested by Meyer and Gurevych (2011), cf. Section 3.2.1, v) *DWSA*: The closest candidate sense in the resource graph is aligned as we suggested in (Matuschek and Gurevych, 2013), cf. Section 3.2.2, vi) *HYB*: A hybrid approach of using *DWSA* first and then *SIM* as a backoff, also suggested by us (Matuschek and Gurevych, 2013). The latter approach represents state-of-the-art performance for WSA. Note that for the two Wiktionary-Wikipedia datasets, no previous results were available, so we created similarity-based and Dijkstra-WSA alignments ourselves, based on the same versions of the resources as in the previous work. For the other datasets, we used the numbers reported in the original papers (Matuschek and Gurevych, 2013; Meyer and Gurevych, 2011).

Overview Tables 2 and 3 present the results for all setups. Although the best classifiers for each dataset always outperform the previous state of the art and the baselines by a significant margin, there is no consistent pattern in the results across different LSRs and classifiers. One reason for this is that the range of feature values varies substantially between different datasets. For instance, Dijkstra-WSA distances tend to be greater when Wikipedia is involved simply by its virtue of being larger than the other LSRs, and gloss similarities also differ depending on the average length of the glosses and the language. Another factor are the gold standards, which are quite different in terms of size and composition (see Table 1). Thus, no classifier is the undisputed "winner", but Bayesian Networks proved most robust in our experiments, showing competitive results in every case. As training them is also computationally cheap (compared to SVMs, for instance), we would generally recommend this kind of classifier for WSA tasks. In the following, we also provide a more detailed discussion of the results for each individual dataset.

WordNet-OmegaWiki In this case, the precision of the alignment is satisfactory for every classifier, while both previously reported approaches struggle for different reasons (Gurevych et al., 2012; Matuschek and Gurevych, 2013). The strength of the machine learning becomes apparent especially in comparison with the *HYB* approach: While the latter merely combines independent alignment decisions, hence achieving better recall but failing to improve precision (cf. Section 2.3), the joint usage of features leads to a massive improvement. Analysis of the decision tree classifier shows that, as we suspected, the "edge cases" are explicitly reflected in the learned model, i.e. examples with high gloss similarity but also a high Dijkstra-WSA distance (or vice versa) are ruled out with higher confidence. This observation generally also holds for the other datasets. As an example, the two senses of *genome* in biology (*"The non-redundant genetic information stored in DNA sequences that defines an individual organism"*) and algorithmics (*"In the context of a genetic algorithm, the information that defines an individual entity"*) have similar glosses; they are, however, quite far apart in the graph and thus not aligned. The Bayesian Network achieves the best results as it comprehensively models this interdependence of features. The SVM achieves the best precision, but the distribution of feature values does not lend itself well to linear separation in this case, leading to unsatisfactory recall.

WordNet-Wiktionary For this dataset, the results look similar to WordNet-OmegaWiki as far as the improvement of precision is concerned, as the joint usage of features helps to make a correct decision on

[2]While this corresponds to the most frequent sense baseline in other setups, note that no explicit frequency information is available for OmegaWiki, Wiktionary and Wikipedia, so that the first sense baseline is only a rough approximation.

	WordNet-OmegaWiki				WordNet-Wiktionary			
	P	R	F_1	A	P	R	F_1	A
Random	0.46	0.35	0.40	0.51	0.21	0.59	0.31	0.67
1:1	0.36	0.64	0.46	0.55	0.68	0.19	0.30	0.88
1st	0.34	0.80	0.48	0.47	0.33	0.51	0.40	0.80
SIM	0.55	0.53	0.54	0.73	0.67	0.65	0.66	0.91
DWSA	0.56	0.69	0.62	0.74	0.68	0.27	0.39	0.89
HYB	0.57	**0.75**	0.65	0.75	0.68	0.71	0.69	0.92
SVM	**0.95**	0.32	0.48	0.79	**0.82**	0.61	0.70	0.93
Naive Bayes	0.73	0.62	0.67	0.82	0.71	0.79	0.75	0.92
Bayesian Network	0.75	0.72	**0.74**	**0.84**	0.70	**0.84**	**0.77**	**0.94**
Perceptron	0.73	0.58	0.65	0.81	0.74	0.72	0.73	0.92
Decision Tree	0.68	0.63	0.66	0.80	0.78	0.66	0.72	0.93
Agreement	-	-	0.84	0.85	-	-	0.78	0.93

Table 2: Alignment results for WordNet-OmegaWiki and WordNet-Wiktionary: Using baselines (top), approaches from previous work (middle) and different machine learning classifiers (bottom). We report precision, recall, F-measure (the harmonic mean of both) and accuracy. Best results for each value and dataset are marked in bold. The inter-annotator agreements A_0 and F_1 are given as upper bounds.

borderline examples. However, in this case the recall is also substantially improved, especially for the Bayesian classifiers. This was an issue in the original Dijkstra-WSA results (Matuschek and Gurevych, 2013) due to the low connectivity of the English Wiktionary graph. The combination of distances and gloss similarities is able to alleviate this shortcoming of Wiktionary to some extent, as examples with missing Dijkstra-WSA distance can still be aligned in case of sufficient gloss similarity. SVMs also show the best precision here, but are challenged by the suboptimal separability of the feature space.

Wiktionary-Wikipedia (English) The low connectivity of Wiktionary is not as much an issue here as for WordNet-Wiktionary, mostly due to the different composition of the gold standard – higher-frequency words tended to be retained (see Section 3.1.2), which in turn are better connected within Wiktionary. This leads to reasonable results for Dijkstra-WSA alone. The hybrid approach reaches the best recall, but due to the relatively low precision of the *SIM* alignment, the overall result leaves room for improvement. This improvement is again achieved via joint modeling of features. As for the datasets discussed above, the precision is improved significantly; this is especially true for the Bayesian Network classifier. Precision and recall for the SVM classifier are also satisfactory in this case (due to the better linear separability of the feature space), making it the best overall classifier along with the Perceptron.

Wiktionary-Wikipedia (German) On this dataset, the naive baselines are very strong, due to the disproportionately large number of positive examples – this is especially true for the *1:1* setup which reaches perfect precision. In other words, whenever there is only one alignment candidate, it is already the correct one. The *HYB* approach also yields good results thanks to the high precision of its two components, but recall is an issue for gloss similarity due to the richer morphology and different formation of compounds in German. We did not use a compound splitter (an obvious extension for future work), so that, for instance *"Kinderspiel"* and *"Spiel für Kinder"* (both meaning *"a game for children"*) could not be lexically matched. However, when machine learning is applied, the recall can again be significantly improved at only a negligible expense of precision. Here, as for the WordNet-Wiktionary dataset, the joint modeling of distance and gloss similarity allows to correctly align more borderline examples. While the strong bias towards positive examples might make this dataset not fully representative of a full alignment task (which is the eventual goal of WSA), the results still beat the strong baselines in terms of F-measure and thus indicate that WSA, and especially our approach, works well on such a large-scale dataset.

	Wiktionary-Wikipedia (En)				Wiktionary-Wikipedia (De)			
	P	R	F_1	A	P	R	F_1	A
Random	0.41	0.49	0.45	0.48	0.68	0.40	0.51	0.46
1:1	0.17	0.56	0.26	0.33	**1.0**	0.63	0.77	0.75
1st	0.23	0.88	0.36	0.37	0.93	0.66	0.78	0.74
SIM	0.60	0.67	0.63	0.84	0.85	0.46	0.60	0.57
DWSA	0.78	0.55	0.65	0.87	0.85	0.61	0.71	0.66
HYB	0.62	**0.79**	0.70	0.86	0.90	0.72	0.80	0.75
SVM	0.82	0.70	**0.76**	0.92	0.76	0.84	0.80	0.71
Naive Bayes	0.79	0.69	0.73	0.92	0.85	0.54	0.66	0.62
Bayesian Network	**0.91**	0.63	0.74	**0.93**	0.86	0.81	0.83	0.77
Perceptron	0.82	0.70	**0.76**	0.92	0.75	**0.92**	0.82	0.73
Decision Tree	0.79	0.69	0.73	0.92	0.87	0.81	**0.84**	**0.78**
Agreement	-	-	0.79	0.95	-	-	0.85	0.89

Table 3: Results for Wiktionary-Wikipedia alignment in English and German: Using baselines (top), approaches from previous work (middle) and different machine learning classifiers (bottom). We report precision, recall, F-measure (the harmonic mean of both) and accuracy. Best results for each value and dataset are marked in bold. The inter-annotator agreements A_0 and F_1 are given as upper bounds.

Error analysis Error sources for our system are mostly the same as for the previously reported approaches – if equivalent concepts are described very differently (known as the "lexical gap", e.g. the senses *"divulge confidential information"* and *"to confess under interrogation"* of the verb *to sing*) and happen to be not very close in the resource graph, i.e. both similarity measures fail at once, they are likely not aligned (false negatives). On the other hand, false positives occur for examples such as *Brand*, which is the name of districts in two different German cities (Aachen and Zwickau). The sense descriptions are very much alike, and the senses are also located in similar regions of the resource graphs (roughly speaking, *German geography*), which makes the distinction hard. Addressing these issues might be possible by computing more sophisticated gloss similarity measures (e.g. using lexical expansion (Iida et al., 2008)) or enhancing the graph construction process. In general, however, there are no discernible systematic errors made by our system.

5 Conclusions and future work

We have shown that through joint modeling of different similarity measures for WSA the overall alignment quality in terms of F-measure can be significantly improved over the state of the art for each and every of the considered four datasets. This proves that such a joint usage of global structure as well as the content of the LSRs is indeed preferable over using either of them in isolation or combining them in a simple backoff approach, since it effectively utilizes both ways of calculating similarity.

Apart from substantially improving WSA performance, we also present two new datasets for Wiktionary-Wikipedia alignment in English and German which fill a considerable gap in the previous work on WSA. One of Wiktionary's explicit purposes is to complement the knowledge in Wikipedia, so that an alignment between these widely used resources seems a natural and important extension to the body of work in this field. Especially for (semi-) automatic translation tasks, this resource combination seems extremely promising due to the abundant multilingual content in both resources (see Section 3.1.1). We suggested a comparable combination of Wiktionary and OmegaWiki in the past (Matuschek et al., 2013), but the much larger Wikipedia is bound to hold even more potential. Moreover, the German dataset is of unprecedented size, allowing more credible statements about the performance of WSA algorithms in a full alignment scenario. Another interesting aspect is that this dataset was derived from links created by the crowd of Wiktionary editors, not by expert annotators; thus, it can be considered the first crowdsourced WSA dataset. This type of dataset creation is also one aspect of future work. We want to investigate in more detail to what extent these alignments are trustworthy, what steps are necessary

to improve the dataset's size and quality, and how negative examples (i.e. non-alignments) can be more reliably derived. We also plan to find out if such datasets could be created for other Wiktionary language editions.

The fact that the achieved results are close to the human agreement suggests that, for the datasets considered, there is not much room for improvement. Thus, we plan to apply and adapt the algorithm to LSRs with different properties than the ones considered here, such as the more syntax-focused FrameNet (Ruppenhofer et al., 2010) which only recently has received research attention in automatic WSA (Hartmann and Gurevych, 2013). The usage of syntactic features to express sense similarity has not been thoroughly explored yet, and it seems a promising direction to make further progress in WSA. Usage of more elaborate textual similarity features (e.g. covering semantic similarity or using lexical expansion) as it was suggested for text reuse detection (Bär et al., 2012) would be another direction worth exploring.

Inspired by the semi-automatic construction of the Wiktionary-Wikipedia gold standard for English from existing datasets, we also want to investigate whether an alignment of more than two resources at once (n-way alignment) is feasible, using joint knowledge from all LSRs involved. For instance, the information that two senses in resources A and B share a strong resemblance to a sense in another resource C could be expressed by an additional feature.

Acknowledgements

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg Professorship Program under grant No. I/82806 and by the Hessian research excellence program "Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz (LOEWE)" as part of the research center "Digital Humanities". We would also like to thank the anonymous reviewers for their helpful remarks.

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41, Athens, Greece.

Daniel Bär, Torsten Zesch, and Iryna Gurevych. 2012. Text Reuse Detection Using a Composition of Text Similarity Measures. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 167–184, Mumbai, India, December.

Francis Bond and Ryan Foster. 2013. Linking and Extending an Open Multilingual Wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1352–1362, Sofia, Bulgaria, August.

Kostadin Cholakov, Judith Eckle-Kohler, and Iryna Gurevych. 2014. Automated verb sense labelling based on linked lexical resources. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, pages 68–77, Gothenburg, Sweden, April.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, pages 1–8, Philadelphia, USA.

Jordi Daudé, Lluís Padró, and German Rigau. 2003. Validation and tuning of wordnet mapping techniques. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP'03)*, Borovets, Bulgaria.

Gerard De Melo and Gerhard Weikum. 2008. A Machine Learning Approach to Building Aligned Wordnets. In *Proceedings of the First International Conference on Global Interoperability for Language Resources*, pages 163–170, Hong Kong.

Gerard De Melo and Gerhard Weikum. 2010. Providing Multilingual, Multimodal Answers to Lexical Database Queries. In *Proceedings of the 7th Language Resources and Evaluation Conference (LREC 2010)*, pages 348–355, Valetta, Malta.

Edsger W. Dijkstra. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271.

Judith Eckle-Kohler, Iryna Gurevych, Silvana Hartmann, Michael Matuschek, and Christian M. Meyer. 2012. UBY-LMF - A Uniform Model for Standardizing Heterogeneous Lexical-Semantic Resources in ISO-LMF. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12)*, pages 275–282, Istanbul, Turkey.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, USA.

Tiziano Flati and Roberto Navigli. 2012. The CQC algorithm: Cycling in graphs to semantically enrich and enhance a bilingual dictionary. *Journal of Artificial Intelligence Research (JAIR)*, 43:135–171.

Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth. 2012. UBY - A Large-Scale Unified Lexical-Semantic Resource Based on LMF. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL'12)*, pages 580–590, Avignon, France.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. volume 11, pages 10–18.

Silvana Hartmann and Iryna Gurevych. 2013. FrameNet on the Way to Babel: Creating a Bilingual FrameNet Using Wiktionary as Interlingual Connection. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, volume 1, pages 1363–1373, August.

Verena Henrich, Erhard Hinrichs, and Tatiana Vodolazova. 2011. Semi-Automatic Extension of GermaNet with Sense Definitions from Wiktionary. In *Proceedings of the 5th Language and Technology Conference (LTC 2011)*, pages 126–130, Poznan, Poland.

Ryu Iida, Diana McCarthy, and Rob Koeling. 2008. Gloss-based semantic similarity metrics for predominant sense acquisition. In *Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP '08)*, pages 561–568.

Egoitz Laparra, German Rigau, and Montse Cuadros. 2010. Exploring the integration of WordNet and FrameNet. In *Proceedings of the 5th Global WordNet Conference (GWC'10)*, Mumbai, India.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation (SIGDOC '86)*, pages 24–26, Toronto, Canada.

Michael Matuschek and Iryna Gurevych. 2013. Dijkstra-WSA: A Graph-Based Approach to Word Sense Alignment. *Transactions of the Association for Computational Linguistics (TACL)*, 1:151–164, May.

Michael Matuschek, Christian Meyer, and Iryna Gurevych. 2013. Multilingual Knowledge in Aligned Wiktionary and OmegaWiki for Translation Applications. *Translation: Computation, Corpora, Cognition*, 3(1):87–118.

Christian M. Meyer and Iryna Gurevych. 2011. What psycholinguists know about chemistry: Aligning Wiktionary and WordNet for increased domain coverage. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 883–892, Chiang Mai, Thailand.

Christian M. Meyer and Iryna Gurevych. 2012. Wiktionary: A new rival for expert-built lexicons? Exploring the possibilities of collaborative lexicography. In Sylviane Granger and Magali Paquot, editors, *Electronic Lexicography*, chapter 13, pages 259–291. Oxford University Press.

Andrea Moro, Hong Li, Sebastian Krause, Feiyu Xu, Roberto Navigli, and Hans Uszkoreit. 2013. Semantic rule filtering for web-scale relation extraction. In *Proceedings of the* 12^{th} *International Semantic Web Conference (ISWC 2013)*, pages 347–362, Sydney, Australia.

Kevin P. Murphy. 2012. *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)*. The MIT Press, August.

Roberto Navigli and Simone Paolo Ponzetto. 2012a. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Roberto Navigli and Simone Paolo Ponzetto. 2012b. BabelRelate! A Joint Multilingual Approach to Computing Semantic Relatedness. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, Toronto, Canada, July.

Roberto Navigli. 2006. Meaningful Clustering of Senses Helps Boost Word Sense Disambiguation Performance. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 105–112, Sydney, Australia.

Roberto Navigli. 2009. Using Cycles and Quasi-Cycles to Disambiguate Dictionary Glosses. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL'09)*, pages 594–602, Athens, Greece.

Elisabeth Niemann and Iryna Gurevych. 2011. The People's Web meets Linguistic Knowledge: Automatic Sense Alignment of Wikipedia and WordNet. In *Proceedings of the 9th International Conference on Computational Semantics (IWCS)*, pages 205–214, Oxford, UK.

Martha Palmer. 2009. SemLink: Linking PropBank, VerbNet and FrameNet. In *Proceedings of the Generative Lexicon Conference (GenLex-09)*, pages 9–15, Pisa, Italy.

Simone Paolo Ponzetto and Roberto Navigli. 2009. Large-scale taxonomy mapping for restructuring and integrating Wikipedia. In *Proceedings of the* 21^{st} *International Joint Conference on Artificial Intelligence*, pages 2083–2088, Pasadena, CA, USA.

Josef Ruppenhofer, Michael Ellsworth, Miriam R. L. Petruck, Christopher R. Johnson, and Jan Scheffczyk. 2010. *FrameNet II: Extended Theory and Practice*. International Computer Science Institute, Berkeley, CA, September.

Lei Shi and Rada Mihalcea. 2005. Putting Pieces Together: Combining FrameNet, VerbNet and WordNet for Robust Semantic Parsing. In *Computational Linguistics and Intelligent Text Processing: 6th International Conference*, volume 3406 of *Lecture Notes in Computer Science*, pages 100–111. Berlin/Heidelberg: Springer.

Multi-view Chinese Treebanking

Likun Qiu[1,2,3], **Yue Zhang**[1], **Peng Jin**[4] **and Houfeng Wang**[2]
[1]Singapore University of Technology and Design, Singapore
[2]Institute of Computational Linguistics, Peking University, China
[3]School of Chinese Language and Literature, Ludong University, China
[4]Lab of Intelligent Information Processing and Application, Leshan Normal University, China
{qiulikun,jandp,wanghf}@pku.edu.cn, yue_zhang@sutd.edu.sg

Abstract

We present a multi-view annotation framework for Chinese treebanking, which uses dependency structures as the base view and supports conversion into phrase structures with minimal loss of information. A multi-view Chinese treebank was built under the proposed framework, and the first release (PMT 1.0) containing 14,463 sentences is be made freely available. To verify the effectiveness of the multi-view framework, we implemented an arc-standard transition-based dependency parser and added phrase structure features produced by the phrase structure view. Experimental results show the effectiveness of additional features for dependency parsing. Further, experiments on dependency-to-string machine translation show that our treebank and parser could achieve similar results compared to the Stanford Parser trained on CTB 7.0.

1 Introduction

Phrase structures (PS) and dependency structures (DS) are two of the most popular grammar formalisms for statistical parsing (Collins, 2003; Charniak, 2000; McDonald et al., 2005; Nivre, 2006; Petrov and Klein, 2007; Zhang and Clark, 2008). While DS trees emphasize the grammatical relation between heads and dependents, PS trees stress the hierarchical constituent structures of sentences. Several researchers have explored DS and PS simultaneously to enhance the quality of syntactic parsing (Wang and Zong, 2010; Farkas and Bohnet, 2012; Sun and Wan, 2013) and tree-to-string machine translation (Meng et al., 2013), showing that the two types of information complement each other for NLP tasks.

Most existing Chinese and English treebanks fall into the phrase structure category, and much work has been done to convert PS into DS (Magerman, 1994; Collins et al., 1999; Collins, 2003; Sun and Jurafsky, 2004; Johansson and Nugues, 2007; Duan et al., 2007; Zhang and Clark, 2008). Research on statistical dependency parsing has frequently used dependency treebanks converted from phrase structure treebanks, such as the Penn Treebank (PTB) (Marcus et al., 1993) and Penn Chinese Treebank (CTB) (Xue et al., 2000). However, previous research shows that dependency categories in converted treebanks are simplified (Johansson and Nugues, 2007), and the widely used head-table PS to DS conversion approach encounters ambiguities and uncertainty, especially for complex coordination structures (Xue, 2007). The main reason is that the PS treebanks were designed without consideration of DS conversion, leading to inherent ambiguities in the mapping, and loss of information in the resulting DS treebanks. To minimize information loss during treebank conversions, a treebank could be designed by considering PS and DS information simultaneously; such treebanks have been proposed as *multi-view* treebanks (Xia et al., 2009). We develop a multi-view treebank for Chinese, which treats PS and DS as different views of the same internal structures of a sentence.

We choose the DS view as the base view, from which PS would be derived. Our choice is based on the effectiveness of information transfer rather than convenience of annotation (Rambow, 2010; Bhatt and Xia, 2012). Research on Chinese syntax (Zhu, 1982; Chen, 1999; Chen, 2009) shows that the phrasal category of a constituent can be derived from the phrasal categories of its immediate subconstituents and

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 257–268, Dublin, Ireland, August 23-29 2014.

PKU POS	Our POS
Ag, a, ad, ia, ja, la	a (adjective)
Bg,b, ib, jb, jm, lb	b (distinguishing words)
Dg, d, dc, df, id, jd, ld	d (adverb)
m, mq	m(number)
n, an, in, jn, ln, Ng, vn, nr, kn	n (noun)
Qg,q, qb, qc, qd, qe, qj, ql, qr, qt, qv, qz	q (measure word)
Rg,r, rr, ry, ryw, rz, rzw	r (pronoun)
Tg, t, tt	t (temporal noun)
u, ud, ue, ui, ul, uo, us, uz, Ug	u (auxiliary word)
v, iv, im, jv, lv, Vg, vd, vi, vl, vq,vu, vx, vt,kv	v (verb)
w, wd, wf, wj, wk, wky, wkz, wm,wp, ws, wt, wu, ww, wy, wyy, wyz	w (punctuation)

Table 1: Mapping from PKU POS to our POS.

the dependency categories between them (for terminal words, parts-of-speech can be used as phrasal categories). Consequently, in Chinese, the canonical PS, containing information of constituent hierarchies and phrasal categories, can be derived naturally from the canonical DS. As Xia et al. (2009) stated, a rich set of dependency categories should be designed to ensure lossless conversion from DS to PS. When the information of PS has been represented in DS explicitly or implicitly, we can convert DS to PS without ambiguity (Rambow et al., 2002).

Given our framework, a multi-view Chinese treebank, containing 14,463 sentences and 336K words, is constructed. This main corpus is based on the Peking University People's Daily Corpus. We name our treebank the Peking University Multi-view Chinese Treebank (PMT) release 1.0. To verify the usefulness of the treebank for statistical NLP, a transition-based dependency parser is implemented to include PS features produced in the derivation process of phrasal categories. We perform a set of empirical evaluations, with experimental results on both dependency parsing and dependency-to-string machine translation showing the effectiveness of the proposed annotation framework and treebank. We make the treebank, the DS to PS conversion script and the parser freely available.

2 Annotation Framework

2.1 Part-of-speech Tagset

Our part-of-speech (POS) tagset is based on the Peking University (PKU) People's Daily corpus, which consists of over 100 tags (Yu et al., 2003). We simplify the PKU tagset by syntactic distribution. The simplified tagset contains 33 POS tags. The mapping from the original PKU POS to our simplified POS is shown in Table 1. For instance, *Ag* (adjective morpheme), *ad* (adjective acting as an adverb), *ia* (adjective idioms), *ja* (adjective abbreviation) and *la* (temporary phrase acting as an adjective) are all mapped to one tag *a* (adjective). A set of basic PKU POS tags, including *c* (conjunction), *e* (interjection), *f* (localizer), *g* (morpheme), *h* (prefix), *i* (idiom), *j* (abbreviation), *k* (suffix), *l* (temporary phrase), *nr* (personal name), *nrf* (family name), *nrg* (surname), *ns* (toponym), *nt* (organization name), *nx* (non-Chinese noun), *nz* (other proper noun), *o* (onomonopeia), *p* (preposition), *q* (measure word), *r* (pronoun), *s* (locative), *x* (other non-Chinese word), *y* (sentence final particle), *z* (state adjective), are left unchanged.

2.2 Dependency Category Tagset

In a DS, the modifier is tagged with a dependency category, which denotes the role the modifier plays with regard to its head. The root word of a sentence is dependent on a virtual root node *R* and tagged with the dependency category *HED*. Table 2 lists the 32 dependency categories used in our annotation guideline. These categories are designed in consideration of PS conversion with minimal ambiguities, and can be classified according to the following criteria:

(1) whether the head dominates a compound clause (i.e. has an IC modifier) in the PS view. According to this, dependency categories can be *cross-clause* or *in-clause*. For instance, in Figure 1, the last punctuation (。) is labeled with the *cross-clause* tag *PUS*, and its head dominates an *IC* modifier. (2) the relative position of the modifier to the head. According to this, dependency categories can be *left*, *right* or *free*. For instance, the *LAD, SBV, ADV, COS, DE and ATT* labels in Figure 1 are all *left*. The *VOB* label

Tag	Description	Tag	Description
ACT	action object	LAD	left additive
ADV	adverbial	MT	modality and time
APP	appositive element	NUM	number
ATT	attribute	POB	propositional object
CMP	complement	PUN	punctuation
COO	other coordination element	PUS	cross-clause punctuation
COS	share-right-child coordination element	QUC	post-positional quantity
DE	de (modifier of 的(special function word))	QUCC	non-shared post-positional quantity
DEI	dei (modifier of 得(special function word))	QUN	quantity
DI	di (modifier of 地(special function word))	RAD	right additive
FOC	focus	RADC	non-shared right additive
HED	root of a sentence	RED	reduplicate element
IC	independent clause	SBV	subject
IOB	indirect object	TPC	topic
IS	independent structure	VOB	direct object
ISC	non-shared independent structure	VV	serial verb construction

Table 2: Proposed dependency category set.

is *right*, while the *PUS, PUN, IC* labels are *free* and can lie on both sides. (3) whether the modifiers of a head follows the right-to-left order when combined with the head for deriving the PS structure. According to this, dependency categories can be *special* (not following the right-to-left order) or *common*. For instance, in Figure 1, the word "观察 (observe)" was labeled with the *special left* tag *COS*, because it is combined with its head "体贴 (consider)" before "体贴 (consider)"'s VOB modifier on the right.

Combining the three perspectives, the 32 dependency categories can be classified into 8 classes. Categories in different classes have different priorities when attached to the head word during PS conversion.

(1) *Special left* (2 labels): COS and RED. If there is a word tagged with the special left category, all the words between this word and its head word should be taken as *special left*.

(2) *Common left* (13 labels): ADV, APP, ATT, DE, DI, FOC, NUM, QUN, SBV, TPC, VV, PUN and IS. For instance, "就要 (must)" in Figure 1 is labeled with the *common left* tag ADV and follows the right-to-left order, being combined with its head "善于 (be good at)" after "体贴 (consider)".

(3) *Common left cross-clause* (5 labels): ADV, SBV, LAD, TPC and IS. A *common left cross-clause* modifier can also act like common left in-clause, but not vice versa.

(4) *Common right* (7 labels): ACT, CMP, DEI, IOB, MT, POB and VOB. For instance, the word "心理 (psychology)" in Figure 1 is labeled with *VOB* and follows the right-to-left order.

(5) *Special right* (4 labels): QUC, RAD, PUN, IS. In particular, PUN and IS are common categories when appearing on the left side but special categories on the right side of the head.

(6) *Special right* (attached before COO) (3 labels): QUCC, RADC and ISC. These categories differ from those in the previous class in that they would be combined to the head before COO modifiers.

(7) *Free cross-clause* (2 labels): IC, PUS. IC is a clausal category and so can be used to connect two clauses. PUS denotes cross-clause punctuations.

(8) *Common left coordination* (2 labels): COO and LAD.

2.3 Rules for Annotating Punctuations

To resolve the ambiguity of finding the head of a punctuation, we make the following rules.

(1) Coupled punctuations (e.g. brackets and quotation marks) take the head word of the phrase between the two punctuations as their head.

(2) Full stops, question marks, exclamatory marks and semicolons take the topmost head word (without violating projectivity) on their left as their heads.

(3) Commas take the nearest word on the right with HED or IC, or the topmost head words on the right (if there is no right node tagged with HED or IC), or the nearest words on their left tagged with HED or IC as their heads, all under the condition of not breaking projectivity.

(4) Colons take the topmost head word (without violating projectivity) on their right as their heads.

(5) Slight-pause marks (、) take the head of the COO or COS constituent on their left as their heads.

3 Automatic Derivation of Phrase Function and Hierarchy

In our treebank, DS is represented explicitly and PS implicitly. The conversion from DS to PS consists of two steps. First, a binary PS hierarchy is generated bottom-up according to the DS. Second, each non-terminal node in the hierarchy is tagged with a phrasal tag (e.g. NP, VP) based on manual rules. We adopt the PS tagset of the CTB (Xue et al., 2000) for our treebank.

3.1 Derivation of Phrase Hierarchy

3.1.1 Derivation Algorithm

The PS trees in our grammar are binary-branching, making the derivation of hierarchical PS from DS relatively straightforward. With leaf nodes being pre-terminals, a PS is derived bottom-up by recursive combinations of neighbouring spans according to the dependency links in a sentence. In this process, a head word is always combined with the nearest modifier that is currently not in the constituents it dominates. The only ambiguities lie in the orders in which neighbouring PS are combined to form a larger PS, which can be denoted as (A (B C)) versus ((A B) C), with A, B, and C being three neighbouring spans. For the above ambiguity to exist, the head word for each span must bare the dependency links ($A \curvearrowleft B \curvearrowright C$), with the head word of B being the head of those of A and C.

In most cases, the (A (B C)) structure is chosen. An intuitive example is that a verb is first combined with the object (VOB, a *common right* category) to form a VP, before being combined with the subject (SBV, a *common left* category) to form an IP. One example of ((A B) C) structures is the coordination structure shown in Figure 1, where the spans headed by "观察 (observe)" and "心理 (psychology)" are combined after those by "观察 (observe)" and "体贴 (consider)", due to the fact that "心理 (psychology)" is a shared object to the coordinated verbs, linked by a COS (a *special left* category) arc. In general, the modifiers of a given head are attached according the following priorities:

(1) the *special left* category $>$ (2) the *common right* category $>$ (3) the *common left* category $>$ (4) the *special right* category before COO $>$ (5) the *common left* coordination category $>$ (6) the other *special right* category $>$ (7) the *free cross-clause* clausal category (IC) $>$ (8) the *common left cross-clause* category $>$ (9) the *free cross-clause* punctuations (PUS).

3.1.2 A Case Study: Generating the Hierarchy of Coordination Structure

We take coordination structures as an example to illustrate the PS hierarchy generation process. Typically, researchers treat the rightmost conjunct as the head of a coordinate structure. However, doing so introduces modifier scope ambiguities when modifiers are also attached to the rightmost head. Vice versa, treating the leftmost conjunct as the head will lead to ambiguities when modifiers attached to the left head (Che et al., 2012). Another choice is treating the conjunction as the head (Huang et al., 2000; Xue, 2007). However, this is usually not preferred since it makes parsing more difficult and a choice still has to be made between the left and right elements when there is no conjunction in a coordinate structure (Xue, 2007). Our strategy is as follows: (1) Choose the rightmost conjunct as the head to eliminate the ambiguities when the modifiers are attached to the left; (2) Classify coordinate structures into common coordinate structures (COO) and sharing-right-child coordinate structures (COS). COO words are taken as *common left* nodes (as shown in Figure 2), while COS words are *special left* nodes (as shown in Figure 1). Doing so avoids the aforementioned scope ambiguities for modifiers.

3.2 Derivation of Phrasal Category

Several Chinese linguists discuss the issue of deriving phrasal categories from the syntactic categories of the PS and DS context. Both Zhu (1982) and Chen (1999) state that if two phrases have constituents with the same phrasal categories and the dependency types between them are also the same, the phrasal categories of their combinations must be the same. Consequently, it is natural to derive the category of a phrase from the phrasal categories of the immediate constituents and the dependency type between the constituents. We make a set of rules for the derivation, each being a DS pattern/phrasal type pair. The DS pattern is a modifier-head link with associated information such as the dependency category (DepCate)

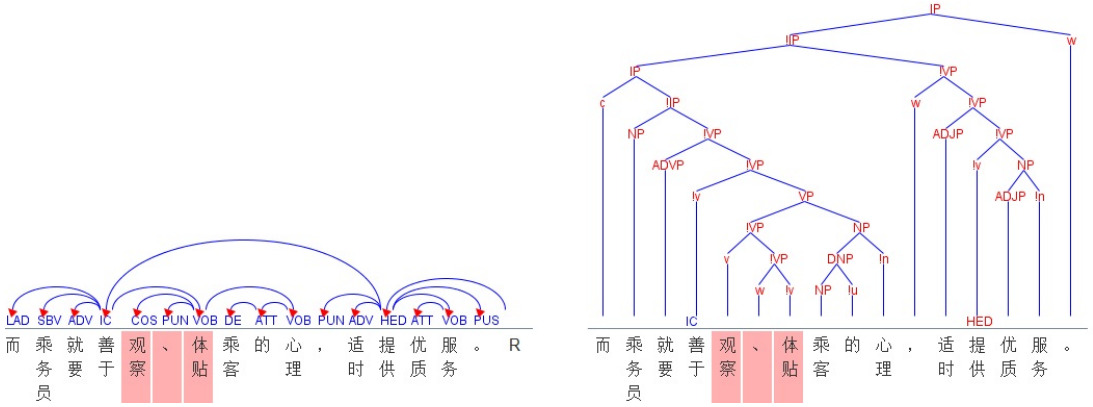


Figure 1: An instance of DS-PS conversion (而 (moreover) 乘务员 (crew) 就要 (must) 善于 (be good at) 观察 (observe) 体贴 (consider) 乘客 (passenger) 的 ('s) 心理 (psychology)， 适时 (timely) 提供 (provide) 优质 (quality) 服务 (service)). "!" denotes the head constituent.

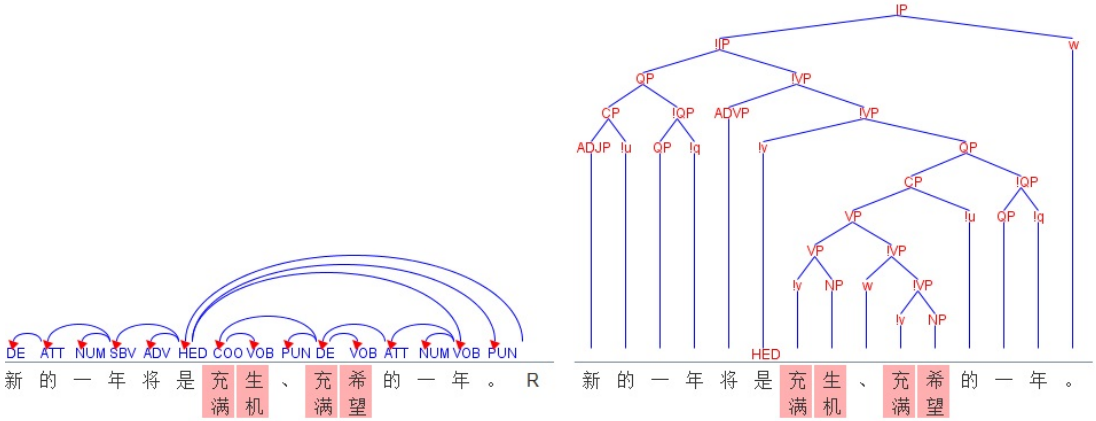


Figure 2: A second instance of DS-PS conversion (新 (new) 的 (de, an auxiliary word) 一 (one) 年 (year) 将 (will) 是 (be) 充满 (be full of) 生机 (vitality) 、 充满 (be full of) 希望 (hope) 的 (de, an auxiliary word) 一 (one) 年 (year)). "!" denotes the head constituent.

and the phrasal categories (POS tags for terminial nodes) of the subphrases that the modifier and head dominates. Some high-frequency rules are listed in Table 3.

For instance, the phrasal category of 充满 (be full of) 生机 (vitality) in Figure 2 is *VP* using the rule (v-NP-VOB, VP). Executing the derivation algorithms in Section 3.1 and derivation rules in Section 3.2, a DS in the proposed framework can be converted into corresponding PS, as shown in Figure 1 and 2.

4 The Annotation Process of PMT

According to the proposed schema, we constructed the multi-view Chinese treebank (PMT), version 1.0, which contains about 14,463 sentences and 336K words, and supports both the PS view and DS view. Our treebanking is based on the work of Yu et al. (2003), who built a segmented and POS-tagged Chinese corpus (the PFR Corpus), and released a sub-corpus containing about 1.1M words for free[1]. We choose the previous 14,463 sentences from the corpus, follow the original word segmentation standard but simplify the POS tagset according to the mapping rules described in Section 2.1. Then each sentence is annotated into a projective dependency tree according to the annotation framework described in this paper.

To speed up the annotation, a statistical dependency parser is used to give automatic parse trees and annotators are required to check each tree on a visualized annotation platform, which supports detecting

[1]http://klcl.pku.edu.cn/ShowNews.aspx?id=110

HCate	MCate	DepCate	PCate	HCate	MCate	DepCate	PCate
v	NP	VOB	VP	VP	NP	SBV	IP
IP	w	PUS	IP	NP	w	PUN	NP
n	NP	ATT	NP	VP	IP	IC	IP
p	NP	POB	PP	n	CP	ATT	NP
n	DNP	ATT	NP	NP	CP	ATT	NP
u	VP	DE	CP	NP	DNP	ATT	NP
NP	n	COO	NP	v	IP	VOB	VP
VP	n	SBV	IP	VP	v	ADV	VP
u	IP	DE	CP	NP	NP	ATT	NP
p	LCP	POB	PP	VP	c	LAD	VP
VP	d	ADV	VP	VP	PP	ADV	VP
IP	IP	IC	IP	VP	NP	VOB	VP
u	NP	DE	DNP	VP	r	SBV	IP
NP	NP	COO	NP	VP	r	SBV	IP
NP	n	ATT	NP	VP	VP	IC	IP

Table 3: Some rules for generating phrasal categories. HCate, MCate and PCate denote the phrasal category of the head subphrase, the modifier subphrase and the combined phrase, respectively.

invalid derivation from DS to PS.

For quality control, a detailed annotation guideline is provided with abundant instances for different types of syntactic structures in Mandarin Chinese. More information of the guideline can be found in an extended version of this paper. In addition, we adopt the annotation strategy for the construction of the Penn Chinese Treebank (Xue et al., 2000) — one annotator examines an automatic parse tree first, and a second annotator verifies the annotation of the first annotator.

5 A Transition-Based Parser for Multi-view Treebank

In order to demonstrate the usefulness of our treebank in comparison with existing Chinese treebanks, we perform empirical analysis to the treebank, by the statistical dependency parsing and dependency-to-string machine translation tasks. Several researchers explored joint DS and PS information to enhance the quality of syntactic parsing (Wang and Zong, 2010; Farkas and Bohnet, 2012; Sun and Wan, 2013). Most tried to combine the outputs of constituent and dependency parsers by stacking or bagging. Since our treebank is multi-view, it is possible to combine DS features and PS features directly in the decoding process.

We implemented an arc-standard transition-based dependency parser (Nivre, 2008) based on the arc-eager parser of Zhang and Nivre (2011), which is a state-of-the-art transition-based dependency parser (Zhang and Nivre, 2012). It is more reasonable to derive the phrasal category of a phrase after the complete subtree (phrase) rather than partial subtree headed by a word has been built. The arc-standard parser differs from the arc-eager parser in that it postpones the attachment of right-modifiers until the complete subtrees headed by the modifiers themselves have been built. Because of this, we add PS features into an arc-standard parser rather than an arc-eager one.

The parser processes a sentence from left to right, using a stack to maintain partially built derivations and a queue to hold next incoming words. Three transition actions (LEFT, RIGHT and SHIFT) are defined to consume input words from the queue and construct arcs using the stack (Nivre, 2008):

LEFT pops the second top item off the stack, and adds it as a modifier to the top of the stack;

RIGHT pops the top item off the stack, and adds it as a modifier to the second top of the stack;

SHIFT removes the front of the queue and pushes it onto the top of the stack.

Table 4 show the feature templates of our parser, most of which are based on those of Zhang and Nivre (2011). The contextual information consists of the top four nodes of the stack (S_3, S_2, S_1 and S_0), the next three input words (N_0, N_1 and N_2), the left and right children (ld, rd) of these nodes, and the distance between S_0 and S_1. Word and POS information from the context are manually combined.

Due to the multi-view nature of our treebank, the DS parser can be extended naturally to incorporate PS information. Further, because our PS is binary branching, each constituent corresponds to a dependency link. In the decoding process, we derive the phrasal category c of a subtree whenever a dependency link

features of stack top	$S_0wt; S_0w; S_0t; S_1wt; S_1w; S_1t; S_2wt; S_2w; S_2t; S_3wt; S_3w; S_3t; N_0wt;$
features of next input	$N_0w; N_0t; N_1wt; N_1w; N_1t; N_2wt; N_2w; N_2t;$
bigram features	$S_0wS_1w; S_0wS_1t; S_0tS_1w; S_0tS_1t; S_0wN_0w; S_0wN_0t; S_0tN_0w; S_0tN_0t;$
children features of S_0	$S_0ldw; S_0ldt; S_0ldwt; S_0ldd; S_0rdw; S_0rdt; S_0rdwt; S_0rdd;$
children features of S_1	$S_1ldw; S_1ldt; S_1ldwt; S_1ldd; S_1rdw; S_1rdt; S_1rdwt; S_1rdd;$
distance features	$S_0wDistance(S_0, S_1); S_0tDistance(S_0, S_1); S_1wDistance(S_0, S_1); S_1tDistance(S_0, S_1);$
PS features	$S_0c; S_1c; S_0cS_1c; S_0wS_1c; S_0tS_1c; S_0wS_1dS_1c; S_1wS_0c; S_1tS_0c; S_1wS_0dS_0c; S_0cS_1cS_0S_1c$

Table 4: Transition-based feature templates for the arc-standard dependency parser. *w*=word; *t*=POS tag. *d*=dependency category. *c*=phrasal category.

is established, using the derivation rules in Table 3. Using *c* and its combination with other features, we can produce several PS features, as shown in Table 4. By this simple extension of features, we arrive at an efficient linear-time joint DS and PS parser.

6 Experiments

6.1 Syntactic Parsing

PMT 1.0 contains all the articles of People's Daily from January 1st to January 10th, 1998. Sentences 12001-13000 and 13001-14463 are used as the development and test set, respectively. The remaining sentences are used as training data.

Several state-of-the-art statistical parsers, including Mate-tools (Bohnet, 2010)[2], BerkeleyParser (Petrov and Klein, 2007)[3], ZPar-dep (Zhang and Nivre, 2011) and ZPar-con (Zhang and Clark, 2009; Zhu et al., 2013)[4] are used for comparison. We used the gold segmentation, and the Stanford POS tagger (Toutanova et al., 2003) (version 3.3.1) to provide automatic POS tags for all the experiments. The POS tagger was trained on the PKU corpus (Yu et al., 2003) containing articles of People's Daily from January 2000 to June 2000. It achieved a 95.78% precision on the PMT. In the baseline parser (Ours-standard), the feature templates in Table 4 except the PS features are used. We refer to the parser after adding PS features as Ours-PS. The results of dependency (ZPar-eager, Ours-standard, Ours-PS and Mate-tools) and constituent parsers (BerkeleyParser and ZPar-con) are measured by the unlabeled accuracy score (UAS), labeled accuracy score (LAS) and bracketing f-measure (BF), respectively.

We display the parsing results in Table 5. Our dependency parser (Ours-PS) outperforms the baseline parser (Ours-standard) with a 0.47% increase in UAS. For additional evaluation, we also converted the DS trees parsed by the dependency parsers to PS using the conversion procedure in Section 3, in order to compare the results of dependency parsers and constituent parsers. The three ZPar-based dependency parsers gave higher accuracies than the two state-of-the-art constituent parsers. In particular, the DS2PS outputs of Ours-PS parser outperforms the PS outputs of Berkeley Parer with 0.62% higher BF.

Both Zhang and Clark (2011) and Petrov and McDonald (2012) show that DS trees converted from the outputs of PS parsers outperform those produced directly by DS parsers trained on DS conversions of the CTB. Interestingly, our evaluation on the PMT gave results in the opposite direction: parsers trained on the DS treebank outperforms parsers trained on the PS conversion. One possible reason is that parser errors can be hidden in the conversion process. Take the sentence in Figure 3 for example. Figure 3(a) shows the correct PS while Figure 3(b) shows an incorrect parser output. In particular, "黎明 (dawn)" is put under the incorrect constituent. When converted into DS, both lead to the correct link, with "黎明 (dawn)" being the SBV modifier of "降临 (come)" (Figure 3(c)). As a result, the PS parser error is erased in the conversion into DS. The same can happen in DS to PS conversion.

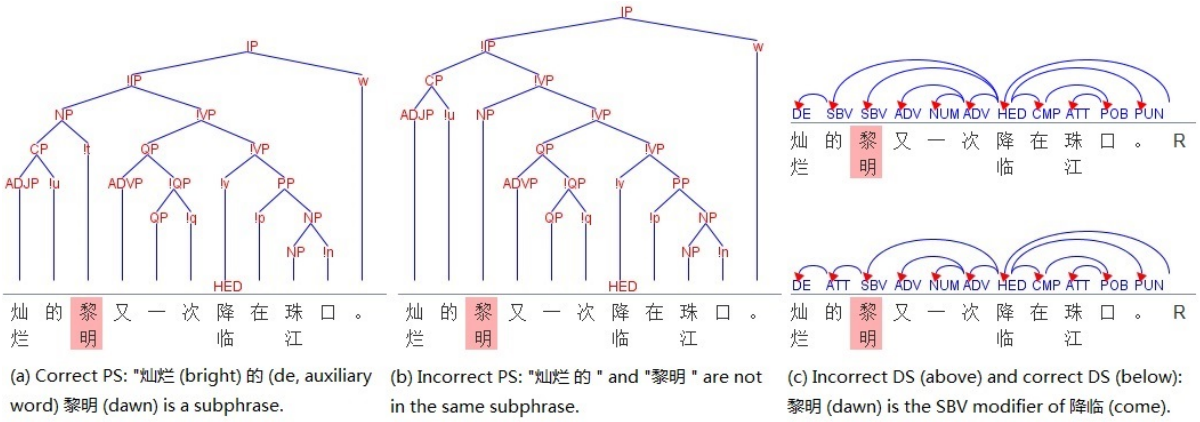
6.2 Dependency-to-string Machine Translation

We compare the effects of our treebank and the Stanford dependencies converted from CTB on machine translation, using the dependency-to-string system of Xie et al. (2011). Our training corpus consists of

[2]https://code.google.com/p/mate-tools/

[3]http://code.google.com/p/berkeley-parser-analyser/

[4]http://sourceforge.net/projects/zpar/



(a) Correct PS: "灿烂 (bright) 的 (de, auxiliary word) 黎明 (dawn) is a subphrase.

(b) Incorrect PS: "灿烂 的 " and "黎明 " are not in the same subphrase.

(c) Incorrect DS (above) and correct DS (below): 黎明 (dawn) is the SBV modifier of 降临 (come).

Figure 3: An instance where PS parser error is erased in the PS to DS conversion (灿烂 (bright) 的 (de, an auxiliary word) 黎明 (dawn) 又 (again) 一 (one) 次(time) 降临 (come) 在 (in) 珠江 (the Pearl River) 口 (estuary)). "!" denotes the head constituent.

Dependency Parsing			Constituent Parsing		Constituent Parsing(DS2PS)	
Parsers	**UAS**	**LAS**	**len<=40 words**	**Unlimited**	**len<=40 words**	**Unlimited**
Mate-tools	82.98	79.37	/	/	84.77	83.43
ZPar-dep	82.73	80.20	/	/	85.47	84.33
Ours-standard	82.81	80.04	/	/	85.53	84.47
Ours-PS	**83.28**	**80.50**	/	/	**85.92**	**84.84**
Berkeley Parser	/	/	85.25	84.22	/	/
ZPar-con	/	/	85.02	84.12	/	/

Table 5: Parsing results on our treebank using automatic POS-tags.

31K Chinese-English sentence pairs from the Xinhua Corpus (Liu et al., 2006), and we used NIST MT Evaluation 2006 test set as the development set, and the NIST 2003 (MT03), 2004 (MT04) and 2005 (MT05) test sets as the test sets. For Stanford dependency trees, we parsed the source sentences with the Stanford Parser (Chang et al., 2009) (version 3.3.1), which was trained on CTB 7.0. For the PMT treebank, we used the Ours-PS parser, trained with 14000 sentences (the last 463 sentences are used as development data for the parser). All the MT configurations are the same as Xie et al. (2011).

The results are shown in Table 6. The Chinese-English translation outputs using our parser and treebank are slightly lower but comparable to those using the Stanford Parser. Note that our treebank contains 336K words on People's Daily, while the CTB 7.0 contains about 1.19M words, most on Xinhua, the source of the MT training and test data. This result to some degree demonstrates the usefulness of our treebank for NLP applications, in comparison with a well-established treebank.

7 Related Work

PS Treebanks and DS Conversion PTB (Marcus et al., 1993) and CTB (Xue et al., 2000) are the most widely used treebanks for English and Chinese in the literature. Both are in PS. For conversion from PS to DS, a head-table approach (Magerman, 1994; Collins, 2003; Yamada and Matsumoto, 2003; Sun and Jurafsky, 2004; Nivre, 2006; Johansson and Nugues, 2007; Duan et al., 2007; Zhang and Clark, 2008) is widely used. However, the reliability of head tables has been questioned (Xue, 2007). Xue (2007) proposed a novel approach that better exploits the structural information in the CTB and pointed out that the results of the approach and the widely used Penn2Malt tools[5] agree only 60.6% in terms of unlabeled dependency. The coordination structures, in particular, are not properly converted by Penn2Malt.

DS Treebanks and PS Conversion An existing DS treebank for Chinese is the Chinese Dependency Treebank (Che et al., 2012), which is not designed as a multi-view treebank. For conversion from DS to PS, Xia and Palmer (2001) compare three algorithms. These algorithms do not use a rich set of

[5]http://stp.lingfil.uu.se/ nivre/research/Penn2Malt.html

Parsers	Treebank	MT03(BLEU4)	MT04(BLEU4)	MT05(BLEU4)
Stanford Parser	CTB 7.0	28.23	29.00	25.72
Ours-PS	PMT 1.0	27.73	28.71	25.20

Table 6: Results of dependency-to-string machine translation.

dependency categories, only distinguishing arguments and modifiers. Xia et al. (2009) propose a DS-to-PS algorithm, which assumes that a given DS is identical to a flattened version of the desired PS, and then introduce a set of conversion rules. Their error analysis show that coordination and punctuation amount to about 32.1% of conversion errors, while other errors fall into missing content in DS and inconsistency in the target treebank (PTB). This analysis demonstrates that coordination and punctuation should be tackled carefully for the conversion between PS and DS, which we do in the design of our treebank. Bhatt et al. (2011) presented three scenarios arising in the conversion of DS into PS. Bhatt and Xia (2012) further described 7 phenomena of incompatibility in the conversion from DS to PS, mainly involving the annotation of empty categories, yet coordination structure and punctuation were not discussed.

Multi-view Treebanks The Tiger (Brants et al., 2002) and TüBa-D/Z (Telljohann et al., 2003) treebanks for German seek to explicitly represent both PS and DS by labeling both nodes and edges in the syntactic tree. For these treebanks, both dependency categories and phrasal categories have been annotated explicitly. The English side of the Czech-English parallel corpus is annotated and linked also as both PS (original PTB annotation) and DS (Hajic et al., 2012), while the DS is a conversion of the original PS. Our multi-view treebank is different in that dependency categories and phrasal categories derive from each other. The Hindi/Urdu treebank (Xia et al., 2009; Palmer et al., 2009; Bhatt et al., 2009) can be taken as a multi-view treebank. Its PS view is derived automatically from the DS. However, the converted PS is not a PS with a full hierarchy but a flattened one (Xia et al., 2009).

8 Conclusion

We presented an DS-based multi-view annotation framework, and built a Chinese treebank according to the framework and an arc-standard transition-based dependency parser that exploits the multi-view nature of the treebank. We used SMT as an example to demonstrate the usefulness of our treebank for NLP applications. Experiments showed that the proposed treebank and parser can give similar results to the Stanford Parser trained on CTB 7.0. We make our treebank (PMT 1.0) (`http://klcl.pku.edu.cn/ResourceList.aspx`), the DS to PS conversion script and the proposed parser (`http://sourceforge.net/projects/zpar/`) freely available.

Acknowledgments

We gratefully acknowledge the invaluable assistance of Jun Xie in helping us make experiments on dependency-to-string machine translation. We thank Ji Ma for helpful suggestions and comments, and Yijia Liu for providing a generic arc-standard dependency parser. We also thank all the students from Ludong University and Leshan Normal University for annotating the large-scale treebank. Special thanks to Xiaoyan Wang, Na Wei and Huarong Ni from Ludong University. Finally, we thank the anonymous reviewers for their constructive comments, and gratefully acknowledge the support of the Singapore Ministry of Education (MOE) AcRF Tier 2 grant T2MOE201301, the National Natural Science Foundation of China (No. 61103089, No. 61373056), National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101), Major National Social Science Fund of China (No. 12&ZD227), Scientific Research Foundation of Shandong Province Outstanding Young Scientist Award (No. BS2013DX020) and Humanities and Social Science Projects of Ludong University (No. WY2013003).

References

Rajesh Bhatt and Fei Xia. 2012. Challenges in converting between treebanks: a case study from the hutb. In *Proceedings of META-RESEARCH Workshop on Advanced Treebanking, in conjunction with LREC-2012, Istanbul, Turkey*.

Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 186–189.

Rajesh Bhatt, Owen Rambow, and Fei Xia. 2011. Linguistic phenomena, analyses, and representations: Understanding conversion between treebanks. In *Proceedings of IJCNLP 2011*, pages 1234–1242.

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of COLING 2010*, pages 89–97.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the workshop on treebanks and linguistic theories*, volume 168.

Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D Manning. 2009. Discriminative reordering with chinese grammatical relations features. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation*, pages 51–59.

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

Wanxiang Che, Li Zhenghua, and Liu Ting. 2012. *Chinese Dependency Treebank 1.0*. Linguistic Data Consortium.

Baoya Chen. 1999. *Chinese Linguistic Methodology in the 21th Century, 1898-1998*. Shandong Education Publishing House.

Baoya Chen. 2009. *Contemporary Linguistics*. Higher Education Press.

Michael Collins, Lance Ramshaw, Jan Hajič, and Christoph Tillmann. 1999. A statistical parser for czech. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 505–512. Association for Computational Linguistics.

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

Xiangyu Duan, Jun Zhao, and Bo Xu. 2007. Probabilistic parsing action models for multi-lingual dependency parsing. In *Proceedings of EMNLP-CoNLL 2007*, pages 940–946.

Richárd Farkas and Bernd Bohnet. 2012. Stacking of dependency and phrase structure parsers. In *Proceedings of COLING 2012*, pages 849–866.

Jan Hajic, Eva Hajicová, Jarmila Panevová, Petr Sgall, Ondrej Bojar, Silvie Cinková, Eva Fucíková, Marie Mikulová, Petr Pajas, Jan Popelka, et al. 2012. Announcing prague czech-english dependency treebank 2.0. In *LREC*, pages 3153–3160.

Chu-Ren Huang, Feng-Yi Chen, Keh-Jiann Chen, Zhao-ming Gao, and Kuang-Yu Chen. 2000. Sinica treebank: design criteria, annotation guidelines, and on-line interface. In *Proceedings of the second workshop on Chinese language processing: held in conjunction with the 38th ACL-Volume 12*, pages 29–37.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *16th Nordic Conference of Computational Linguistics*, pages 105–112. University of Tartu.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of COLING-ACL 2006*, pages 609–616. Association for Computational Linguistics.

David M Magerman. 1994. Natural language parsing as statistical pattern recognition. *arXiv preprint cmp-lg/9405009*.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):313–330.

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

Fandong Meng, Jun Xie, Linfeng Song, Yajuan Lü, and Qun Liu. 2013. Translation with source constituency and dependency trees. In *Proceedings of EMNLP 2010*, pages 1066–1076.

Joakim Nivre. 2006. *Inductive dependency parsing*. Springer.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

Martha Palmer, Rajesh Bhatt, Bhuvana Narasimhan, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure. In *The 7th International Conference on Natural Language Processing*, pages 14–17.

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

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*, volume 59.

Owen Rambow, Cassandre Creswell, Rachel Szekely, Harriet Taber, and Marilyn A Walker. 2002. A dependency treebank for english. In *Proceedings of LREC 2002*.

Owen Rambow. 2010. The simple truth about dependency and phrase structure representations: An opinion piece. In *Proceedings of HLT-NAACL 2010*, pages 337–340.

Honglin Sun and Daniel Jurafsky. 2004. Shallow semantc parsing of Chinese. In *Proceedings of HLT-NAACL 2004*, pages 249–256.

Weiwei Sun and Xiaojun Wan. 2013. Data-driven, PCFG-based and pseudo-PCFG-based models for Chinese dependency parsing. *Transactions of the Association for Computational Linguistics*, 1(1):301–314.

Heike Telljohann, Erhard W Hinrichs, Sandra Kübler, Heike Zinsmeister, and Kathrin Beck. 2003. Stylebook for the tübingen treebank of written german (tüba-d/z). In *Seminar für Sprachwissenschaft, Universität Tübingen, Germany*.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of NAACL 2003-Volume 1*, pages 173–180.

Zhiguo Wang and Chengqing Zong. 2010. Phrase structure parsing with dependency structure. In *Proceedings of Coling 2010: Posters*, pages 1292–1300.

Fei Xia and Martha Palmer. 2001. Converting dependency structures to phrase structures. In *Proceedings of HLT 2001*, pages 1–5. Association for Computational Linguistics.

Fei Xia, Owen Rambow, Rajesh Bhatt, Martha Palmer, and Dipti Misra Sharma. 2009. Towards a multi-representational treebank. In *The 7th International Workshop on Treebanks and Linguistic Theories*.

Jun Xie, Haitao Mi, and Qun Liu. 2011. A novel dependency-to-string model for statistical machine translation. In *Proceedings of EMNLP 2011*, pages 216–226.

Nianwen Xue, Fei Xia, Shizhe Huang, and Anthony Kroch. 2000. The bracketing guidelines for the penn chinese treebank (3.0).

Nianwen Xue. 2007. Tapping the implicit information for the PS to DS conversion of the chinese treebank. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistics Theories*.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT 2003*, volume 3.

Shiwen Yu, Huiming Duan, Xuefeng Zhu, Bin Swen, and Baobao Chang. 2003. Specification for corpus processing at peking university: Word segmentation, pos tagging and phonetic notation. *Journal of Chinese Language and Computing*, 13(2):121–158.

Yue Zhang and Stephen Clark. 2008. A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. In *Proceedings of EMNLP 2008*, pages 562–571.

Yue Zhang and Stephen Clark. 2009. Transition-based parsing of the Chinese treebank using a global discriminative model. In *Proceedings of IWPT 2009*, pages 162–171.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of ACL-HLT 2011: short papers-Volume 2*, pages 188–193.

Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *COLING (Posters)*, pages 1391–1400.

Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. 2013. Fast and accurate shift-reduce constituent parsing. In *Proceedings of ACL 2013*, pages 434–443.

Dexi Zhu. 1982. *Grammar Finder*. Commercial Press.

Rapid Development of a Corpus with Discourse Annotations using Two-stage Crowdsourcing

Daisuke Kawahara[†‡] **Yuichiro Machida**[†] **Tomohide Shibata**[†‡] **Sadao Kurohashi**[†‡]
Hayato Kobayashi[§] **Manabu Sassano**[§]

[†]Graduate School of Informatics, Kyoto University
[‡]CREST, Japan Science and Technology Agency
[§]Yahoo Japan Corporation

{dk, shibata, kuro}@i.kyoto-u.ac.jp, machida@nlp.ist.i.kyoto-u.ac.jp,
{hakobaya, msassano}@yahoo-corp.jp

Abstract

We present a novel approach for rapidly developing a corpus with discourse annotations using crowdsourcing. Although discourse annotations typically require much time and cost owing to their complex nature, we realize discourse annotations in an extremely short time while retaining good quality of the annotations by crowdsourcing two annotation subtasks. In fact, our experiment to create a corpus comprising 30,000 Japanese sentences took less than eight hours to run. Based on this corpus, we also develop a supervised discourse parser and evaluate its performance to verify the usefulness of the acquired corpus.

1 Introduction

Humans understand text not by individually interpreting clauses or sentences, but by linking such a text fragment with another in a particular context. To allow computers to understand text, it is essential to capture the precise relations between these text fragments. This kind of analysis is called discourse parsing or discourse structure analysis, and is an important and fundamental task in natural language processing (NLP). Systems for discourse parsing are, however, available only for major languages, such as English, owing to the lack of corpora with discourse annotations.

For English, several corpora with discourse annotations have been developed manually, consuming a great deal of time and cost in the process. These include the Penn Discourse Treebank (Prasad et al., 2008), RST Discourse Treebank (Carlson et al., 2001), and Discourse Graphbank (Wolf and Gibson, 2005). Discourse parsers trained on these corpora have also been developed and practically used. To create the same resource-rich environment for another language, a quicker method than the conventional time-consuming framework should be sought. One possible approach is to use crowdsourcing, which has actively been used to produce various language resources in recent years (e.g., (Snow et al., 2008; Negri et al., 2011; Hong and Baker, 2011; Fossati et al., 2013)). It is, however, difficult to crowdsource the difficult judgments for discourse annotations, which typically consists of two steps: finding a pair of spans with a certain relation and identifying the relation between the pair.

In this paper, we propose a method for crowdsourcing discourse annotations that simplifies the procedure by dividing it into two steps. The point is that by simplifying the annotation task it is suitable for crowdsourcing, but does not skew the annotations for use in practical discourse parsing. First, finding a discourse unit for the span is a costly process, and thus we adopt a clause as the discourse unit, since this is reliable enough to be automatically detected. We also limit the length of each target document to three sentences and at most five clauses to facilitate the annotation task. Secondly, we detect and annotate clause pairs in a document that hold logical discourse relations. However, since this is too complicated to assign as one task using crowdsourcing, we divide the task into two steps: determining the existence of logical discourse relations and annotating the type of relation. Our two-stage approach is a robust method in that it confirms the existence of the discourse relations twice. We also designed the tagset of discourse relations for crowdsourcing, which consists of two layers, where the upper layer contains the following three classes: "CONTINGENCY," "COMPARISON" and "OTHER." Although the task

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 269–278, Dublin, Ireland, August 23-29 2014.

settings are simplified for crowdsourcing, the obtained corpus and knowledge of discourse parsing could be still useful in general discourse parsing.

In our experiments, we crowdsourced discourse annotations for Japanese, for which there are no publicly available corpora with discourse annotations. The resulting corpus consists of 10,000 documents, each of which comprises three sentences extracted from the web. Carrying out this two-stage crowdsourcing task took less than eight hours. The time elapsed was significantly shorter than the conventional corpus building method.

We also developed a discourse parser by exploiting the acquired corpus with discourse annotations. We learned a machine learning-based model for discourse parsing based on this corpus and evaluated its performance. An F1 value of 37.9% was achieved for contingency relations, which would be roughly comparable with state-of-the-art discourse parsers on English. This result indicates the usefulness of the acquired corpus. The resulting discourse parser would be effectively exploited in NLP applications, such as sentiment analysis (Zirn et al., 2011) and contradiction detection (Murakami et al., 2009; Ennals et al., 2010).

The novel contributions of this study are summarized below:

- We propose a framework for developing a corpus with discourse annotations using two-stage crowdsourcing, which is both cheap and quick to execute, but still retains good quality of the annotations.
- We construct a Japanese discourse corpus in an extremely short time.
- We develop a discourse parser based on the acquired corpus.

The remainder of this paper is organized as follows. Section 2 introduces related work, while Section 3 describes our proposed framework and reports the experimental results for the creation of a corpus with discourse annotations. Section 4 presents a method for discourse parsing based on the corpus as well as some experimental results. Section 5 concludes the paper.

2 Related Work

Snow et al. (2008) applied crowdsourcing to five NLP annotation tasks, but the settings of these tasks are very simple. There have also been several attempts to construct language resources with complex annotations using crowdsourcing. Negri et al. (2011) proposed a method for developing a cross-lingual textual entailment (CLTE) corpus using crowdsourcing. They tackled this complex data creation task by dividing it into several simple subtasks: sentence modification, type annotation and sentence translation. The creative CLTE task and subtasks are quite different from our non-creative task and subtasks of discourse annotations. Fossati et al. (2013) proposed FrameNet annotations using crowdsourcing. Their method is a single-step approach to only detect frame elements. They verified the usefulness of their approach through an experiment on a small set of verbs with only two frame ambiguities per verb. Although they seem to be running a larger-scale experiment, its result has not been revealed yet. Hong and Baker (2011) presented a crowdsourcing method for selecting FrameNet frames, which is a part of the FrameNet annotation process. Since their task is equivalent to word sense disambiguation, it is not very complex compared to the whole FrameNet annotation process. These FrameNet annotations are still different from discourse annotations, which are our target. To the best of our knowledge, there have been no attempts to crowdsource discourse annotations.

There are several manually-crafted corpora with discourse annotation for English, such as the Penn Discourse Treebank (Prasad et al., 2008), RST Discourse Treebank (Carlson et al., 2001), and Discourse Graphbank (Wolf and Gibson, 2005). These corpora were developed from English newspaper articles. Several attempts have been made to manually create corpora with discourse annotations for languages other than English. These include the Potsdam Commentary Corpus (Stede, 2004) for German (newspaper; 2,900 sentences), Rhetalho (Pardo et al., 2004) for Portuguese (scientific papers; 100 documents; 1,350 sentences), and the RST Spanish Treebank for Spanish (da Cunha et al., 2011) (several genres; 267 documents; 2,256 sentences). All of these consist of relatively small numbers of sentences compared with the English corpora containing several tens of thousands sentences.

In recent years, there have been many studies on discourse parsing on the basis of the above hand-annotated corpora (e.g., (Pitler et al., 2009; Pitler and Nenkova, 2009; Subba and Di Eugenio, 2009; Hernault et al., 2010; Ghosh et al., 2011; Lin et al., 2012; Feng and Hirst, 2012; Joty et al., 2012; Joty et al., 2013; Biran and McKeown, 2013; Lan et al., 2013)). This surge of research on discourse parsing can be attributed to the existence of corpora with discourse annotations. However, the target language is mostly English since English is the only language that has large-scale discourse corpora. To develop and improve discourse parsers for languages other than English, it is necessary to build large-scale annotated corpora, especially in a short period if possible.

3 Development of Corpus with Discourse Annotations using Crowdsourcing

3.1 Corpus Specifications

We develop a tagged corpus in which pairs of discourse units are annotated with discourse relations. To achieve this, it is necessary to determine target documents, discourse units, and a discourse relation tagset. The following subsections explain the details of these three aspects.

3.1.1 Target Text and Discourse Unit

In previous studies on constructing discourse corpora, the target documents were mainly newspaper texts, such as the Wall Street Journal for English. However, discourse parsers trained on such newspaper corpora usually have a problem of domain adaptation. That is to say, while discourse parsers trained on newspaper corpora are good at analyzing newspaper texts, they generally cannot perform well on texts of other domains.

To address this problem, we set out to create an annotated corpus covering a variety of domains. Since the web contains many documents across a variety of domains, we use the Diverse Document Leads Corpus (Hangyo et al., 2012), which was extracted from the web. Each document in this corpus consists of the first three sentences of a Japanese web page, making these short documents suitable for our discourse annotation method based on crowdsourcing.

We adopt the clause as a discourse unit, since spans are too fine-grained to annotate using crowdsourcing and sentences are too coarse-grained to capture discourse relations. Clauses, which are automatically identified, do not need to be manually modified since they are thought to be reliable enough. Clause identification is performed using the rules of Shibata and Kurohashi (2005). For example, the following rules are used to identify clauses as our discourse units:

- clauses that function as a relatively strong boundary in a sentence are adopted,
- relative clauses are excluded.

Since workers involved in our crowdsourcing task need to judge whether clause pairs have discourse relations, the load of these workers increases combinatorially as the number of clauses in a sentence increases. To alleviate this problem, we limit the number of clauses in a document to five. This limitation excludes only about 5% of the documents in the original corpus.

Our corpus consists of 10,000 documents corresponding to 30,000 sentences. The total number of clauses in this corpus is 39,032, and thus the average number of clauses in a document is 3.9. The total number of clause pairs is 59,426.

3.1.2 Discourse Relation Tagset

One of our supposed applications of discourse parsing is to automatically generate a bird's eye view of a controversial topic as in Statement Map (Murakami et al., 2009) and Dispute Finder (Ennals et al., 2010), which identify various relations between statements, including contradictory relations. We assume that expansion relations, such as elaboration and restatement, and temporal relations are not important for this purpose. This setting is similar to the work of Bethard et al. (2008), which annotated temporal relations independently of causal relations. We also suppose that temporal relations can be annotated separately for NLP applications that require temporal information. We determined the tagset of discourse relations

Upper type	Lower type	Example
CONTINGENCY	Cause/Reason	【ボタンを押したので】【お湯が出た。】 [since (I) pushed the button] [hot water was turned on]
	Purpose	【試験に受かるために】【必死に勉強した。】 [to pass the exam] [(I) studied a lot]
	Condition	【ボタンを押せば】【お湯が出る。】 [if (you) push the button] [hot water will be turned on]
	Ground	【ここにカバンがあるから】【まだ社内にいるだろう。】 [here is his/her bag] [he/she would be still in the company]
COMPARISON	Contrast	【あのレストランは寿司はおいしいが】【ラーメンは普通だ。】 [at that restaurant, sushi is good] [ramen is so-so]
	Concession	【あのレストランは確かにおいしいが】【値段は高い。】 [that restaurant is surely good] [the price is high]
OTHER	(Other)	【家に着いてから】【雨が降ってきた。】 [After being back home] [it began to rain]

Table 1: Discourse relation tagset with examples.

by referring to the Penn Discourse Treebank. This tagset consists of two layers, where the upper layer contains three classes and the lower layer seven classes as follows:

- CONTINGENCY
  - Cause/Reason (causal relations and not conditional relations)
  - Purpose (purpose-action relations where the purpose is not necessarily accomplished)
  - Condition (conditional relations)
  - Ground (other contingency relations including pragmatic cause/condition)
- COMPARISON (same as the Penn Discourse Treebank)
  - Contrast
  - Concession
- OTHER (other weak relation or no relation)

Note that we do not consider the direction of relations to simplify the annotation task for crowdsourcing. Table 1 shows examples of our tagset.

Therefore, our task is to annotate clause pairs in a document with one of the discourse relations given above. Sample annotations of a document are shown below. Here, clause boundaries are shown by "::" and clause pairs that are not explicitly marked are allocated the "OTHER" relation.

Cause/Reason 気がつけば::梅雨も明けてました。::毎日暑い日が続きますね。::【父の手術も無事に終わり、】::【少しだけほっとしてます。】
... [the surgery of my father ended safely] [(I) am relieved a little bit]

Contrast 今日とある企業のトップの話を聞くことが出来た。::経営者として何事も全てビジネスチャンスに変えるマインドが大切だと感じた。::【生きていく上で追い風もあれば、】::【逆風もある。】
... [There is tailwind to live,] [there is also headwind.]

3.2 Two-stage Crowdsourcing for Discourse Annotations

We create a corpus with discourse annotations using two-stage crowdsourcing. We divide the annotation task into the following two subtasks: determining whether a clause pair has a discourse relation excluding "OTHER," and then, ascertaining the type of discourse relation for a clause pair that passes the first stage.

Probability	Number
$= 1.0$	64
> 0.99	554
> 0.9	1,065
> 0.8	1,379
> 0.5	2,655
> 0.2	4,827
> 0.1	5,895
> 0.01	9,068
> 0.001	12,277
> 0.0001	15,554

Table 2: Number of clause pairs resulting from the judgments of discourse relation existence.

3.2.1 Stage 1: Judgment of Discourse Relation Existence

This subtask determines whether each clause pair in a document has one of the following discourse relations: Cause/Reason, Purpose, Condition, Ground, Contrast, and Concession (that is, all the relations except "OTHER"). Workers are shown examples of these relations and asked to determine only the existence thereof.

In this subtask, an item presented to a worker at a particular time consists of all the judgments of clause pairs in a document. By adopting this approach, each worker considers the entire document when making his/her judgments.

3.2.2 Stage 2: Judgment of Discourse Relation Type

This subtask involves ascertaining the discourse relation type for a clause pair that passes the first stage. The result of this subtask is one of the seven lower types in our discourse relation tagset. Workers are shown examples of these types and then asked to select one of the relations. If a worker chooses "OTHER," this corresponds to canceling the positive determination of the existence of the discourse relation in stage one.

In this subtask, an item is the judgment of a clause pair. That is, if a document contains more than one clause pair that must be judged, the judgments for this document are divided into multiple items, although this is rare.

3.3 Experiment and Discussion

We conducted an experiment of the two-stage crowdsourcing approach using Yahoo! Crowdsourcing.[1] To increase the reliability of the produced corpus, we set the number of workers for each item for each task to 10. The reason why we chose this value is as follows. While Snow et al. (2008) claimed that an average of 4 non-expert labels per item in order to emulate expert-level label quality, the quality of some tasks increased by increasing the number of workers to 10. We also tested hidden gold-standard items once every 10 items to examine worker's quality. If a worker failed these items in serial, he/she would have to take a test to continue the task.

We obtained judgments for the 59,426 clause pairs in the 10,000 documents of our corpus in the first stage of crowdsourcing, i.e., the subtask of determining the existence of discourse relations. We calculated the probability of each label using GLAD[2] (Whitehill et al., 2009), which was proved to be more reliable than the majority voting. This probability corresponds to the probability of discourse relation existence of each clause pair. Table 2 lists the results. We set a probability threshold to select those clause pairs whose types were to be judged in the second stage of crowdsourcing. With this threshold set to 0.01, 9,068 clause pairs (15.3% of all the clause pairs) were selected. The threshold was set fairly low to allow low-probability judgments to be re-examined in the second stage.

[1] http://crowdsourcing.yahoo.co.jp/

[2] http://mplab.ucsd.edu/~jake/OptimalLabelingRelease1.0.3.tar.gz

Lower type	All	prob > 0.8
Cause/Reason	2,104	1,839 (87.4%)
Purpose	755	584 (77.4%)
Condition	1,109	925 (83.4%)
Ground	442	273 (61.8%)
Contrast	437	354 (81.0%)
Concession	80	49 (61.3%)
Sum of the above discourse relations	4,927	4,024 (81.7%)
Other	4,141	3,753 (90.6%)
Total	9,068	7,777 (85.8%)

Table 3: Results of the judgments of lower discourse relation types.

Upper type	All	prob > 0.8
CONTINGENCY	4,439	3,993 (90.0%)
COMPARISON	516	417 (80.8%)
Sum of the above discourse relations	4,955	4,410 (89.0%)
OTHER	4,113	3,753 (91.2%)
Total	9,068	8,163 (90.0%)

Table 4: Results of the judgments of upper discourse relation types.

The discourse relation types of the 9,068 clause pairs were determined in the second stage of crowdsourcing. We extended GLAD (Whitehill et al., 2009) for application to multi-class tasks, and calculated the probability of the labels of each clause pair. We assigned the label (discourse relation type) with the highest probability to each clause pair. Table 3 gives some statistics of the results. The second column in this table denotes the numbers of each discourse relation type, while the third column gives the numbers of each type of clause pair with a probability higher than 0.80. Table 4 gives statistics of the results when the lower discourse relation types are merged into the upper types. Table 5 shows some examples of the resulting annotations.

Carrying out the two separate subtasks using crowdsourcing took approximately three hours and five hours with 1,458 and 1,100 workers, respectively. If we conduct this task at a single stage, it would take approximately 33 (5 hours / 0.153) hours. It would be four times longer than our two-stage approach. Such single-stage approach is also not robust since it does not have a double check mechanism, with which the two-stage approach is equipped. We spent 111 thousand yen and 113 thousand yen (approximately 1,100 USD, respectively) for these subtasks, which would be extremely less expensive than the projects of conventional discourse annotations.

For the examples in Table 5, we confirmed that the discourse relation types of the top four examples were surely correct. However, we judged the type (Contrast) of the bottom example as incorrect. Since the second clause is an instantiation of the first clause, the correct type should be "Other." We found such errors especially in the clause pairs with a probability lower than 0.80.

4 Development of Discourse Parser based on Acquired Discourse Corpus

To verify the usefulness of the acquired corpus with discourse annotations, we developed a supervised discourse parser based on the corpus, and evaluated its performance. We built two discourse parsers using the annotations of the lower and upper discourse relation types, respectively. From the annotations in the first stage of crowdsourcing (i.e., judging the existence of discourse relations), we assigned annotations with a probability less than 0.01 as "OTHER." Of the annotations acquired in the second stage (i.e., judging discourse relation types), we adopted those with a probability greater than 0.80 and discarded the rest. After this preprocessing, we obtained 58,135 (50,358 + 7,777) instances of clause pairs for the lower-type discourse parser and 58,521 (50,358 + 8,163) instances of clause pairs for the upper-type

Prob	# W	Type	Document
1.00	6/10	Cause/Reason	ツツジ科・ツツジ属。【花が陰暦五月に咲くため】【「皐月」と呼ばれている。】市制２０年を記念して、１９７９年１１月３日に制定された。 ... [Since the flower blooms in the fifth lunar month] [it is called "Satsuki."] ...
0.99	4/10	Condition	【↓マップ上の吹き出しをクリックすると】【おすすめルートがご覧になれます。】市町村名をクリックすると「見どころ・体験・食」の情報がご覧になれます。緑色の表記は各スポットの写真がご覧になれます。 [If you click the balloon on the map] [you can see the recommended route] ...
0.81	3/10	Purpose	ダイランティアはマナによって支えられた世界。しかし、人類の繁栄と共に世界樹が３年に一度結実させる「大いなる実り」だけでは人類の繁栄を支えることができなくなってしまった。【そして「大いなる実り」を求めて】【各国が戦争を繰り広げていく。】 ... [And seeking "Great harvest"] [each country is engaged in a war]
0.61	2/10	Cause/Reason	スケールは（一部を除き）１／３２とされている。これは単３形乾電池２本が入りやすいようにしたサイズである。動力は単３形乾電池２本とＦＡ－１３０サイズのモーター１個で、【ギヤーとシャフトの組み合わせにより動力を前後の車軸に伝達し、】【４輪を駆動する。】 ... [by transmitting power to the front and rear axle with the combination of gears and shafts] [(it) drives the four wheels.]
0.54	3/10	Contrast	来年春には、阪急百貨店が新博多駅に東急ハンズと共にお目見えする。そうなると【百貨店による顧客の奪い合いが厳しくなる。】【そこに浮上するのが、三越福岡の閉鎖の可能性である。】 ... [a scramble for customers by department stores would be severe.] [What comes out is the possibility of the closure of Fukuoka Mitsukoshi.]

Table 5: Examples of Annotations. The first column denotes the estimated label probability and the second column denotes the number of workers that assigned the designated type. In the fourth column, the clause pair annotated with the type is marked with 【】 ([ ] in English translations).

discourse parser. Of these, 4,024 (6.9%) and 4,410 (7.5%) instances, respectively, had one of the types besides "OTHER." We conducted experiments using five-fold cross validation on these instances.

To extract features of machine learning, we applied the Japanese morphological analyzer, JUMAN,[3] and the Japanese dependency parser, KNP,[4] to the corpus. We used the features listed in Table 6, which are usually used for discourse parsing.

We adopted Opal (Yoshinaga and Kitsuregawa, 2010)[5] for the machine learning implementation. This tool enables online learning using a polynomial kernel. As parameters for Opal, we used the passive-aggressive algorithm (PA-I) with a polynomial kernel of degree two as a learner and the extension to multi-class classification (Matsushima et al., 2010). The numbers of classes were seven and three for the lower- and upper-type discourse parsers, respectively. We set the aggressiveness parameter C to 0.001, which generally achieves good performance for many classification tasks. Other parameters were set to the default values of Opal.

To measure the performance of the discourse parsers, we adopted precision, recall and their harmonic mean (F1). These metrics were calculated as the proportion of the number of correct clause pairs to the

[3]http://nlp.ist.i.kyoto-u.ac.jp/EN/?JUMAN
[4]http://nlp.ist.i.kyoto-u.ac.jp/EN/?KNP
[5]http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/opal/

Name	Description
clause distance	clause distance between two clauses
sentence distance	sentence distance between two clauses
bag of words	bag of words (lemmas) for each clause
predicate	a content word (lemma) of the predicate of each clause
conjugation form of predicate	a conjugation form of the predicate of each clause
conjunction	a conjunction if it is located at the beginning of a clause
word overlapping ratio	an overlapping ratio of words between the two clauses
clause type	a lexical type output by KNP for each clause (about 100 types)
topic marker existence	existence of a topic marker in each clause
topic marker cooccurrence	existence of a topic marker in both clauses

Table 6: Features for our discourse parsers.

Type	Precision	Recall	F1
Cause/Reason	0.623 (441/708)	0.240 (441/1,839)	0.346
Purpose	0.489 (44/90)	0.075 (44/584)	0.131
Condition	0.581 (256/441)	0.277 (256/925)	0.375
Ground	0.000 (0/12)	0.000 (0/273)	0.000
Contrast	0.857 (6/7)	0.017 (6/354)	0.033
Concession	0.000 (0/0)	0.000 (0/49)	0.000
Other	0.944 (53,702/56,877)	0.992 (53,702/54,111)	0.968

Table 7: Performance of our lower-type discourse parser.

Type	Precision	Recall	F1
CONTINGENCY	0.625 (1,084/1,735)	0.272 (1,084/3,993)	0.379
COMPARISON	0.412 (7/17)	0.017 (7/417)	0.032
OTHER	0.942 (53,454/56,769)	0.988 (53,454/54,111)	0.964

Table 8: Performance of our upper-type discourse parser.

number of all recognized or gold-standard ones for each discourse relation type. Tables 7 and 8 give the accuracies for the lower- and upper-type discourse parsers, respectively.

From Table 8, we can see that our upper-type discourse parser achieved an F1 of 37.9% for contingency relations. It is difficult to compare our results with those in previous work due to the use of different data set and different languages. We, however, anticipate that our results would be comparable with those of state-of-the-art English discourse parsers. For example, the end-to-end discourse parser of Lin et al. (2012) achieved an F1 of 20.6% – 46.8% on the Penn Discourse Treebank.

We also obtained a low F1 for comparison relations. This tendency is similar to the previous results on the Penn Discourse Treebank. The biggest cause of this low F1 is the lack of unambiguous explicit discourse connectives for these relations. Although there are explicit discourse connectives in Japanese, many of them have multiple meanings and cannot be used as a direct clue for discourse relation detection (e.g., as described in Kaneko and Bekki (2014)). As reported in Pitler et al. (2009) and other studies, the identification of implicit discourse relations are notoriously difficult. To improve its performance, we need to incorporate external knowledge sources other than the training data into the discourse parsers. A promising way is to use large-scale knowledge resources that are automatically acquired from raw corpora.

5 Conclusion

We presented a rapid approach for building a corpus with discourse annotations and a discourse parser using two-stage crowdsourcing. The acquired corpus is made publicly available and can be used for research purposes.[6] This corpus can be used not only to build a discourse parser but also to evaluate its performance. The availability of the corpus with discourse annotations will accelerate the development and improvement of discourse parsing. In the future, we intend integrating automatically acquired knowledge from corpora into the discourse parsers to further enhance their performance. We also aim to apply our framework to other languages without available corpora with discourse annotations.

References

Steven Bethard, William Corvey, Sara Klingenstein, and James H. Martin. 2008. Building a corpus of temporal-causal structure. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, pages 908–915.

Or Biran and Kathleen McKeown. 2013. Aggregated word pair features for implicit discourse relation disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 69–73.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue*.

Iria da Cunha, Juan-Manuel Torres-Moreno, and Gerardo Sierra. 2011. On the development of the RST Spanish treebank. In *Proceedings of the 5th Linguistic Annotation Workshop (LAW V)*, pages 1–10.

Rob Ennals, Beth Trushkowsky, and John Mark Agosta. 2010. Highlighting disputed claims on the web. In *Proceedings of the 19th international conference on World Wide Web*, pages 341–350.

Vanessa Wei Feng and Graeme Hirst. 2012. Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 60–68. Association for Computational Linguistics.

Marco Fossati, Claudio Giuliano, and Sara Tonelli. 2013. Outsourcing FrameNet to the crowd. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 742–747.

Sucheta Ghosh, Sara Tonelli, Giuseppe Riccardi, and Richard Johansson. 2011. End-to-end discourse parser evaluation. In *Fifth IEEE International Conference on Semantic Computing (ICSC)*, pages 169–172.

Masatsugu Hangyo, Daisuke Kawahara, and Sadao Kurohashi. 2012. Building a diverse document leads corpus annotated with semantic relations. In *Proceedings of 26th Pacific Asia Conference on Language Information and Computing*, pages 535–544.

Hugo Hernault, Helmut Prendinger, David duVerle, and Mitsuru Ishizuka. 2010. HILDA: A discourse parser using support vector machine classification. *Dialogue & Discourse*, 1(3):1–33.

Jisup Hong and Collin F. Baker. 2011. How good is the crowd at "real" WSD? In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 30–37.

Shafiq Joty, Giuseppe Carenini, and Raymond Ng. 2012. A novel discriminative framework for sentence-level discourse analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 904–915.

Shafiq Joty, Giuseppe Carenini, Raymond Ng, and Yashar Mehdad. 2013. Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 486–496.

Kimi Kaneko and Daisuke Bekki. 2014. Building a Japanese corpus of temporal-causal-discourse structures based on SDRT for extracting causal relations. In *Proceedings of the EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)*, pages 33–39.

[6] http://nlp.ist.i.kyoto-u.ac.jp/EN/?DDLC

Man Lan, Yu Xu, and Zhengyu Niu. 2013. Leveraging synthetic discourse data via multi-task learning for implicit discourse relation recognition. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 476–485.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2012. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, pages 1–34.

Shin Matsushima, Nobuyuki Shimizu, Kazuhiro Yoshida, Takashi Ninomiya, and Hiroshi Nakagawa. 2010. Exact passive-aggressive algorithm for multiclass classification using support class. In *Proceedings of 2010 SIAM International Conference on Data Mining (SDM2010)*, pages 303–314.

Koji Murakami, Eric Nichols, Suguru Matsuyoshi, Asuka Sumida, Shouko Masuda, Kentaro Inui, and Yuji Matsumoto. 2009. Statement map: Assisting information credibility analysis by visualizing arguments. In *Proceedings of the 3rd Workshop on Information Credibility on the Web*, pages 43–50.

Matteo Negri, Luisa Bentivogli, Yashar Mehdad, Danilo Giampiccolo, and Alessandro Marchetti. 2011. Divide and conquer: Crowdsourcing the creation of cross-lingual textual entailment corpora. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 670–679.

Thiago Alexandre Salgueiro Pardo, Maria das Graças Volpe Nunes, and Lucia Helena Machado Rino. 2004. Dizer: An automatic discourse analyzer for Brazilian Portuguese. In *Advances in Artificial Intelligence–SBIA 2004*, pages 224–234. Springer.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 13–16.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 683–691.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn discourse treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, pages 2961–2968.

Tomohide Shibata and Sadao Kurohashi. 2005. Automatic slide generation based on discourse structure analysis. In *Proceedings of Second International Joint Conference on Natural Language Processing*, pages 754–766.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263.

Manfred Stede. 2004. The Potsdam commentary corpus. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation*, pages 96–102.

Rajen Subba and Barbara Di Eugenio. 2009. An effective discourse parser that uses rich linguistic information. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 566–574.

Jacob Whitehill, Paul Ruvolo, Ting fan Wu, Jacob Bergsma, and Javier Movellan. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2035–2043.

Florian Wolf and Edward Gibson. 2005. Representing discourse coherence: A corpus-based study. *Computational Linguistics*, 31(2):249–287.

Naoki Yoshinaga and Masaru Kitsuregawa. 2010. Kernel slicing: Scalable online training with conjunctive features. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING2010)*, pages 1245–1253.

Cäcilia Zirn, Mathias Niepert, Heiner Stuckenschmidt, and Michael Strube. 2011. Fine-grained sentiment analysis with structural features. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 336–344.

Chinese Word Ordering Errors Detection and Correction for Non-Native Chinese Language Learners

Shuk-Man Cheng, Chi-Hsin Yu, Hsin-Hsi Chen
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan
{smcheng,jsyu}@nlg.csie.ntu.edu.tw; hhchen@ntu.edu.tw

Abstract

Word Ordering Errors (WOEs) are the most frequent type of grammatical errors at sentence level for non-native Chinese language learners. Learners taking Chinese as a foreign language often place character(s) in the wrong places in sentences, and that results in wrong word(s) or ungrammatical sentences. Besides, there are no clear word boundaries in Chinese sentences. That makes WOEs detection and correction more challenging. In this paper, we propose methods to detect and correct WOEs in Chinese sentences. Conditional random fields (CRFs) based WOEs detection models identify the sentence segments containing WOEs. Segment point-wise mutual information (PMI), inter-segment PMI difference, language model, tag of the previous segment, and CRF bigram template are explored. Words in the segments containing WOEs are reordered to generate candidates that may have correct word orderings. Ranking SVM based models rank the candidates and suggests the most proper corrections. Training and testing sets are selected from HSK dynamic composition corpus created by Beijing Language and Culture University. Besides the HSK WOE dataset, Google Chinese Web 5-gram corpus is used to learn features for WOEs detection and correction. The best model achieves an accuracy of 0.834 for detecting WOEs in sentence segments. On the average, the correct word orderings are ranked 4.8 among 184.48 candidates.

1 Introduction

Detection and correction of grammatical errors are practical for many applications such as document editing and language learning. Non-native language learners usually encounter problems in learning a new foreign language and are prone to generate ungrammatical sentences. Sentences with various types of errors are written by language learners of different backgrounds. In the HSK corpus, which contains compositions of students from different countries who study Chinese in Beijing Language and Culture University (http://nlp.blcu.edu.cn/online-systems/hsk-language-lib-indexing-system.html), there are 35,884 errors at sentence level. The top 10 error types and their occurrences are listed below: Word Ordering Errors (WOE) (8,515), Missing Component (Adverb) (3,244), Missing Component (Predicate) (3,018), Grammatical Error ("Is … DE") (2,629), Missing Component (Subject) (2,405), Missing Component (Head Noun) (2364), Grammatical Error ("Is" sentence) (1,427), Redundant Component (Predicate) (1,130), Uncompleted Sentence (1,052), and Redundant Component (Adverb) (1,051). WOEs are the most frequent type of errors (Yu and Chen, 2012).

The types of WOEs in Chinese are different from those in English. A Chinese character has its own meaning in text, while individual characters are meaningless in English. Learners taking Chinese as a foreign language often place character(s) in the wrong places in sentences, and that results in wrong word(s) or ungrammatical sentences. Besides, there are no clear word boundaries in Chinese sentences.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 279–289, Dublin, Ireland, August 23-29 2014.

Word segmentation is fundamental in Chinese language processing (Huang and Zhao, 2007). WOEs may result in wrong segmentation. That may make WOEs detection and correction more challenging.

This paper aims at identifying the positions of WOEs in the text written by non-native Chinese language learners, and proposes candidates to correct the errors. It is organized as follows. Section 2 surveys the related work. Section 3 gives an overview of the study. Section 4 introduces the dataset used for training and testing. Sections 5 and 6 propose models to detect and correct Chinese WOEs, respectively. Section 7 concludes this study and propose some future work.

2 Related Work

There are only a few researches on the topic of detection and correction of WOEs in Chinese language until now. We survey the related work from the four aspects: (1) grammatical errors made by non-native Chinese learners, (2) word ordering errors in Chinese language, (3) computer processing of grammatical errors in Chinese language, and (4) grammatical error correction in other languages.

Leacock et al. (2014) give thorough surveys in automated grammatical error detection for language learners. Error types, available corpora, evaluation methods, and approaches for different types of errors are specified. Several shared tasks on grammatical error correction in English have been organized in recent years, including HOO 2011 (Dale and Kilgarriff, 2011), HOO 2012 (Dale et al., 2012) and CoNLL 2013 (Ng et al., 2013). Different types of grammatical errors are focused: (1) HOO 2011: article and preposition errors, (2) HOO 2012: determiner and preposition errors, and (3) CoNLL 2013: article or determiner errors, preposition errors, noun number errors, verb form errors, and subject-verb agreement errors. In Chinese, spelling check evaluation was held at SIGHAN Bake-off 2013 (Wu et al., 2013). However, none of the above evaluations deals with word ordering errors.

Wang (2011) focuses on the Chinese teaching for native English-speaking students. He shows the most frequent grammatical errors made by foreigners are missing components, word orderings and sentence structures. One major learning problem of foreign learners is the influence of negative transfer of mother tongue. Lin (2011) studies the biased errors of word order in Chinese written by foreign students in the HSK corpus. Sun (2011) compares the word orderings between English and Chinese to figure out the differences in sentence structures. Yu and Chen (2012) propose classifiers to detect sentences containing WOEs, but they do not deal with where WOEs are and how to correct them.

Wagner et al. (2007) deal with common grammatical errors in English. They consider frequencies of POS n-grams and the outputs of parsers as features. Gamon et al. (2009) identify and correct errors made by non-native English writers. They first detect article and preposition errors, and then apply different techniques to correct each type of errors. Huang et al. (2010) propose a correction rule extraction model trained from 310,956 sets of erroneous and corrected pairwise sentences. Some studies related to word orderings are specific to the topic of pre-processing or post-processing of statistical machine translation, such as Galley and Manning (2008), Setiawan et al. (2009), and DeNero and Uszkoreit (2011).

The major contributions of this paper cover the following aspects: (1) application aspect: detecting and correcting a common type of Chinese written errors of foreign learners with HSK corpus; (2) language aspect: considering the effects of words and segments in Chinese sentences; and (3) resource aspect: exploring the feasibility of using a Chinese web n-gram corpus in WOE detection/correction.

3 Overview of a Chinese Word Ordering Detection and Correction System

Figure 1 sketches an overview of our Chinese WOE detection and correction system. It is composed of three major parts, including dataset preparation, WOE detection, and WOE correction. At first, a corpus is prepared. Sentences containing WOEs are selected from the corpus and corrected by two Chinese native speakers. This corpus will be used for training and testing. Then, a sentence is segmented into a sequence of words, and chunked into several segments based on punctuation marks. Regarding words and segments as fundamental units reduce the number of reordering and limit the reordering scope. The segments containing WOEs are identified by using CRF-based models. Finally, the candidates are generated by reordering and ranked by Ranking SVM-based models. To examine the performance of WOE correction, two datasets, C_{ans} and C_{sys}, consisting of error segments labelled by human and detected by our system, respectively, are employed.

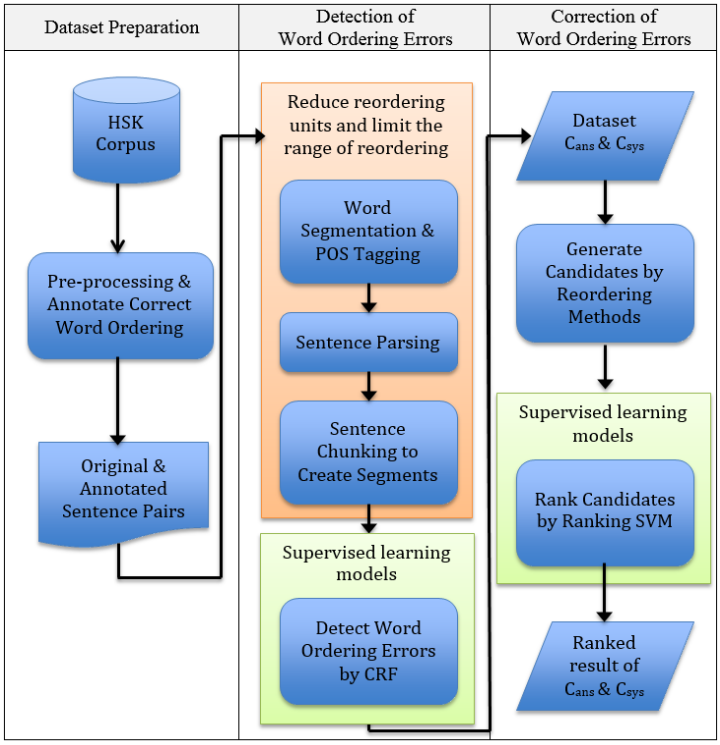


Figure 1: Overview of word ordering error detection and correction.

The example shown below demonstrates the major steps. This sentence is composed of three segments. The second segment contains a WOE, i.e., 今年夏天毕业了大学 (Graduated college this summer). The correct sentence should be 今年夏天大学毕业了 (Graduated from college this summer).

(1) Reduce the number of reordering units in a sentence by using word segmentation.

我	叫	王大安	，	今年	夏天	毕业	了	大学	，	现在	找	工作	。

(I / am /Wang Daan/ , /this /summer /graduated/le /college /, /now /look for/job /.)

(2) Chunk a sentence into segments by punctuation marks.

我	叫	王大安	，	今年	夏天	毕业	了	大学	，	现在	找	工作	。

(3) Detect the possible segments containing WOEs in a sentence by CRF-based methods.

我	叫	王大安	，	今年	夏天	毕业	了	大学	，	现在	找	工作	。

(4) Reorder words in an erroneous segment and generate candidates.

我	叫	王大安	，	今年	夏天	毕业	大学	了	，	现在	找	工作	。

…

我	叫	王大安	，	今年	夏天	大学	毕业	了	，	现在	找	工作	。

(5) Rank candidates and suggest correct word ordering by Ranking SVM-based methods.

我	叫	王大安	，	今年	夏天	大学	毕业	了	，	现在	找	工作	。

…

4 A Word Ordering Errors (WOEs) Corpus

HSK dynamic composition corpus created by Beijing Language and Culture University is adopted. It contains the Chinese composition articles written by non-native Chinese learners. There are 11,569 articles and 4.24 million characters in 29 composition topics. Composition articles are scanned into text and annotated with tags of error types ranging from character level, word level, sentence level, to discourse level. There are 35,884 errors at sentence level, and WOEs are the most frequent type at this level. Total 8,515 sentences are annotated with WOEs. We filter out sentences with multiple error types and remove duplicate sentences. Total 1,150 error sentences with WOEs remain for this study.

Two Chinese native speakers are asked to correct the 1,150 sentences. Only reordering operation is allowed during correction. A dataset composed of 1,150 sets of original sentence S and its two corrections A1 and A2 is formed for training and testing in the experiments. A1 may be different from A2. The following shows an example. Without context, either A1 or A2 is acceptable.

S: 她我们兄弟姊妹鼓励学音乐和外语。
(She we encouraged to study music and foreign languages.)
A1: 我们兄弟姊妹鼓励她学音乐和外语。
(We encouraged her to study music and foreign languages.)
A2: 她鼓励我们兄弟姊妹学音乐和外语。
(She encouraged us to study music and foreign languages.)

In some cases, A1 and/or A2 may be equal to S. That is, the annotators may think S is correct. That may happen when context is not available. Finally, 327 of 1,150 sets contain different corrections. Both A1 and A2 are equal to S in 27 sets. Total 47 sentences corrected by one annotator are the same as the original sentences, and total 65 sentences corrected by another annotator are the same as the original sentences. This corpus is available at http://nlg.csie.ntu.edu.tw/nlpresource/woe_corpus/.

Figure 2 shows the Damerau Levenshtein distance between the original sentences S and the corrections A1 and A2. It counts the minimum number of operations needed to transform a source string into a target one. Here the operation is the transposition of two adjacent characters. Total 823 sets of A1 and A2 have a distance of 0. It means 71.5% of sentences have the same corrections by the two Chinese native speakers. The distances between S and A1 are similar to those between S and A2. Total 850 sets of original sentences and the corrections have a distance below 10 characters and 1,014 sets of sentences have a distance below 20. We can also observe that the number of sentences with even distances is larger than that of sentences with odd distances because most of the Chinese words are composed of two characters.

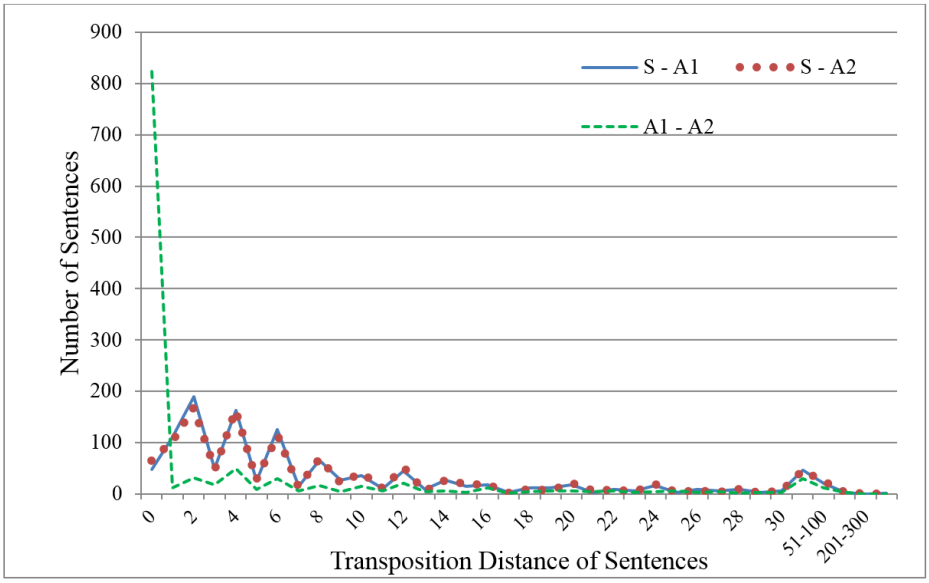


Figure 2: Transposition distance among the original sentences and two corrections.

5 Detection of Word Ordering Errors

This section first defines the fundamental units for error detection, then introduces the error detection models along with their features, and finally presents and discusses the experimental results.

5.1 Fundamental Units for Reordering

Permutation is an intuitive way to find out the correct orderings, but its cost is very high. Unrestrictive permutation will generate too many candidates to be acceptable in computation time. What units to be reordered in what range under what condition has be considered. Chinese is different from English in that characters are the smallest meaningful units, and there are no clear word boundaries. Computation cost and segmentation performance is a trade-off to select character or word as a reordering unit. On the one hand, using words as the reordering units will reduce the number of candidates generated. On the other hand, word segmentation results will affect the performance of WOE detection and correction. The following two examples show that reordering the words cannot generate the correct answers. In these two examples, a word in the original sentence (S) is segmented into two words in the correct sentence (A). These words are underlined. Because a word is regarded as a unit for reordering, the correct sentence cannot be generated by word reordering only in these two cases.

S: 他 / 教给 / 学生们 / 英语 /。
(He / teach to / students / English / .)
A: 他 / 给 / 学生们 / 教 / 英语 / 。
(He / for / students / teach / English / .)
S: 最近 / 我 / 开始 / 学 / 中国 / 的 / 做菜。
(Recently / I / start to / learn / China / ‘s / cooking cuisine.)
A: 最近 / 我 / 开始 / 学 / 做 / 中国 / 的 / 菜。
(Recently / I / start to / learn / cooking / China / ‘s /cuisine.)

Total 76 sets of sentences belong to such cases. They occupy 6% of the experimental dataset. Considering the benefits of words, we still adopt words as reordering units in the following experiments.

To prevent reordering all the words in the original sentences, we further divide a sentence into segments based on comma, caesura mark, semi-colon, colon, exclamation mark, question mark, and full stop. Sentence segments containing WOEs will be detected and words will be reordered within the segments to generate the candidates for correction. In our dataset, there are only 31 sets of sentences (i.e., 2.7%) with WOEs across segments. The following shows two examples. The underlined words are moved to other segments.

S: 其实，我还是做事情的时候，不怎么老实。
(In fact, when I am still working, I am not honest.)
A: 其实，我做事情的时候，还是不怎么老实。
(In fact, when I am working, I am still not honest.)
S: 所以有绝对的导游工作经验，无须再培训。
(Therefore we have absolute guide work experience, we do not need retraining.)
A: 有绝对的导游工作经验，所以无须再培训。
(We have absolute guide work experience, therefore we do not need retraining.)

In summary, the upper bound of the correction performance would be 91.3%. That is, 6%+2.7% of sentences cannot be resolved.

5.2 Word Ordering Errors Detection Models

Conditional random fields (CRFs) (Lafferty, 2001) are used to implement the WOE detection in sentence segments. Segments with WOEs are labelled with answer tags before training. The original sentence S written by non-native Chinese learner is compared with the annotated correct sentence A. Characters are compared from the start and the end of sentences, respectively. The positions are marked ERR_{start} and ERR_{end} once the characters are different. All words within ERR_{start} and ERR_{end} are marked ERR_{range}. The longest common subsequence (LCS) within ERR_{range} of S and ERR_{range} of A are excluded from ERR_{range} and the remaining words are marked ERR_{words}. Figure 3 shows an example. We use BIO encoding (Ramshaw and Marcus, 1995) to label segments with WOEs. Segments contain-

ing words in ERR_{words} are defined to be segments with WOEs. The leftmost segment with WOEs is tagged B, and the following segment with WOEs are tagged I. Those segments without WOEs are tagged O.

Figure 3: An example for ERR_{range} and ERR_{words}.

Table 1 lists the distribution of B, I and O segments. Recall that two Chinese native speakers are asked to correct the 1,150 sentences, thus we have two sets of B-I-O tagging.

Tagging→	B Tag		I Tag		O Tag		Total
Statistics→	#Segments	Percentage	#Segments	Percentage	#Segments	Percentage	Segments
Annotator 1	1111	40.6%	53	1.9%	1572	57.5%	2736
Annotator 2	1097	40.1%	59	2.2%	1580	57.7%	2736

Table 1: Distribution of B, I, and O segments.

Five features are proposed as follows for CRF training. Google Chinese Web 5-gram corpus (Liu, Yang and Lin, 2010) is adopted to get the frequencies of Chinese words for f_{PMI}, f_{Diff} and f_{LM}.

(1) Segment Pointwise Mutual Information (f_{PMI})
$PMI(Seg_i)$ defined below measures the coherence of a segment Seg_i by calculating PMI of all word bigrams in Seg_i. To avoid the bias from different lengths, the sum of PMI of all word bigrams is divided by n-1 for normalization, where n denotes the segment length. The segment PMI values are partitioned into intervals by equal frequency discretization. Feature f_{PMI} of the segment Seg_i reflects the label of the interval to which $PMI(Seg_i)$ belongs.

$$PMI(Seg_i) = \frac{1}{n-1}\sum_{k=1}^{n-1} log\frac{P(w_k, w_{k+1})}{P(w_k)P(w_{k+1})}$$

(2) Inter-segment PMI Difference (f_{Diff})
Feature f_{Diff} captures the PMI difference between two segments Seg_{j-1} and Seg_j. It aims to measure the coherence between segments. The feature setting is also based on equal frequency discretization.

(3) Language Model (f_{LM})
Feature f_{LM} uses bigram language model to measure the log probability of the words in a segment defined below. Labels of interval are also determined by equal frequency discretization.

$$LM(Seg_i) = log\ [P(w_1)\prod_{k=1}^{n-1}\frac{P(w_k,w_{k+1})}{P(w_k)}] = log\,P(w_1) + \sum_{k=1}^{n-1} log\frac{P(w_k,w_{k+1})}{P(w_k)}$$

(4) Tag of the previous segment (f_{Tag})
Feature f_{Tag} reflects the tag B, I or O of the previous segment.

(5) CRF bigram template (f_B)
Feature f_B is a bigram template given by SGD-CRF tool[1]. Bigram template combines the tags of the previous segment and current segment, and generates $T*T*N$ feature functions, where T is number of tags and N is number of strings expanded with a macro.

[1] http://leon.bottou.org/projects/sgd

5.3 Results and Discussion

WOE detection models will annotate the segments of a sentence with labels B, I or O. These labels will determine which segments may contain WOEs. In the experiments, we use 5-fold cross-validation to evaluate the proposed models. Performance for detecting WOEs is measured at the segment and the sentence levels, respectively. The metrics at the segment level are defined as follows. Here set notation is adopted. The symbol $|S|$ denotes the number of elements in the set S which is derived by the logical formula after vertical bar. $TAG_{pred}(SEG)$ and $TAG_{ans}(SEG)$ mean the labels of segment SEG tagged by WOE detection model and human, respectively. The symbol m denotes total number of segments in the test set.

$$Accuracy = \frac{|\{\, SEG \mid TAG_{pred}(SEG) = TAG_{ans}(SEG) \,\}|}{m}$$

$$Recall = \frac{|\{\, SEG \mid TAG_{pred}(SEG) \in (\mathrm{B},\mathrm{I}) \cap TAG_{ans}(SEG) \in (\mathrm{B},\mathrm{I}) \,\}|}{|\{\, SEG \mid TAG_{ans}(SEG) \in (\mathrm{B},\mathrm{I}) \,\}|}$$

$$Precision = \frac{|\{\, SEG \mid TAG_{pred}(SEG) \in (\mathrm{B},\mathrm{I}) \cap TAG_{ans}(SEG) \in (\mathrm{B},\mathrm{I}) \,\}|}{|\{\, SEG \mid TAG_{pred}(SEG) \in (\mathrm{B},\mathrm{I}) \,\}|}$$

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

The metrics at the sentence level are defined as follows:

$$Accuracy = \frac{|\{\, SENT \mid \forall\, SEG \in SENT,\ TAG_{pred}(SEG) = TAG_{ans}(SEG) \,\}|}{1150}$$

$$Correctable\ Rate = \frac{|\{\, SENT \mid \nexists\, SEG \in SENT,\ TAG_{pred}(SEG) = \mathrm{O}\ AND\ TAG_{ans}(SEG) \in (\mathrm{B},\mathrm{I}) \,\}|}{1150}$$

Accuracy and F1-score measure whether the models can find out segments with WOEs. *Correctable Rate* of sentences measures whether it is possible that the candidates of the correct word order can be generated by the WOE correction models. If a segment without WOEs is misjudged to be erroneous, the word order still has a chance to be kept by the WOE correction models. However, if a segment with WOEs is misjudged to be correct, words in the misjudged segment will not be reordered in the correction part because the error correction module is not triggered. A sentence is said to be "correctable" if no segments in it are misjudged as "correct". The ratio of the "correctable" sentences is considered as a metric at the sentence level.

Table 2 shows the performance of WOE detection. Five models are compared. We regard tagging all the segments with the labels B and O respectively as two baselines. Clearly, the recall at the segment level and the correctable rate at the sentence level are 1 by the all-tag-B baseline. However, its accuracy at the segment and the sentence levels are low. The all-tag-O baseline has better accuracy at the segment level than the all-tag-B baseline, but has very bad F1-score, i.e., 0. The proposed models are much better than the two baselines. Among the feature combinations, $f_{PMI}f_{Diff}f_{Tag}f_{B}$ show the best performance. The accuracy at the segment level is 0.834, and the correctable rate is 0.883. The best detection result will be sent for further correction.

Model	Segment				Sentence	
	Accuracy	Recall	Precision	F1-Score	Accuracy	Correctable Rate
Baseline (*all tag B*)	0.404	1.000	0.424	0.595	0.271	1.000
Baseline (*all tag O*)	0.576	0.000	0.000	0.000	0.074	0.074
$f_{PMI}f_{LM}f_{Tag}f_{B}$	0.830	0.781	0.802	0.791	0.787	0.862
$f_{PMI}f_{Diff}f_{Tag}f_{B}$	**0.834**	**0.795**	0.805	**0.800**	**0.788**	**0.883**
$f_{PMI}f_{Diff}f_{LM}f_{Tag}f_{B}$	0.831	0.769	**0.823**	0.795	0.777	0.850

Table 2: Performance of word ordering error detection

6 Correction of Word Ordering Errors

This section deals with generating and ranking candidates to correct WOEs. Two datasets, C_{ans} and C_{sys}, are explored in the experiments. We evaluate the optimal performance of the WOE correction models with the C_{ans} dataset, and evaluate WOE detection and correction together with the C_{sys} dataset.

6.1 Candidate Generation

Instead of direct permutation, we consider three strategies shown as follows to correct the error sentences. The complexity of generating candidates by permutation is O(n!). The complexity of using these three strategies decreases to O(n^2).

(1) Reorder single unit (R_{single})
R_{single} strategy reorders only one reordering unit (i.e., a word) to n-1 positions within a segment containing n words. Total $(n\text{-}1)^2$ candidates can be generated by this strategy. The following shows an example.

S: 今天 / 学校 / 去
(Today / school / go to)
A: 今天 / 去 / 学校
(Today / go to / school)

(2) Reorder bi-word ($R_{bi\text{-}word}$)
$R_{bi\text{-}word}$ is similar to R_{single}, but two reordering units are merged into a new word before reordering. Because n-1 bi-words can be generated in a segment and n-2 positions are available for each merged bi-word, $(n\text{-}1)(n\text{-}2)$ candidates are generated by $R_{bi\text{-}word}$. The following shows an example.

S: 早/就/一家/公司/找/我/工作
(before / already / one / company / employ / me / work)
A: 一家/公司/早/就/找/我/工作
(one / company / before / already / employ / me / work)

(3) Reorder tri-word ($R_{tri\text{-}word}$)
$R_{tri\text{-}word}$ works similarly to $R_{bi\text{-}word}$, but three reordering units are merged before reordering. Total $(n\text{-}2)(n\text{-}3)$ candidates are generated by $R_{tri\text{-}word}$. The following shows an example.

S: 我/需要/工作/的/经验/在/您/的/公司。
(I / need / working / (de) / experience / in / your / (de) / company.)
A: 我/需要/在/您/的/公司/工作/的/经验。
(I / need / in / your / (de) / company / working / (de) / experience.)

Table 3 shows the recall rate of each candidate generation strategy. With the C_{ans} dataset, correct word ordering can be generated for 85.8% of the original sentences by fusing R_{single}, $R_{bi\text{-}word}$ and $R_{tri\text{-}word}$. The candidates generated by using the C_{sys} dataset cover 69.7% of the correct word orderings. The difference would probably be due to the error propagation of word ordering error detection specified in Section 5.3. Furthermore, 6% of correct word orderings are unable to be generated by using the reordering units due to the word segmentation issue as mentioned in Section 5.1. We can also find that 72.3% of sentences with WOEs can be corrected by the R_{single} strategy using the C_{ans} dataset. It means most of the WOEs made by non-native Chinese learners can be corrected by moving only one word.

Strategy\Dataset	C_{ans}	C_{sys}
R_{single}	0.723	0.577
$R_{bi\text{-}word}$	0.365	0.308
$R_{tri\text{-}word}$	0.239	0.217
$R_{single} \cup R_{bi\text{-}word} \cup R_{tri\text{-}word}$	**0.858**	**0.697**

Table 3: Recall of candidate generation strategies

6.2 Candidate Ranking

We use Ranking SVM (Joachims, 2002) for candidates ranking. Because WOEs may produce abnormal POS sequence, POS bigrams and POS trigrams are considered as features for Ranking SVM. We

use a *k*-tuple feature vector for each candidate sentence, where *k* is the number of features. In each dimension, binary weight is assigned: 1 if the feature exists in a candidate, and 0 otherwise. Score for each candidate is assigned by a binary classifier: 1 if the candidate is the same as either of the annotated corrections, and 0 otherwise.

6.3 Results and Discussion

Mean Reciprocal Rank (MRR) defined below is used for performance evaluation. The reciprocal rank is the multiplicative inverse of the rank of the first correct answer. MRR is the mean of reciprocal rank for all sentences *S*, value from 0 to 1. The larger MRR means the correct answer more closes to the top ranking.

$$MRR = \frac{1}{|S|}\sum_{i=1}^{|S|}\frac{1}{rank_1}$$

Percentage of answers having rank 1 is another metric. Five-fold cross-validation is used for training and testing. In the C_{ans} and C_{sys} datasets, 182.03 and 184.48 candidates are proposed by the approach of fusing the results of R_{single}, $R_{bi\text{-}word}$, and $R_{tri\text{-}word}$ on the average. Experimental results are listed in Table 4. The proposed candidate ranking method achieves an MRR of 0.270 in the C_{ans} dataset. It means the correct candidates are ranked 3.7 on the average. In contrast, the MRR by using the C_{sys} dataset is 0.208. It means the correct candidates are ranked 4.8 on the average when error detection and correction are performed in pipelining.

Metric\Dataset	C_{ans}	C_{sys}
MRR	0.270	**0.208**
% of rank 1	0.195	**0.144**

Table 4: Performance of candidate ranking

There are some major types of errors shown as follows in WOE correction.

(1) Word ordering errors across segments
Section 5.1 mentions there are 31 sets of sentences (i.e., 2.7%) with WOEs across segments. Our algorithm cannot capture such kinds of sentences.

(2) Propagation errors from candidate generation
Table 3 shows the recall of word ordering error detection using the C_{ans} dataset is 0.858. Besides, 6% of sentences mentioned in Section 5.1 cannot be reordered to correct word ordering due to word segmentation issue.

(3) Limitation of our models
In the fused n-gram models, only one n-gram can be moved. It reduces the number of candidates to be generated, but some types of reorderings are missed. An example is shown as follows. The 2-gram 出生 / 于 (was born in) and the unigram 于 (on) have to be exchanged.

S：我 / 出生 / 于 / 1968 年 10 月 25 日 / 在 / 维也纳。
(I / was born / in / 25 October 1968 / on / Vienna.)
A：我 / 在 / 1968 年 10 月 25 日 / 出生 / 于 / 维也纳。
(I / on / 25 October 1968 / was born / in / Vienna.)

7 Conclusion

In this paper, we consider words as the reordering units in WOE detection and correction. Sentences are chunked into segments based on punctuation marks and the CRF technique is used to detect segments that possibly contain WOEs. The best error detection model achieves an accuracy of 0.834. Three reordering strategies are further proposed to generate candidates with correct word ordering and reduce the numerous number of candidates generated by permutation. If the segments containing WOEs are known, 85.8% of correct sentences can be generated by our approach. Finally, Ranking SVM orders the generated candidates based on POS bigrams and POS trigrams features, and achieves an MRR of 0.270 when all erroneous segments are given and an MRR of 0.208 when both detection and correction modules are considered.

Using words as the reordering unit reduces the cost to generate numerous candidates, but 6% of sentences are unable to reorder due to the word segmentation issue. How to balance the trade-off has to be investigated further. In the candidate ranking, selection of proper weights for POS bigram and trigram features may improve the ranking performance. Since the corpus of WOEs in Chinese is still in a limited size, expanding the related corpus for further research is also indispensable.

Acknowledgements

This research was partially supported by National Taiwan University and Ministry of Science and Technology, Taiwan under grants 103R890858, 101-2221-E-002-195-MY3 and 102-2221-E-002-103-MY3. We are also very thankful to the anonymous reviewers for their helpful comments to revise this paper.

References

Robert Dale, Ilya Anisimoff and George Narroway. 2012. HOO 2012: A Report on the Preposition and Determiner Error Correction Shared Task. In Proceedings of The 7th Workshop on the Innovative Use of NLP for Building Educational Applications, pages 54–62, Montre´al, Canada.

Robert Dale and Adam Kilgarriff. 2011. Helping Our Own: The HOO 2011 Pilot Shared Task. In *Proceedings of the 13th European Workshop on Natural Language Generation* (*ENLG*), pages 242–249, Nancy, France.

John DeNero and Jakob Uszkoreit. 2011. Inducing Sentence Structure from Parallel Corpora for Reordering. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Edinburgh, Scotland, UK.

Michel Galley and Christopher D. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu.

Michael Gamon, Claudia Leacock, Chris Brockett, William B. Dolan, Jianfeng Gao, Dmitriy Belenko, and Alexandre Klementiev. 2009. Using Statistical Techniques and Web Search to Correct ESL Errors. *CALICO Journal*, 26(3):491–511.

An-Ta Huang, Tsung-Ting Kuo, Ying-Chun Lai, and Shou-De Lin. 2010. Discovering Correction Rules for Auto Editing. *Computational Linguistics and Chinese Language Processing*, 15(3-4):219-236.

Chang-ning Huang and Hai Zhao. 2007. Chinese Word Segmentation: A Decade Review. *Journal of Chinese Information Processing*, 21(3):8-19.

Thorsten Joachims. 2002. Optimizing Search Engines using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133-142, Edmonton, Alberta, Canada.

John Lafferty, Andrew McCallum, and Fernando C.N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labelling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning* (*ICML 2001*), pages 282-289, San Francisco, CA, USA.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2014. *Automated Grammatical Error Detection for Language Learners*. 2nd Edition. Morgan and Claypool Publishers.

Jia-Na Lin. 2011. *Analysis on the Biased Errors of Word Order in Written Expression of Foreign Students*. Master Thesis. Soochow University.

Fang Liu, Meng Yang, Dekang Lin. 2010. *Chinese Web 5-gram Version 1*. Linguistic Data Consortium, Philadelphia. http://catalog.ldc.upenn.edu/LDC2010T06.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 Shared Task on Grammatical Error Correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12, Sofia, Bulgaria.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text Chunking Using Transformation-based Learning. In *Proceedings of Third Workshop on Very Large Corpora*. Pages 82-94.

Hendra Setiawan, Min-Yen Kan, Haizhou Li, and Philip Resnik. 2009. Topological Ordering of Function Words in Hierarchical Phrase-based Translation. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pages 324–332, Suntec, Singapore.

Li-Li Sun. 2011. *Comparison of Chinese and English Word Ordering and Suggestion of Chinese Teaching for Foreign Learners*. Master Thesis. Heilongjiang University.

Joachim Wagner, Jennifer Foster, and Josef van Genabith. 2007. A Comparative Evaluation of Deep and Shallow Approaches to the Automatic Detection of Common Grammatical Errors. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 112–121, Prague, Czech Republic.

Zhuo Wang. 2011. *A Study on the Teaching of Unique Syntactic Pattern in Modern Chinese for Native English-Speaking Students*. Master Thesis. Northeast Normal University.

Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. 2013. Chinese Spelling Check Evaluation at SIGHAN Bake-off 2013. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing* (SIGHAN-7), pages 35–42, Nagoya, Japan.

Chi-Hsin Yu and Hsin-Hsi Chen. 2012. Detecting Word Ordering Errors in Chinese Sentences for Learning Chinese as a Foreign Language. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 3003-3018, Mumbai, India.

Unsupervised Training Set Generation for Automatic Acquisition of Technical Terminology in Patents

Alex Judea[1] Hinrich Schütze[2] Sören Brügmann[3]
[1]Heidelberg Institute for Theoretical Studies, Heidelberg, Germany
[2]Center for Information and Language Processing, University of Munich, Germany
[3]Brügmann Software GmbH, Papenburg, Germany

Abstract

NLP methods for automatic information access to rich technological knowledge sources like patents are of great value. One important resource for accessing this knowledge is the technical terminology of the patent domain. In this paper, we address the problem of automatic terminology acquisition (ATA), i.e., the problem of automatically identifying all technical terms in a document. We analyze technical terminology in patents and define the concept of technical term based on the analysis. We present a novel method for labeling large amounts of high-quality training data for ATA in an unsupervised fashion. We train two ATA methods on this training data, a term candidate classifier and a conditional random field (CRF), and investigate the utility of different types of features. Finally, we show that our method of automatically generating training data is effective and the two ATA methods successfully generalize, considerably increasing recall while preserving high precision relative to a state-of-the-art baseline.

1 Introduction

A large part of our technological knowledge is encoded in patents. Methods for automatically finding information in patents and inferring information from patents are thus of great value. An important step in getting access to patent information is identification of technical terminology, i.e., finding the linguistic expressions that denote the technical concepts of a patent: the methods, processes, substances and objects that are part of the invention or modified by it. In the example "The present invention relates to a **charging apparatus** of a **bicycle dynamo**", the bolded compound nouns are the main content words and refer to specific technological concepts. We call such linguistic expressions *(technical) terms* or TERMS and their totality the *(technical) terminology* of a document or domain.

We address the task of *automatic terminology acquisition* (ATA), the task of finding technical TERMS in texts without reliance on existing resources that list TERMS of the domain. In contrast to this stands *automatic terminology recognition* (ATR), which we define as finding *known* TERMS and their variants (Jacquemin and Bourigault, 2003). ATA provides input to downstream components like automatic summarization, machine translation, ontology building, information extraction and retrieval. TERMS extracted by ATA can be semantically classified or mapped to entries in a semantic database (Krauthammer and Nenadic, 2004), but we focus on identifying them without further classification in this paper.

Our main contributions are as follows. (i) We present a method for automatically labeling large amounts of training data for ATA. (ii) We show that two types of statistical classifiers trained on this training data beat a state-of-the-art baseline, indicating that the automatic labeling is of high quality. (iii) We study different feature types for ATA and investigate how much they contribute to good performance. We investigate a semi-supervised setting in which features are selected based on a manually labeled evaluation set and a completely unsupervised setting where the feature selection is performed on an automatically produced set. (iv) Finally, we show that performance strongly depends on correct identification of the boundaries of TERMS and could be enhanced considerably by improving candidate identification.

The paper is organized as follows. Section 2 gives a definition of technical terminology and provides a brief analysis of TERMS in patents. Section 3 presents related work. Section 4 describes the architecture of our ATA system: preprocessing, linguistic filtering, automatic labeling of training data, feature selection and postprocessing. Section 5 reports evaluation results and analyzes selected features and errors. Section 6 presents our conclusions.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 290–300, Dublin, Ireland, August 23-29 2014.

2 Problem Description

Let $w_{1...k}$ be a sequence of words $w_1, w_2, \ldots, w_k$ and w_k a head noun. $w_{1...k}$ is a TERM of domain D iff (i) the head noun w_k is unmodified ($k = 1$) or (for $k > 1$) is modified by sequences of other nouns ("disk controller"), adjectives ("secondary controller") or present participles ("writing controller") and (ii) it denotes a concept specific to D.

(i) and (ii) describe the syntactic and semantic properties of a TERM, respectively. Part (i) restricts TERMS to parts of noun phrases. This is a reasonable restriction that covers most technical terms (Daille et al., 1996) and it has been frequently made in the computational terminology literature. We exclude comparatives and superlatives as modifying adjectives because they are rarely used attributively in patents and usually modify quantities or qualities of TERMS(e.g., "*higher* shunt currents"); in other words, only "positive" (base-form) adjectives are included in our definition. Note that the number of tokens per TERM is not restricted by the definition. Our approach aims to find TERMS of arbitrary length.

Part (ii) of the definition restricts TERMS to be specific to a domain D. We can set D to a general domain like 'electricity' and be on a par with many prior definitions (Ananiadou, 1994; Georgantopoulos and Piperidis, 2000; Zhang and Fang, 2010), but we can also set D to a narrow domain like 'emergency protective circuit arrangements' (IPC code[1] H02H).

Here, we choose the most general technical domain possible: the domain of all technical subjects. This is a good setting for many downstream tasks, e.g., information retrieval should benefit from a broad coverage of D. It also makes annotation easier: Non-experts can carry it out with good agreement (Section 5.1) because they simply look for all technical expressions.

The syntactic and semantic parts of our definition of TERM correspond to the concepts of *unithood* and *termhood*, respectively. Unithood is the degree to which a sequence of tokens is a linguistic unit; and termhood the degree to which a linguistic unit is a TERM of a domain (Kageura and Umino, 1996). Both aspects have to be covered by ATA systems.

Terms in Patents

In addition to traditional TERMS like simple nouns (1, "voltage"), modified nouns (2, "secondary arm") and nouns modified by prepositional phrases (3, "trajectory of the lever"), patents provide also coordinations (4, "constant and variable current") and complex constructions (5, "storage device storing a target temperature value which a battery is intended to reach").

For ATA, it seems advisable to exclude infrequent and complex nominal expressions from the definition of TERM, both from a terminological and a computational point of view. Most nominal expressions that are generally viewed as terms are single nouns, compound nouns, and nouns with an adjectival modifier (Daille et al., 1996); our syntactic definition covers these three types. Nominal expressions like (5) tend to be long; if we were to count such cases as TERMS, then it would be unclear where the TERM ends. When analyzing (5), our first take might be that there is a nucleus ("storage device") which is modified by a verbal phrase ("storing a target temperature value") and that the rest of the phrase is not part of the TERM. But it turns out that the *whole phrase* appears multiple times in its patent; it is a stable way of denoting a part of the invention. However, the underlying concept is also denoted by simpler constructions like the nucleus itself, or synonymous TERMS like "control circuit"; these simpler constructions are covered by our definition.

Coordinations like (4) mix multiple concepts (here, "constant current" and "variable current") without making this explicit on the surface. It is difficult to identify "constant current" as a potential TERM because it is non-contiguous and is only indicated by an adjective. Our treatment of coordinations in this paper is to only consider sequences satisfying the syntactic definition (i.e., "variable current") to be TERMS and discard other parts (i.e., "constant"). Of course, if both conjuncts are complete TERMS and satisfy the syntactic definition, both will be identified as TERMS.

Finally, prepositional phrases like (3) are rather infrequent compared to terms covered by our syntactic definition. They also tend to be highly ambiguous and the underlying concept is often expressed by terms covered by our definition ("lever trajectory").

3 Related Work

Previous work on ATA either employs *filtering* or *sequence models*. Filtering combines linguistic and statistical criteria for (i) *extracting a list of candidates* (typically word n-grams) based on simple linguistic criteria, (ii) *computing candidate statistics* and (iii) using ranking, classification or some other mechanism for *producing a pruned list of* TERMS *as output*. Because variation of the surface form of TERMS is limited,

[1] `wwwcms10.wipo.int/classifications/ipc/en/`

it makes sense to use word n-grams as the basis for candidate identification – even though there are cases that cannot be found this way, e.g., "constant current" or alternations like "pressure regulating valve" vs. "valve regulating pressure".

The main difference between ATA methods that rely on filtering is in how they accomplish the ranking/pruning of the candidate list. See Kageura and Umino (1996), Jacquemin (2001) and Pazienza et al. (2005) for an overview. In this paper, we accomplish this by training a statistical model to classify TERM candidates. We also run experiments with a sequence model. Our main innovation is that these models are trained on automatically labeled training data.

It is difficult to directly compare computational terminology systems because of differences in domain, language, application and task definition. As an example consider Takeuchi and Collier (2005) who report an F_1 of .742. However, their task definition includes assigning terms to pre-defined categories such as DNA and protein as opposed to simply identifying TERMS. In addition, terminologies in the biomedical and technological domains are different. In biomedicine, categories like DNA and protein dominate. For these TERMS, shape features are informative – in contrast to TERMS in patents. Another difference is that TERMS in patents tend to be long whereas DNA and proteins are often single-token abbreviations.

3.1 Training Data Collection

One of our main contributions is unsupervised training data generation (Section 4.3). Prior work has used automatically recognized training data for computational terminology, specifically for ATR (Craven and Kumlien, 1999; Hatzivassiloglou et al., 2001; Morgan et al., 2003; Zhang et al., 2010) in the biomedical domain. Given large precompiled TERM lists they search for occurrences of list elements, e.g., genes, in texts and use the occurrences they find as training examples. This is similar to *distant supervision* (Mintz et al., 2009) which also uses pre-existing resources such as gazetteers for, e.g., relation extraction.

In contrast, our method is applied to ATA for the technological domain and does not rely on precompiled resources – we make use of figure references, which are an inherent part of patents. Our method can be characterized as training data *identification*: we exploit given conditions in patents for our search of training data. In contrast, training data *recognition* methods need precompiled resources as input and search for instances of resource elements in texts.

3.2 Learning Algorithms and Features

Different learning algorithms and feature sets have been used for computational terminology. Foo and Merkel (2010) use Ripper (Cohen, 1995) with a variety of features to classify uni- and bigram TERM candidates. Hatzivassiloglou et al. (2001) compare C4.5 (Quinlan, 1993) and Naive Bayes (Duda and Hart, 1973). Zhang et al. (2010) acquire novel TERMS using CRFs and syntactic features. Takeuchi and Collier (2005) find that more training data results in higher F scores. Large training sets have the same positive effect in our experiments. Our approach has the added advantage that the training sets are generated completely automatically.

4 Approach

As discussed in the introduction, we address the problem of ATA. We use the abbreviation ATAS (automatic terminology acquisition system) to refer to our approach in general as well as to the specific implementation we evaluate in this paper.

ATAS consists of three parts: (i) training set generation, (ii) parameter selection and training of the TERM candidate classifier (ATAS-TC) and the CRF (ATAS-CRF) and (iii) identification of terminology in documents.

Processing in step (iii) is document by document because some of our features are document-based. ATAS takes a document as input and identifies all TERMS in the document, using the TERM candidate classifier or the CRF learned in (ii).

The TERM candidate classifier (ATAS-TC) decides on entire (multi-token) candidates while the CRF decides on single tokens. ATAS-TC heavily relies on candidate computation and its decisions are mutually independent, which is clearly incorrect. In contrast, ATAS-CRF is less dependent on candidate computation and models dependence of decisions correctly; but it lacks the more 'global' view of ATAS-TC on entire candidates. We want to investigate which approach is more suited for ATA.

In what follows we describe how we preprocess patents, the linguistic filters used to implement our syntactic definition of TERM, automatic labeling of training data (step (i) of ATAS), training of TERM candidate classifier and CRF (step (ii) of ATAS), features and feature selection.

4.1 Preprocessing

The preprocessing pipeline consists of the ANNIE tokenizer, OpenNLP sentence splitter, Mate POS tagger (Bohnet (2010), retrained for patents) and Mate lemmatizer. Preprocessing has a big influence on computational terminology because special domain text poses problems for off-the-shelf components. For example, patents tend to use common language words in rare functions or meanings, e.g., "said" as a de facto determiner in contexts such as "the structure of said component". Other problems are the use of special language words, e.g., substances like "triphenylphosphine" and acronyms like "AC". Such properties pose serious problems to POS taggers. Patent citations, acronyms and even product names can include punctuation, confusing sentence splitters. Chemical formulas may confuse tokenizers.

We adapted our POS tagger and sentence splitter for patent language to deal with unusual punctuation and POS tags – especially unusual POS tags of common-language words like "said". This adaptation involves training on a manually labeled training set of patent text and some other adjustments; e.g., we only allow the tag NN for the acronyms "AC", "DC" and "A/D".

4.2 Filter

We now describe how we find TERM candidates that satisfy the syntactic definition; recall that only (possibly modified) nouns can be TERMS (Section 2).

In general, candidate identification strategies using linguistic knowledge perform better. There are two different strategies of this type: (i) parsing the sentence, extracting nominal chunks from the parse and further processing the nominal chunks and (ii) POS tagging the sentence and extracting word sequences that satisfy a set of predefined POS patterns. Because many patent sentences are long and difficult to parse, we adopt the POS pattern approach in this paper. To this end, we define two simple POS-based rules for finding term candidates.[2]

PREMODS. This rule defines a modifier *sequence*. It matches a sequence of noun pre-modifiers: (JJ|"/"|VBG|RB|N(N|P))*.[3] We include RB because the POS tagger sometimes misclassifies JJ as RB. We include "/" because the tokenizer splits abbreviations containing it.

CANDIDATE. This rule defines a TERM candidate. It matches either a single noun or PREMODS followed by a noun: (PREMODS N(N|P)). The last noun must be longer than two characters. We add a flag indicating if the candidate comes before a figure reference. A figure reference consists of an optional keyword (e.g., "Figure", "Fig.") and a sequence of numbers and letters, optionally enclosed in parentheses.

We select the longest match in case of overlapping matches and the first longest match in case of overlapping matches of the same length.

These simple rules will find all TERMS – as well as many non-terms that we will train ATAS to identify – with two exceptions. First, due to POS errors some candidates are spurious. Second, unwanted modifiers may be part of candidates. E.g., the rules will only identify "same battery" as a candidate and not "battery". But only "battery" is a valid TERM. To address the latter, we manually compiled a stop list of 67 modifiers, mostly numerals ("first") and adjectives in anaphoric function ("above-mentioned"). These modifiers are removed from TERM candidates.

4.3 Automatic Labeling of Training Data

We view ATA as either a binary classification task where a TERM candidate classifier decides if a candidate is a TERM or not, or as a sequence labeling task where a CRF decides if a token (word) belongs to a TERM or not.

Large training sets are needed to train such models. Usually, these sets are produced by expensive human labeling. We present a method for generating high quality training data in an unsupervised way without the necessity of precompiled resources. In principle, our method can be used for any language for which machine-readable patents are available.

Our starting point is that patents typically contain figure references, i.e., pointers to drawings illustrating the invention or its parts. Consider the example: "...so that first **clamp-holding secondary arms** (1) ..." Here, the figure reference ("(1)") points to the illustration "Figure 1" and is preceded by the illustrated TERM ("clamp-holding secondary arms"). Illustrated TERMS may be concrete, as in this example, or abstract, e.g., a diagram illustrating properties of a method.

We call a TERM candidate that precedes a figure reference a *basic figure reference term candidate* (bFRTC). In a manual inspection of bFRTCs in 12 patents we found that almost 95% of bFRTCs were

[2] JJ, VBG, and RB are POS tags for positive adjectives, gerunds/present participles, and adverbs, respectively.
[3] '*' is the Kleene star, '?' denotes optionality, and '|' denotes alternation.

TERMS. Thus, bFRTCs can be used as positive training examples because they usually denote technical concepts; they have the advantage of being identifiable with high precision using simple patterns.

Once the bFRTCs have been identified, there is a simple way to further increase the size of the training data: we add all *extended FRTCs* (eFRTCs) to the training set, where we define an eFRTC as a TERM candidate whose suffix is a bFRTC. E.g., if we have identified "shunt current" as a bFRTC, then "AC shunt current" is an eFRTC. eFRTCs typically are hyponyms since the modifiers added at the beginning restrict the bFRTC to a more specific meaning. This kind of hyponymy is a special case of term derivation, a modification where a base term is further specified by prefixes (Daille et al., 1996). The strategy of identifying eFRTCs can also be applied to free word order languages because figure references tend to have a local and fixed occurrence pattern similar to English. We use the term FRTC to refer to both bFRTCs and eFRTCs.

We identify all FRTCs and add them as positive examples to the training set. We also add the 5% most frequent candidates as positive examples; most of them are FRTCs, so that this step usually adds few new training examples.

We label the following candidates as negative training examples: candidates appearing only once in a patent; patent citations; and measurements. Citations and measurements ("3 cm") are clear non-terms. We identify them using regular expressions. Many singletons are non-terms because they denote common language (i.e., nontechnical) concepts, e.g., "time". These heuristics for finding negative training examples are not applied to a candidate if it has the same head as a positive training example.

We exclude from the training set candidates that do not satisfy any positive or negative criteria.

4.4 Classifiers

We use the L2-regularized logistic regression of LIBLINEAR (Fan et al., 2008) as our TERM candidate classifier. We use LIBLINEAR's default normalization for continuous-valued attributes (normalization to range $[0, 1]$) and the default representation for categorical attributes. As LIBLINEAR cannot handle missing values, we replace them with their means and modes. We set the regularization parameter $c = 1$. Our sequence model is CRF++[4], order 1, with default parameters. The CRF features are adapted from the ATAS-TC features, e.g., TERM-level features (e.g., TFIDF) are propagated down to the individual tokens of the TERM. We also include word trigrams. We discretize numeric features to three values.

4.5 Features

We developed a set of 74 features for ATA. Some of these features are taken from the literature, some are specific to our approach and make use of the concept of FRTC and some exploit other properties of patents (e.g., the importance of the title and the claims in patents). A final group consists of other novel features that we designed in the course of developing our system. We now provide an overview. c refers to a TERM candidate.

Corpus and document statistics. This feature type captures termhood and unithood of c as well as the position of c's first occurrence in the document. We use a corpus of technical text C_T and a general language corpus C_G. For every $c \in C_T$ we collect the number of patents it appears in, its frequency and its FRTC frequency, i.e., the number of its occurrences that are FRTCs. Features that are intended to indicate termhood include simple frequencies and distributional characteristics (in C_T or in a single patent). Finally, we define a measure of frequency deviation (or 'keywordness') of $h(c)$, the head of c:

$$\text{bias}(h(c)) = \frac{f_{C_G}(h(c))}{|C_G|}|C_T| - f_{C_T}(h(c))$$

f_{C_G} (resp., f_{C_T}) are the frequencies in C_G (resp., C_T), $|X|$ is the sum of frequencies of all $x \in X$. $\text{bias}(h(c))$ measures the deviation between expected frequency of the head of c (estimated on C_G) and its actual frequency. The intuition here is that the frequency of a general language noun like "time" will be similar over text types, resulting in a lower bias.

Context. This feature type captures unigrams and bigrams adjacent to c as well as their POS tags.

Part-of-speech. This feature type captures the POS sequence of c.

A patent usually focuses on a narrow technological subdomain. As a result, many of its TERMS are semantically related to each other. We would like to include features that directly capture semantic similarity to other TERMS because a candidate that is semantically similar to several other already recognized TERMS is likely to be a TERM itself.

Our goal in this paper is to address ATA using simple and efficient methods. For this reason, we approximate semantic similarity using string similarity because a subset of semantically similar terms are

[4] `crfpp.googlecode.com`

	$\mathcal{T}^{\text{u}}_{\text{tdg}}$	$\mathcal{T}^{\text{l}}_{\text{test}}$	$\mathcal{T}^{\text{l}}_{\text{dev}}$	$\mathcal{T}^{\text{u}}_{\text{sel}}$
patents	365	5	11	25
word tokens	3,422,131	50,007	74,000	152,715
word types	292,994	3711	7391	4141
bFRTCs	119,316	1264	2558	6503
FRTCs	240,240	2371	4942	10,110
candidates	353,238	8836	13,099	27,164
TERMS		3814	7220	

Table 1: Data set statistics

	P	R	F_1	description
1	.704	.797	.748	mean string similarity of c and FRTCs
2	.712	.832	.767	frequency of c as an FRTC in C_T
3	.694	.887	.779	TFIDF of c
4	.703	.888	.784	is c uppercase?
5	.708	.893	.790	is c followed by a figure reference?
6	.710	.896	.792	TFIDF of $h(c)$
7	.711	.895	.793	frequency of $h(c)$ as an FRTC in C_T
8	.718	.892	.795	bias($h(c)$)
9	.720	.891	.797	# sentences with FRTCs that c occurs in
10	.720	.893	.797	C-value of c
11	.721	.893	.798	frequency of $h(c)$ in C_G

Table 2: Features selected on $\mathcal{T}^{\text{l}}_{\text{dev}}$ (setting S). c: TERM candidate. $h(c)$: head of c

also similar on the surface. E.g., the semantic similarity between "AC power supply source" and "AC supply source" also manifests itself as string similarity.

String similarity. When designing a similarity measure, we wanted it to satisfy the following criteria: (i) more words in common should result in higher scores and (ii) words in common *towards the end* of the two strings should be weighted higher than words in common at the beginning. The motivation for (ii) is that candidates differing only in initial modifiers are often cohyponyms and highly related; conversely, candidates with different heads are often not related.

To implement this, we represent a candidate c as a vector $\vec{c}$ in $|V|$-dimensional space where V is the vocabulary. $\vec{c}_i$ is set to the position of word w_i in c if it occurs and 0 otherwise. The string similarity between c and c' is then defined as the cosine of $\vec{c}$ and $\vec{c'}$. Example: for "AC power supply source" and "AC supply source", we get the vectors $(1, 2, 3, 4)$ and $(1, 0, 2, 3)$ and the cosine .927; comparing the first string with "AC power supply" with the vector $(1, 2, 3, 0)$ we get the cosine .683.

Features in our initial set of 74 that make use of this semantic similarity are: maximum similarity of c to any FRTC, average similarity of c to all FRTCs in the patent and similarity of c to the rightmost TERM candidate in the title.

Frantzi and Ananiadou (1997) define *C-value(c)* as:

$$\text{C-value}(c) = \log_2 |c|(f(c) - \frac{1}{|T_c|}\sum_{b \in T_c} f(b))$$

where T_c is the set of TERM candidates containing c and f is frequency in C_T. C-value is high for TERM candidates that are frequent and occur as parts of many other TERM candidates – this is a good indicator of termhood.

5 Experiments and Evaluation

5.1 Data Sets

We hired three students with a bachelor degree in computer science to annotate 16 patents. The test set $\mathcal{T}^{\text{l}}_{\text{test}}$ consists of 5 patents annotated by all three students. We used majority voting to produce the final gold annotations. The devset $\mathcal{T}^{\text{l}}_{\text{dev}}$ consists of the remaining 11 patents. Each $\mathcal{T}^{\text{l}}_{\text{dev}}$ sentence was annotated by one student.

Inter-annotator agreement on $\mathcal{T}^{\text{l}}_{\text{test}}$ was .76 (Fleiss' κ). Most disagreements concern *modifiers* or *common nouns* (e.g., the TERM "battery" was often not annotated). More extensive training of the annotators should reduce these problems considerably.

As unlabeled data we randomly selected 390 technology patents. We use 365 as $\mathcal{T}^{\text{u}}_{\text{tdg}}$ for training data generation and 25 as $\mathcal{T}^{\text{u}}_{\text{sel}}$ for unsupervised feature selection. We made sure the 390 documents are not

in $\mathcal{T}^{\mathrm{l}}_{\mathrm{dev}}$ and $\mathcal{T}^{\mathrm{l}}_{\mathrm{test}}$. We excluded chemical patents because standard preprocessing components often fail for chemical formulas. Table 1 gives data set statistics.

As our technical corpus C_T we use $\mathcal{T}^{\mathrm{u}}_{\mathrm{tdg}}$ and as our general corpus C_G all nouns in the 2000 most frequent English words from Project Gutenberg[5]. This list contains many general nouns which also appear in patents (e.g., "time") without containing many technical terms (e.g., "battery"); this way, C_T and C_G give us a good contrast between technical and non-technical vocabularies (cf. Section 4.5).

One obstacle to comparing systems for ATA in the technical domain is the lack of publicly available evaluation benchmarks. We are making our data sets and the annotation guidelines available[6].

5.2 Baselines

We define the *FRTC baseline* as the system that labels all FRTCs and only FRTCs as TERMS. Almost all FRTCs are TERMS, but many TERMS are not FRTCs; thus, the FRTC baseline has high precision and low recall. Our goal is to preserve high precision while considerably increasing recall, or to generalize well from FRTCs to other TERMS.

Our state of the art baseline is Z-CRF, a reimplementation of the CRF described in (Zhang et al., 2010). Its feature representation includes POS tags, unigrams, bigrams and syntactic information, e.g., the number of times a particular token is used in a syntactic function like subject in the training set. Syntactic information is extracted with Mate (Bohnet, 2010). Z-CRF is trained on $\mathcal{T}^{\mathrm{u}}_{\mathrm{tdg}}$, just as ATAS.

Our last baseline is the well-known C-value (Frantzi and Ananiadou, 1997). Like our first baseline, it needs no training data. In contrast to our first baseline, it was specifically designed for terminology acquisition. It combines observations about statistical and linguistic properties of TERMS, i.e., a candidate is preferred as a term if it is long and frequently appears as substring of other candidates. Following Frantzi and Ananiadou (1997) we regard a candidate as TERM if its C-value it not zero; unlike them, we do not restrict the length of TERMS because the computation of long terms did not pose computational problems for us.

5.3 Evaluation Setup

We evaluate ATAS using precision, recall and F_1. Evaluation is based on candidate tokens (as opposed to candidate types or word tokens); e.g., each instance of a candidate TERM that is incorrectly classified as a TERM is a false positive. Evaluation is strict in the sense that a TERM is counted as a false positive if there is a single token that is added or missed.

We evaluate ATAS in two settings. In the system (S) setting, the ATAS pipeline described in Section 4 (ATAS-TC or ATAS-CRF) is used to identify TERM candidates. This is the real-world setting since errors in TERM candidate identification – misplaced boundaries, missing candidates, etc. – are a major source of error in ATA.

We would also like to evaluate candidate classification on *gold boundaries* (manually verified boundaries of TERM candidates); this allows us to quantify by how much performance can be improved if candidate identification is perfect. However, since gold boundary annotation is expensive, we instead approximated it: (i) We run automatic TERM candidate identification. (ii) We remove all TERM candidates that overlap with gold (manually annotated) TERMS. (iii) The set of gold TERM candidates is then the union of all remaining automatically identified candidates and the manually annotated TERMS.

In the gold boundary (G) setting, we provide these gold TERM candidates to the ATAS pipeline. This allows us to evaluate the performance of TERM/non-term classification separately from TERM candidate identification.

5.4 Feature Selection

For our feature set of 74, we perform forward feature selection for the TERM candidate classifier by selecting the feature in each step that maximizes system F_1. We perform feature selection (i) on the manually labeled set $\mathcal{T}^{\mathrm{l}}_{\mathrm{dev}}$ (to gauge performance for an optimal or close-to-optimal feature set) and (ii) on the automatically labeled set $\mathcal{T}^{\mathrm{u}}_{\mathrm{sel}}$ (to gauge the performance in a completely unsupervised setting). In the following we explain both settings in more detail.

Table 2 gives the features selected in **supervised feature selection**, i.e., when features are optimized on $\mathcal{T}^{\mathrm{l}}_{\mathrm{dev}}$. Precision remains stable, except for a drop on line 3. Recall rises steadily from .797 to .893. F_1 increases from .748 to .798.

The best feature (line 1) is the mean string similarity of a TERM candidate c to all FRTCs in a document (Section 4.5). Together with the next best feature (frequency of c as an FRTC in C_T) and feature 5 (is

[5] en.wiktionary.org/wiki/Wiktionary:Frequency_lists/PG/2006/04/1-10000

[6] h-its.org/english/research/nlp/download/terminology.php

			ATAS-TC				ATAS-CRF				Baselines					
			S-SEL		U-SEL		S-SEL		U-SEL		Z-CRF		C-value		FRTC	
			S	G	S	G	S	G	S	G	S	G	S	G	S	G
$\mathcal{T}^{l}_{dev}$	1	P	.721	.838	.690	.796	.732	.844	.727	.854	.867	.891	.384	.749	.839	1.000
	2	R	.893	.892	.825	.818	.815	.699	.755	.679	.563	.607	.292	.355	.344	.353
	3	F_1	.798	.864	.752	.807	.771	.765	.741	.756	.683*	.722*	.314*	.471*	.488*	.522*
$\mathcal{T}^{l}_{test}$	4	P	.696	.753	.627	.692	.774	.832	.664	.745	.813	.840	.388	.726	.864	1.000
	5	R	.850	.853	.764	.764	.791	.743	.644	.625	.516	.559	.320	.410	.286	.302
	6	F_1	.765	.800	.689	.728	.783	.785	.654	.680	.631*	.674*	.350*	.519*	.430*	.465*

Table 3: System (S) and gold boundary (G) results with supervised (S-SEL) and unsupervised (U-SEL) feature selection. *: significantly lower than corresponding ATAS-TC and ATAS-CRF scores.

c followed by a figure reference?) this supports our intuition for using FRTCs for automatic training set generation because they are indeed strong indicators for termness. Additionally, feature 9 indicates that candidates occurring often with FRTCs in sentences are probably TERMS. Feature 4 (is c uppercase?) is selected because uppercase TERM candidates are often abbreviations and TERMS.

Feature 3 (TFIDF of c) hurts precision, but increases recall, resulting in increased F_1. This feature models the hypothesis that a TERM is frequent in some patents but does not occur in many patents. Patent writers often invent novel TERMS rather than using standard ones to make finding a patent hard. Thus, a TERM candidate that occurs often in a few patents could be such an obfuscating TERM.

TFIDF is low for TERMS with small term frequency. Features 6 (TFIDF of $h(c)$) and 10 (C-value of c) can help correctly identify such TERM candidates as TERMS.

Features 8 and 11 incorporate information from the general purpose corpus C_G. Feature 8 contrasts the frequency of c in C_G with its frequency in C_T – frequencies of TERMS are higher in C_T, frequencies of non-terms are similar in both corpora. Feature 11 is complementary to this. It makes it more probable that c is a non-term if its head appears more often in C_G. Additionally, string similarity with the patent's title is an effective feature.

Unsupervised feature selection, i.e., selection on $\mathcal{T}^{u}_{sel}$, selected seven features that are similar to those selected by supervised selection and that we will discuss now. The best unsupervised feature (*maximum* string similarity, 1) and the best supervised feature (*mean* string similarity) both capture partial string overlap of c and FRTCs. For similar reasons, the feature "string similarity of c and rightmost NP in patent title" (2) – which exploits the importance of the title in analogy to the importance of figure references – is selected.

Other selected features (relative patent frequency of c and its head (3, 4), number of patent sentences in which c occurs with FRTCs (5), patent frequency of $c = 1?(6)$) are also similar to the features selected in the supervised setting. They capture frequency distributions of c. However, while many features in the supervised setting capture distributions of c in C_T, in the unsupervised setting, distributions of c in the patent are more important. The reason may be that C_T-based features (which use all technical text as opposed to the relevant patent in question) are harder to recognize as good predictors if the set used for selection is automatically labeled and hence noisier.

The last unsupervised feature captures the length of c in tokens (7). Manual inspection revealed that on average TERMS have more tokens than non-terms (1.9 vs. 1.3).

5.5 ATAS Results

Table 3 gives evaluation results for ATA on $\mathcal{T}^{l}_{dev}$ and $\mathcal{T}^{l}_{test}$. We report results for the ATAS versions (ATAS-TC, ATAS-CRF) and for the baselines (Z-CRF, C-value, FRTC) as well as for using supervised (S-SEL) and unsupervised feature selection (U-SEL) in system setting (S) and gold boundary setting (G).

Differences in F_1 between ATAS and baselines (marked with a †) are significant at $p < .01$.[7] If not stated otherwise, numbers below are for the system setting (S).

We note that F_1 of the ATAS versions is consistently and considerably better than all baselines in all settings. E.g., line 6 shows system F_1 on $\mathcal{T}^{l}_{test}$ of ATAS-TC (.765 for S-SEL, .689 for U-SEL) and ATAS-CRF (.783 for S-SEL, .654 for U-SEL) compared to Z-CRF (.631), FRTC (.430), and the C-value baseline (.350) . The better results mainly come from higher recall (except for C-value, which is also beaten in precision). In general, precision of the baselines is higher, but recall much smaller than for ATAS. This shows that (i) statistical classifiers can be successfully trained for ATA using our method

[7]We use approximate randomization (Yeh, 2000) for all significance tests in this paper.

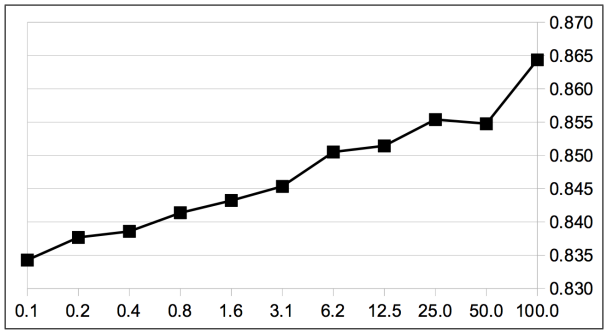


Figure 1: System F_1 as a function of training set size (in percent) in setting G.

for automatically generating training data and (ii) these classifiers beat a state-of-the-art system in both S-SEL and U-SEL settings.

Comparing S-SEL and U-SEL shows that precision and recall for U-SEL are lower than for S-SEL. For instance, F_1 of ATAS-TC on $\mathcal{T}^{\text{l}}_{\text{test}}$ is .765 for S-SEL and .689 for U-SEL; F_1 of ATAS-CRF is .783 for S-SEL and .654 for U-SEL (line 6). In general, we note a bigger drop in recall than in precision, indicating that U-SEL does not generalize as well as S-SEL. However, the U-SEL numbers are significantly better than the Z-CRF FRTC, and C-value baselines.

When comparing ATAS-TC with ATAS-CRF we note that ATAS-CRF consistently has higher precision and lower recall. In most cases, ATAS-TC has considerably higher recall, leading to higher F_1. This is not surprising given that feature selection was performed for ATAS-TC. Nevertheless, ATAS-CRF can compete with ATAS-TC in terms of F_1. Furthermore, ATAS-CRF produces more stable results because it shows less variance in F_1 across settings.

Comparing S and G scores shows that knowing exact boundaries has a great impact on results, especially on precision; looking at S-SEL numbers in line 4 in Table 3, precision for ATAS-TC (resp., ATAS-CRF) is .696 in S vs. .753 in G (resp., .774 in S vs. .832 in G). Similar differences also hold for U-SEL numbers. In general, ATAS-TC profits more from knowing exact boundaries than ATAS-CRF. This leads us to the conclusion that the linguistic filter would greatly benefit from a (statistical) measure of unithood. Note that this also holds for the baselines; deciding about the termness of gold boundary candidates seems to be easier, especially for C-value.

All observations hold for $\mathcal{T}^{\text{l}}_{\text{dev}}$ and $\mathcal{T}^{\text{l}}_{\text{test}}$. However, numbers are higher for $\mathcal{T}^{\text{l}}_{\text{dev}}$ because the ratio of FRTCs to candidates is higher than for $\mathcal{T}^{\text{l}}_{\text{test}}$ (38% vs. 27%) which improves classification performance on $\mathcal{T}^{\text{l}}_{\text{dev}}$ – this holds for ATAS as well as for the baselines.

To investigate the quality of the extracted training data, consider Figure 1. It shows F_1 in setting G as a function of training set size in percent of the total training set $\mathcal{T}^{\text{u}}_{\text{tdg}}$. For each evaluation point, we randomly add training examples from the full set. F_1 starts at .834 for 0.1% of training data (344 training examples) and rises to .864 for 100% (353,238 examples), with a small drop at 50%. Note that 1000 examples roughly correspond to one annotated patent. The main results of this experiment are that (i) a modest amount of automatically labeled training data gives good performance and (ii) the more automatically labeled data the better. The last point is not a trivial finding, given that training data was generated automatically. The logarithmic graph shows a nearly linear increase in F_1 for each doubling of the training data.

To further investigate the quality of the generated training data, we compared automatically and manually produced training examples. We compare results for 13238 manual and 13238 automatic labels (setting G, ATAS-TC). We get precision and recall of .811 and .805 for manual and .762 and .850 for automatic annotations, resulting in similar F1 scores: .808 vs. .804 for manual and automatic annotations, respectively. We believe that the differences in recall are an artifact of the randomization we performed before removing automatic training samples. Manual labels are entire patents; in contrast, automatic labels come from all patents in the training set, leaving us with a more diverse set than the manual version.

5.6 Error Analysis

We found two major types of false negatives. First, infrequent TERMS are problematic. It is hard to judge termness when having limited information about a candidate, especially if it appears only once or twice in a document. Second, POS errors prevent the system from finding some candidates; e.g., the noun "current" is frequently mistagged as adjective. Incorrect POS tags also lead to incorrect boundaries.

We found four major types of false positives. First, incorrect modifiers lead to partially incorrect TERMS. 27% of false positives are of this type. Second, incorrectly recognized figure references cause incorrect system decisions; e.g., our patterns incorrectly parse an expression like "value *PBA*" as a figure reference even though it is instead a named output of a component. Third, very frequent non-terms are commonly classified as TERMS. Almost all frequent candidates are TERMS, so that the TERM candidate classifier has difficulty correctly identifying the exceptions from this pattern.

Finally, if a candidate is a TERM in one context it may be a non-term in another. A good example for this are general single token TERMS like "apparatus". Before figure references they are TERMS, e.g., "one preferred form of apparatus 22". In such cases the figure reference serves as a disambiguator. However, in other positions they are non-terms, e.g., "They include braces, collars, splints and other similar apparatus".

6 Conclusion and Future Work

This paper introduces a method for ATA with two novel aspects: (i) new powerful features for ATA and (ii) a procedure for generating an ATA training set in an unsupervised fashion. The training set generation method produces high quality training data, even when compared to manual annotations. It is language-independent: It can be applied to patents in any language if the definition of TERM candidates is modified for the target language. It is also domain-independent: it can be applied to patents of any domain. The training data can be successfully used to train ATA models, both TERM candidate classification as well as CRF models. Even in a completely unsupervised setting the models outperform a state-of-the-art baseline. We found that using more automatically labeled training data and using better TERM boundaries results in better performance.

In future work, we plan to incorporate TERM variation patterns (Daille et al., 1996; Jacquemin, 2001) in the expansion process to decrease the number of FNs and increase recall. We would also like to improve the terminology identification module because we found that incorrect identified boundaries affect performance greatly.

Finally, we are planning to extend our approach to languages other than English. Our methods are language-independent to the extent that a body of patents exists for many common languages. Since we generate the training set automatically, all we need to do to cover another language is to adapt the linguistic filters for candidate identification.

Acknowledgments. This work was supported by the European Union (Project Topas, FP7-SME-2011 286639) and by SPP 1335 *Scalable Visual Analytics* of Deutsche Forschungsgemeinschaft (DFG grant SCHU 2246/8-2). We would like to thank the anonymous reviewers for their helpful comments and suggestions, and Bianca and Luca for their support.

References

Sophia Ananiadou. 1994. A methodology for automatic term recognition. In *Proceedings of the 15th conference on Computational linguistics - Volume 2*, COLING '94, pages 1034–1038.

Bernd Bohnet. 2010. Top Accuracy and Fast Dependency Parsing is not a Contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 89–97, Beijing, China, August.

William W. Cohen. 1995. Fast effective rule induction. In *Twelft International Conference on Machine Learning (ML95)*, pages 115–123.

Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In Thomas Lengauer, Reinhard Schneider, Peer Bork, Douglas L. Brutlag, Janice I. Glasgow, Hans-Werner Mewes, and Ralf Zimmer, editors, *ISMB*, pages 77–86. AAAI.

Béatrice Daille, Benoît Habert, Christian Jacquemin, and Jean Royauté. 1996. Empirical Observation of Term Variations and Principles for their Description. *Terminology*, 3(2):197–258.

Richard O. Duda and Peter E. Hart. 1973. *Pattern Classification and Scene Analysis*. John Wiley & Sons Inc, 1 edition.

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

Jody Foo and Magnus Merkel. 2010. Using machine learning to perform automatic term recognition. In *Proceedings of the LREC 2010 Workshop on Methods for automatic acquisition of Language Resources and their Evaluation Methods*, pages 49–54.

Katerina T. Frantzi and Sophia Ananiadou. 1997. Automatic Term Recognition using Contextual Cues. In *Proceedings of 3rd DELOS Workshop*, Zurich, Switzerland.

Byron Georgantopoulos and Stelios Piperidis. 2000. Term-based Identification of sentences for Text Summarisation. In *Proceedings of Second International Conference on Language Resources and Evaluation (LREC2000)*, pages 1067–1070, Athens, Greece.

Vasileios Hatzivassiloglou, Pablo Ariel Dubou, and Andrey Rzhetsky. 2001. Disambiguating proteins, genes, and rna in text: a machine learning approach. In *ISMB (Supplement of Bioinformatics)*, pages 97–106.

Christian Jacquemin and Didier Bourigault. 2003. Term extraction and automatic indexing. In Ruslan Mitkov, editor, *The Oxford Handbook of Computational Linguistics*, chapter 33. Oxford University Press.

Christian Jacquemin. 2001. *Spotting and Discovering Terms Through Natural Language Processing*. MIT Press, April.

Kyo Kageura and Bin Umino. 1996. Methods of automatic term recognition: A review. *Terminology*, 3(2):259–289.

Michael Krauthammer and Goran Nenadic. 2004. Term identification in the biomedical literature. *Journal of Biomedical Informatics*, 37(6):512–526, December.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Alex Morgan, Lynette Hirschman, Alexander Yeh, and Marc Colosimo. 2003. Gene Name Extraction Using FlyBase Resources. In *Proceedings of ACL 2003 Workshop on Natural Language Processing in Biomedicine*, pages 1–8, Sapporo, Japan.

Maria Teresa Pazienza, Marco Pennacchiotti, Michele Vindigni, and Fabio Massimo Zanzotto. 2005. Ai/nlp technologies applied to spacecraft mission design. In *Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence*, IEA/AIE'2005, pages 239–248, London, UK, UK.

John Ross Quinlan. 1993. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Koichi Takeuchi and Nigel Collier. 2005. Bio-medical entity extraction using support vector machines. *Artificial Intelligence in Medicine*, 33(2):125–137, February.

Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics - Volume 2*, COLING '00, pages 947–953, Stroudsburg, PA, USA.

Xing Zhang and Alex Chengyu Fang. 2010. An ATE system based on probabilistic relations between terms and syntactic functions. In *10th International Conference on Statistical Analysis of Textual Data*, pages 1135–1143, Sapienza, Italy, June.

Xing Zhang, Yan Song, and Alex Chengyu Fang. 2010. How well conditional random fields can be used in novel term recognition. In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, pages 583–592, Tohoku University, Sendai, Japan, November.

A Data Driven Approach for Person Name Disambiguation in Web Search Results

Agustín D. Delgado[1], **Raquel Martínez**[1], **Víctor Fresno**[1], **Soto Montalvo**[2]
[1]Universidad Nacional de Educación a Distancia (UNED), Madrid, Spain
[2]Universidad Rey Juan Carlos (URJC), Móstoles, Spain
[1]{agustin.delgado,raquel,vfresno}@lsi.uned.es, [2]soto.montalvo@urjc.es

Abstract

This paper presents an unsupervised approach for the task of clustering the results of a search engine when the query is a person name shared by different individuals. We propose an algorithm that calculates the number of clusters and establishes the groups of web pages according to the different individuals without the need to any training data or predefined thresholds, as the successful state of the art systems do. In addition, most of those systems do not deal with social media web pages and their performance could fail in a real scenario. In this paper we also propose a heuristic method for the treatment of social networking profiles. Our approach is compared with four gold standard collections for this task obtaining really competitive results, comparable to those obtained by some approaches with supervision.

1 Introduction

Resolving the ambiguity of person names in web search results is a challenging problem becoming an area of interest for Natural Language Processing (NLP) and Information Retrieval (IR) communities. This task can be defined informally as follows: given a query of a person name in addition to the results of a search engine for that query, the goal is to cluster the resultant web pages according to the different individuals they refer to. Thus, the challenge of this task is estimating the number of different individuals and grouping the pages of the same individual in the same cluster. The difficulty of this task resides in the fact that a single person name can be shared by many people: according to the U.S. Census Bureau, 90000 different names are shared by 100 million people (Artiles et al., 2007). This problem has had an impact in the Internet and that is why several vertical search engines specialized in web people search have appeared in the last years, e.g. `spokeo.com` or `123people.com`. This task should not be mixed up with *entity linking* (EL), which goal is to link name mentions of entities in a document collection to entities in a reference knowledge base (typically Wikipedia), or to detect new entities.

The main difficulties of clustering web pages referring to the same individual come from their possible heterogeneous nature. For example, some pages may be professional sites, while others may be blogs containing personal information. In addition, the popularity of social networking services makes the search engine usually returns several social profiles belonging to different individuals sharing the same name, as much from the same social networking service as from different services. These social pages often introduce noisy information and make the state of the art algorithms break down (Berendsen et al., 2012). Due to these problems, the users have to refine the queries with additional terms. This task gets harder when the person name is shared by a celebrity or a historical figure, because the results of the search engines are dominated by that individual, making the search of information about other individuals more difficult.

WePS[1] (Web People Search) evaluation campaigns proposed this task in a web searching scenario providing several corpora for evaluating the results of their participants, particularly WePS-1, WePS-2 and WePS-3 campaigns. This framework allows our approach to be compared with the state of the art

[1]http://nlp.uned.es/weps/

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 301–310, Dublin, Ireland, August 23-29 2014.

systems. We also evaluate our system with ECIR2012 corpus[2], a data set that includes social networking profiles, providing a more real scenario for this task.

The most successful state of the art systems have addressed this problem with some kind of supervision. This work proposes a data-driven method for this task with the aim of eliminating the elements of human involvement in the process as much as possible. The main contribution of this work is a new unsupervised approach for resolving person name ambiguity of web search results based on the use of capitalized n-grams. In our approach the decision if two web pages have to be grouped only depends on the information of both pages. In addition, we also propose a heuristic method for the treatment of social media profile web pages in this context.

The paper is organized as follows: in Section 2 we discuss related work; Section 3 details the way we represent the web pages, the algorithm and the heuristic for social pages; in Section 4 we describe the collections used for evaluating our method and we show our results making a comparison with other systems; the paper ends with some conclusions and future work in Section 5.

2 Related Work

Several approaches have been proposed for clustering search results for a person name query. The main differences among all of them are the features they use to represent the web pages and the clustering algorithm. However, the most successful of them have in common that they use some kind of supervision: learning thresholds and/or fixing manually the value of some parameters according to training data.

Regarding the way of representing a web page, the most popular features used by the most successful state of the art approaches are Name Entities (NE) and Bag of Words (BoW) weighted by TF-IDF function. In addition to such features, the systems usually use other kind of information. Top systems from WePS-1 and WePS-2 campaigns, CU_COMSEM (Chen and Martin, 2007) and PolyUHK (Chen et al., 2009), distinguish several kind of tokens according to different schemes (URL tokens, title tokens, ...) and build a feature vector for each sort of tokens, using also information based on the noun phrases appearing in the documents. PolyUHK also adds pattern techniques, attribute extraction and detection when a web page is written in a formal way. A more recent system, HAC_Topic (Liu et al., 2011), also uses BoW of local and global terms weighted by TF-IDF. It adds a topic capturing method to create a Hit List of shared high weighted tokens for each cluster obtaining better results than WePS-1 participants. On the other hand, the WePS-3 best system, YHBJ (Chong and Shi, 2010), uses information extracted manually from Wikipedia adding to BoW and NE weighted by TF-IDF.

Regarding the clustering algorithms, looking at WePS campaigns results, the top ranked systems have in common the use of the Hierarchical Agglomerative Clustering algorithm (HAC) described in (Manning et al., 2008). Different versions of this algorithm were used by (Chen and Martin, 2007; Chen et al., 2009; Elmacioglu et al., 2007; Liu et al., 2011; Balog et al., 2009; Chong and Shi, 2010).

(Berendsen et al., 2012) presented another gold standard for this task, ECIR2012, composed by Dutch person names and social media profile web pages. The system of the authors, UvA, distinguishes the web pages between social ones and non social ones, clusters each group separately and then combines both clustering solutions. They represent each web page as a BoW vector weighted by TF-IDF, and use cosine similarity for comparing web pages. They use HAC algorithm for clustering non social web pages, while use a "one in one" policy for the social ones. Finally, they mix both groups by means of an algorithm which penalizes clusters with social webs or simply taking the union of both clustering solutions. They perform a partial parameter sweep on the WePS-2 data set to fix the clustering thresholds, while explore combinations of other system parameters.

The only system that does not use training data, DAEDALUS (Lana-Serrano et al., 2010), which uses k-Medoids, got poor results in WePS-3 campaign. In short, the successful state of the art systems need some kind of supervised learning using training data or fixing parameters manually. In this paper we explore and propose an approach to address this problem by means of data-driven techniques without the use of any kind of supervision.

[2] http://ilps.science.uva.nl/resources/ecir2012rdwps

3 Proposed Approach

We distinguish two main phases in this clustering task: web page representation (Sections 3.1 and 3.2) and web page grouping (Sections 3.3 and 3.4). In addition, we propose an heuristic to deal with social profiles web pages (Section 3.5).

3.1 Feature Selection

The aim of this phase is to extract relevant information that could identify an individual. We assume the main following hypotheses:

(i) Capitalized n-grams co-occurrence could be a reliable way for deciding when two web pages refer the same individual. Capitalized n-grams usually are Named Entities (organizations and company names, locations or other person names related with the individual) or information not detected by some NE recognizers as for example, the title of books, films, TV shows, and so on. In a previous study with WePS-1 training corpus using the Stanford NER[3] to annotate NE, we detected that only 55.78% of the capitalized tokens were annotated as NE or components of a NE by the NER tool. So the use of capitalized tokens allows increase the number of features compared to the use of only NE. We also compared the n-gram representation with capitalized tokens and with NE. We found that 30.97% of the 3-grams of capitalized tokens were also NE 3-grams, and 25.64% of the 4-grams of capitalized tokens were also NE 4-grams. So even in the case of n-grams the use of capitalized tokens increases the number of features compared to the use of only NE. Table 1 shows the differences in performance when using n-grams representation with NE or with capitalized tokens.

(ii) If two web pages share capitalized n-grams, the higher is the value of n, the more probable the two web pages refer to the same individual. In this case we define "long enough n-grams" as those compose by at least 3 capitalized tokens.

Thus, a web page W is initially represented as the sequence of tokens starting in uppercase, in the order as they appear in the web page. In each step of the algorithm, a web page W will be represented by its long enough n-grams, taking different values for n, as we describe in Section 3.4. Notice that some web pages could not be represented with this proposal because all their content was written in lowercase. In the case of the collections that we describe in Section 4.1, 0.63% of the web pages are not represented for this reason.

3.2 Weighting Functions

We test the well known TF and TF-IDF functions, and z-score (Andrade and Medina, 1998). The z-score of a n-gram a in a web page W_i is defined as follows: $z\text{-}score(a, W_i) = \frac{TF(a,W_i)-\mu}{\sigma}$, where $TF(a, W_i)$ is the frequency of the n-gram a in W_i; μ is the mean frequency of the background set; and σ is the standard deviation of the background set. In this context the background set is the set of web pages that share the person name. This score gives an idea of the distance of the frequency of an n-gram in a web page from the general distribution of this n-gram in the background set.

3.3 Similarity Functions

To determine the similarity between two web pages we try the cosine distance, a widely measure used in clustering, and the weighted Jaccard coefficient between two bags of n-grams defined as $W.Jaccard(W_i^n, W_j^n) = \frac{\sum_k min(m(t_{k_i}^n,i),m(t_{k_j}^n,j))}{\sum_k max(m(t_{k_i}^n,i),m(t_{k_j}^n,j))}$, where the meaning of $m(t_{k_i}^n, i)$ is explained in Section 3.4. Since weighted Jaccard coefficient needs non-negative entries and we want the cosine similarity of two documents to range from 0 to 1, we translate the values of the z-score so that they are always non-negative.

3.4 Algorithm

The algorithm $UPND$ (Unsupervised Person Name Disambiguator) can be seen in Algorithm 1. The description of this first algorithm does not take into account social profile web pages.

[3] http://nlp.stanford.edu/software/CRF-NER.shtml

$UPND$ algorithm receives as input a set of web documents with a mention to the same person name, let be $\mathcal{W} = \{W_1, W_2, \ldots, W_N\}$, and starts assigning a cluster C_i for each document W_i. $UPND$ also receives as input a pair of positive integer values r_1 and r_2, such that $r_2 \geq r_1$, specifying the range of values of n in the n-grams extracted from each web document. In each step of the algorithm we assign to each web page W_i a bag of n-grams $W_i^n = \{(t_1^n, m(t_1^n, i)), (t_2^n, m(t_2^n, i)), \ldots, (t_{k_i}^n, m(t_{k_i}^n, i))\}$, where each t_r^n is a n-gram extracted from W_i and $m(t_r^n, i)$ is the corresponding weight of the n-gram t_r^n in the web page W_i, being $r \in \{1, 2, \ldots, k_i\}$. In Algorithm 1 the function $setNGrams(n, \mathcal{W})$ in line 6 calculates for each web page in the set $\mathcal{W}$ its bag of n-grams representation. $Sim(W_i^n, W_j^n)$ in line 9 refers to the similarity between web pages W_i and W_j.

To decide when two web pages refer the same individual we propose a threshold γ. For each pair of web pages represented as bag of n-grams, let be W_i^n and W_j^n, we compute the threshold as follows: $\gamma(W_i^n, W_j^n) = \frac{min(m,k) - shared(W_i^n, W_j^n)}{max(m,k)}$, where m and k are the number of n-grams of W_i and W_j respectively, and $shared(W_i^n, W_j^n)$ is the number of n-grams shared by those web pages i.e. $shared(W_i^n, W_j^n) = |W_i^n \cap W_j^n|$. Notice that $shared(W_i^n, W_j^n)$ is superiorly limited by $min(m, k)$.

This threshold holds two desirable properties: (i) The more n-grams are shared by W_i and W_j, the lower $\gamma(W_i^n, W_j^n)$ is, so the clustering condition of the algorithm is less strict. (ii) It avoids the penalization due to big differences between the size of the web pages.

Thus, we decide that two web pages W_i and W_j refer to the same person if $Sim(W_i^n, W_j^n) \geq \gamma(W_i^n, W_j^n)$, so $C_i = C_i \cup C_j$ (lines 9, 10 and 11).

We assume that we can get accurate and reliable information for disambiguating with n-grams of at least size 3. Thus, we propose to iterate this process for 3-grams and 4-grams, i.e. $UPND(\mathcal{W}, 3, 4)$. We consider that selecting a value of n grater than 4 could lead to find few n-grams, so that many web pages could be under-represented. On the other hand, previous experiments using also bigrams showed that they are not suitable for this approach. This algorithm is polynomial and has a computational cost in $\mathcal{O}(N^2)$, where N is the number of web pages.

Algorithm 1 $UPND(\mathcal{W}, r_1, r_2)$

Require: Set of web pages that shared a person name $\mathcal{W} = \{W_1, W_2, ..., W_N\}$, $r_1, r_2 \geq 1$ such that $r_2 \geq r_1$

Ensure: Set of clusters $\mathcal{C} = \{C_1, C_2, ..., C_l\}$

1: **for** $n = 1$ **to** N **do**
2: $C_i = \{W_i\}$
3: **end for**
4: $\mathcal{C} = \{C_1, C_2, ..., C_N\}$.
5: **for** $n = r_1$ **to** r_2 **do**
6: $setNGrams(n, \mathcal{W})$.
7: **for** $i = 1$ **to** N **do**
8: **for** $j = i + 1$ **to** N **do**
9: **if** $Sim(W_i^n, W_j^n) \geq \gamma(W_i^n, W_j^n)$ **then**
10: $C_i = C_i \cup C_j$
11: $\mathcal{C} = \mathcal{C} \setminus \{C_j\}$
12: **end if**
13: **end for**
14: **end for**
15: **end for**
16: **return** $\mathcal{C}$

3.5 Social Media Treatment

Social networking services have increased their popularity and number of users in the last years. This fact affects this task mainly in two ways. On one hand, as a result of the success of this kind of platforms,

a lot of web pages contain terms related to them (e.g. the name of these platforms: Twitter, Facebook, LinkedIn, etc.). On the other hand, for a person name query in a search engine, it usually returns several profiles of such person name that are as much in the same as in different social networking services. These profiles usually are from different people sharing the same name, so they should be in different clusters. Most of the methods of the state of the art do not take into account this fact, usually taking as features tokens from the URL or the title of each web page, which includes the name of these platforms. This practice could lead to add noise to the representation of the web pages.

(Berendsen et al., 2012) proposed the "one in one" baseline to deal with social platform web pages, which creates a singleton cluster for each social web page. However, its main disadvantage is that it does not consider that a same individual could have accounts in several social platforms. A search engine could also return web pages from a social platform which are not profiles, as for example, a group page of Facebook where a person expounds an opinion, in addition to the profile of the same individual in that social platform. In these cases the "one in one" baseline also fails.

We propose a heuristic method that takes into account the limitations of the "one in one" heuristic, letting group social web pages from different platforms and also cluster social web pages from the same social platform. The algorithm that implements our heuristic is $SUPND$ (Social UPND). This algorithm applies $UPND$ with the following restriction: two web pages assigned to the same social networking service cannot be compared. This policy is taken because when a search engine returns several links from the same social platform, they usually refer to different individuals. However, this does not necessarily imply that two web pages belonging to the same social site cannot belong to the same cluster, because they would be compared to other webs pages separately, possibly ending up in the same cluster in a transitive way. For example, giving two web pages from Facebook, let be FB_1 and FB_2, and a non-social web page W, then FB_1 and FB_2 would not be compared, however each FB_i would be compared with W. If $SUPND$ decides to cluster each FB_i with W, then finally both web pages, from the same platform, would be in the same cluster. To identify the social web pages we obtain a list of social media platforms from Wikipedia[4], so when looking at the URL of a web page, we can detect if it corresponds to any of those social media platforms. If it is the case, we assign to that web page its social media site. The computational cost of $SUPND$ is the same of $UPND$.

4 Experiments

In this section we present the corpora of web pages used, the preprocessing of each web page, the experiments carried out and the obtained results.

4.1 Web People Search Collections

WePS is a competitive evaluation campaign that proposes several tasks including resolution of disambiguation on the Web data. In particular, WePS-1, WePS-2 and WePS-3 campaigns provide an evaluation framework consisting in several annotated data sets composed of English person names.

In these experiments we use WePS-1 (Artiles et al., 2007) test corpus composed by 30 English person names and the top 100 search results from Yahoo! search engine; WePS-2 (Artiles et al., 2009a) containing 30 person names and the top 150 search results from Yahoo! search engine; and WePS-3 (Artiles et al., 2010) containing 300 person names and the top 200 search results from Yahoo! All WePS corpora have few social profile web pages, so the impact of this kind of pages in the results of the algorithms is insignificant. We also use the ECIR2012 corpus, which is composed by 33 Dutch person names selected from query logs of a people search engine. For each person name the web pages set is built retrieving several profiles from social media platforms as Facebook, Twitter or LinkedIn, and results returned by Google, Bing and Yahoo! search engines. This data set gives a more real scenario for this task than the WePS ones, because it includes social network profiles of several person sharing the same name.

[4] en.wikipedia.org/wiki/Category:Social_networking_services

4.2 Corpus Preprocessing

Given a person name and a set of web pages, we first discard web pages that do not mention such name using several patterns that take into account the usual structure of person names.

For each not discarded web page, we delete the name and the surname because they appear in all the remaining documents and are the object of the ambiguity. We also delete stop words.

4.3 Results and Discussion

We present our results for all the corpora comparing them with the state of the art systems. The figures in the tables are macro-averaged, i.e., they are calculated for each person name and then averaged over all test cases. For WePS data sets we get the same results for $UPND$ and $SUPND$ algorithms, because these collections include few social networking profiles. The metrics used in this section are the BCubed metrics defined in (Bagga and Baldwin, 1998): BCubed precision (BP), BCubed recall (BR) and their harmonic mean $F_{0.5}(BP/BR)$. (Artiles, 2009) showed that these metrics are accurate for clustering tasks, particularly for person name disambiguation in the Web. We use the Wilcoxon test (Wilcoxon, 1945) to detect statistical significance in the differences of the results considering a confidence level of 95%. In order to compare our algorithm with the WePS better results using the Wilcoxon test, the samples consist in the pairs of values $F_{\alpha=0.5}(BP/BR)$ of each system for each person name.

First, Table 1 shows the results of $UPND$ using n-grams of capitalized tokens and n-grams of NE with WePS-1 training corpus. Experiments include the three weighting functions and the two similarity functions. The results of using n-grams of NE rank below those obtained with n-grams of capitalized tokens in all cases. The Wilcoxon test comparing the results of both representations shows that there are significant differences between them, except TF and TF-IDF with cosine. So we can conclude that in our approach using n-grams of capitalized tokens outperforms the use of n-grams of NE, what confirms our hypothesis.

	TF		z-score		TF-IDF	
Representation	W. Jaccard	Cosine	W. Jaccard	Cosine	W. Jaccard	Cosine
Capitalized n-gram	**0.82**	**0.69**	**0.83**	**0.78**	**0.81**	**0.63**
NE (Stanford NER)	0.77	0.6	0.77	0.72	0.76	0.6

Table 1: $F_{0.5}(BP/BR)$ results of $UPND$ algorithm comparing capitalized n-gram and NE n-gram representations with WePS-1 training corpus.

In Table 2 we show the results of $UPND$ for all WePS test data sets with the three weighting functions and the two similarity measures.

		WePS-1			WePS-2			WePS-3		
		BP	BR	$F_{0.5}$(BP/BR)	BP	BR	$F_{0.5}$(BP/BR)	BP	BR	$F_{0.5}(BP/BR)$
W. Jaccard	TF	0.73	0.77	0.74	0.82	0.82	**0.81**	0.46	0.70	0.50
	z-score	0.70	0.78	0.72	0.80	0.84	**0.81**	0.44	0.72	0.50
	TF-IDF	0.73	0.77	0.73	0.82	0.82	**0.81**	0.46	0.70	0.50
Cosine	TF	0.92	0.61	0.72	0.95	0.61	0.73	0.75	0.45	0.51
	z-score	0.85	0.69	**0.76**	0.91	0.73	**0.81**	0.62	0.56	**0.53**
	TF-IDF	0.94	0.57	0.7	0.96	0.52	0.65	0.79	0.40	0.49

Table 2: Results of $UPND$ algorithm for WePS test data sets.

The combination of z-score with cosine gets the best balance between the values of BP and BR, reaching the highest results of $F_{\alpha=0.5}$ for the three WePS corpora. The combination of TF-IDF with cosine gets the best BP results, but BR results are the lowest. On the other hand, the combination of z-score and Jaccard gets the best BR results, but the BP results are the lowest.

Regarding the significance of the differences between the best results, the improvement between z-score with cosine and z-score with Jaccard is significant in WePS-1 and WePS-3, but not in WePS-2. The improvement between z-score with cosine and Jaccard with TF is significant only in WePS-3.

Thus, we select the combination of z-score as weight function and cosine as similarity function as the most suitable combination for our algorithm. Therefore we use it in the following experiments.

Table 3 shows the results of $UPND$ with WePS-1 test, WePS-2 and WePS-3 corpora in addition to the top ranking systems of the campaigns, and also the results obtained by HAC_Topic system in the case of WePS-1. We include the results obtained by three unsupervised baselines called ALL_IN_ONE, ONE_IN_ONE and Fast AP. ALL_IN_ONE provides a clustering solution where all the documents are assigned to a single cluster, ONE_IN_ONE returns a clustering solution where every document is assigned to a different cluster, and Fast AP applies a fast version of Affinity Propagation described in (Fujiwara et al., 2011) using the function TF-IDF to weight the tokens of each web page, and the cosine distance to compute the similarity.

	System	BP	BR	$F_{0.5}(BP/BR)$
WePS-1	(+) HAC_Topic	0.79	0.85	**0.81** †
	(-) ***UPND***	0.85	0.69	0.76 •
	(+)(*) CU_COMSEM	0.61	0.83	0.70 †
	(+)(*) PSNUS	0.68	0.73	0.70 †
	(+)(*) IRST-BP	0.68	0.71	0.69 †
	(+)(*) UVA	0.79	0.50	0.61 †
	(+)(*) SHEF	0.54	0.74	0.62 †
	(-) ONE_IN_ONE	**1.00**	0.43	0.57 •
	(-) Fast AP	0.69	0.55	0.56 †
	(-) ALL_IN_ONE	0.18	**0.98**	0.25 •
WePS-2	(+) ORACLE_1	0.89	0.83	**0.85** •
	(+) ORACLE_2	0.91	0.81	**0.85** •
	(+)(*) PolyUHK	0.87	0.79	0.82
	(+)(*) ITC-UT_1	0.93	0.73	0.81
	(-) ***UPND***	0.91	0.73	0.81 •
	(+)(*) UVA_1	0.85	0.80	0.81
	(+)(*) XMEDIA_3	0.82	0.66	0.72 †
	(+)(*) UCI_2	0.66	0.84	0.71 †
	(-) ALL_IN_ONE	0.43	**1.00**	0.53 •
	(-) Fast AP	0.80	0.33	0.41 †
	(-) ONE_IN_ONE	**1.00**	0.24	0.34 •
WePS-3	(+)(*) YHBJ_2	0.61	0.60	**0.55**
	(-) ***UPND***	0.62	0.56	0.53 •
	(+)(*) AXIS_2	0.69	0.46	0.50 †
	(+)(*) TALP_5	0.40	0.66	0.44 †
	(+)(*) RGAI_AE_1	0.38	0.61	0.40 †
	(+)(*) WOLVES_1	0.31	0.80	0.40 †
	(-)(*) DAEDALUS_3	0.29	0.84	0.39 †
	(-) Fast AP	0.73	0.30	0.38 †
	(-) ONE_IN_ONE	**1.00**	0.23	0.35 •
	(-) ALL_IN_ONE	0.22	**1.00**	0.32 •

Table 3: Results of $UPND$ and the top state of the art systems with WePS corpora: (+) means system with supervision; (-) without supervision and (*) campaign participant. Significant differences between $UPND$ and other systems are denoted by (†); (•) means that in this case the statistical significance is not evaluated.

Our method $UPND$ outperforms WePS-1 participants and all the unsupervised baselines described before. HAC_Topic also outperforms the WePS-1 top participant systems and our algorithm. This system uses several parameters obtained by training with the WePS-2 data set: token weight according to the kind of token (terms from URL, title, snippets, ...) and thresholds used in the clustering process. Note that WePS-1 participants used the training corpus provided to the campaign, the WePS-1 training data, so in this case the best performance of HAC_Topic could be not only because of the different approach, but also because of the different training data set.

Our algorithm obtains significative better results than the WePS-1 top participant results, and HAC_Topic obtains significative better results than it according to the Wilcoxon test. $UPND$ obtains significative better results than IRST-BP system (the third in the WePS-1 ranking), also based on the co-ocurrence of n-grams.

Regarding WePS-2 we add in Table 3 two oracle systems provided by the organizers. These systems use BoW of tokens (ORACLE_1) or bigrams (ORACLE_2) weighted by TF-IDF, deleting previously stop words, and later apply HAC with single linkage with the best thresholds for each person name. We do not include the results of the HAC_Topic system since it uses this data set for training their algorithm.

The significance test shows that the top WePS-2 systems PolyUHK, UVA_1 and ITC-UT_1 obtain

similar results than $UPND$, however they use some kind of supervision. The results of all these systems are the closest to the oracle systems provided by the organizers, which know the best thresholds for each person name.

In the case of WePS-3, the organizers did not take into account the whole clustering solution provided by the systems like in previous editions, but only checks the accuracy of the clusters corresponding to two selected individuals per person name. In this case, the first two systems YHBJ_2 and $UPND$ do not have significant difference in their results. Notice that YHBJ_2 system makes use of concepts extracted manually from Wikipedia. Note that $UPND$ also obtains significative better results than DAEDALUS_3, the only one participant that does not use training data.

Regarding the experiments with the ECIR2012 corpus, which contains social profiles, Table 4 shows the results of the two versions of our algorithm and the results of the system of the University of Amsterdam (UvA). As far as we know, no other systems have been tested with this gold standard. $SUPND$ obtains significative better results than $UPND$ due to its special treatment for social web pages. The UvA system outperforms our algorithm $SUPND$ and this improvement is significative. Note that the heuristic for social pages in $SUPND$ outperforms $UPND$ using the "one in one" heuristic.

System	BP	BR	$F_{0.5}(BP/BR)$
(+) UvA (best perf.)	0.90	0.80	**0.83** †
(-) $SUPND$	0.95	0.68	0.78 •
(-) $UPND$ (one in one)	0.98	0.62	0.74 †
(-) $UPND$	0.74	0.74	0.72 †

Table 4: Results of $SUPND$ and $UPND$ algorithms for ECIR2012 corpus: (+) means system with supervision and (-) without supervision. Significant differences between $SUPND$ and other systems are denoted by (†); (•) means that in this case the statistical significance is not evaluated.

After all these experiments, we can conclude that our approach gets the best results of all the completely unsupervised approaches. Moreover, the precision scores for all collections are very high and confirm that our approach is accurate to get relevant information for characterizing an individual. We also obtain competitive recall results, what lead to a competitive system that carries out person name disambiguation in web search results with minimum human supervision.

5 Conclusions and Future Work

We present a new approach for person name disambiguation of web search results. Our method does not need training data to calculate thresholds to determine the number of different individuals sharing the same name, or whether two web pages refer to the same individual or not. Although supervised approaches have been successful in many NLP and IR tasks, they require enough and representative training data to guaranty the results will be consistent for different data collections, which requires a huge human effort.

The two algorithms proposed provide a clustering solution for this task by means of data-driven methods that do not need learning from data. Our approach is not very expensive in computational cost, obtaining very competitive results in several data sets compared with the best state of the art systems.

Our proposal is based on getting reliable information for disambiguating, particularly long n-grams composed by uppercase tokens. According to our results, this hypothesis has shown successful, getting high precision values and acceptable recall scores. Anyway, we would like to improve recall results without losing of precision, filter out noisy capitalized n-grams, and build an alternative representation for web pages containing all their tokens in lowercase.

We have observed that this task gets harder when we have to deal with social media profiles. A system thought for being used in a real scenario has to take into account this kind of web pages, since they are usually returned by search engines when a user introduces a person name as a query. Most state of the art systems do not deal with this problem. We have proposed in this paper a new heuristic method for processing social platforms profiles for this clustering task.

Person name disambiguation has been mainly addressed in a monolingual scenario, e.g. WePS corpora are English data sets and Dutch the ECIR2012 collection. We would like to address this task in a multilingual scenario. Although search engines return their results taking into account the country of the user, with some queries we can get results written in several languages. This scenario has not been considered by the state of the art systems so far.

Acknowledgements

This work has been part-funded by the Spanish Ministry of Science and Innovation (MED-RECORD Project, TIN2013-46616-C2-2-R) and by UNED Project (2012V/PUNED/0004).

References

Miguel A. Andrade, and Alfonso Valencia. 1998. *Automatic extraction of keywords from scientific text: application to the knowledge domain of protein families*. Bioinformatics, 14:600-607, 1998.

Javier Artiles, Julio Gonzalo and Satoshi Sekine. 2007. The SemEval-2007 WePS Evaluation: Establishing a Benchmark for the Web People Search Task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 64–69, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Javier Artiles. 2009. *Web People Search*. PhD Thesis, UNED University.

Javier Artiles, Julio Gonzalo and Satoshi Sekine. 2009b. Weps 2 Evaluation Campaign: Overview of the Web People Search Clustering Task. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

Javier Artiles, Andrew Borthwick, Julio Gonzalo, Satoshi Sekine and Enrique Amigó . 2010. WePS-3 Evaluation Campaign: Overview of the Web People Search Clustering and Attribute Extraction Tasks. In *Third Web People Search Evaluation Forum (WePS-3), CLEF 2010*.

Amit Bagga and Breck Baldwin. 1998. Entity-Based Cross-Document Coreferencing Using the Vector Space Model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 79–85, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics.

Krisztian Balog, Jiyin He, Katja Hofmann, Valentin Jijkoun, Christof Monz, Manos Tsagkias, Wouter Weerkamp, and Maarten de Rijke. 2009. The University of Amsterdam at WePS-2. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

Richard Berendsen, Bogomil Kovachev, Evangelia-Paraskevi Nastou, Maarten de Rijke, and Wouter Weerkamp. 2012. Result Disambiguation in Web People Search. In *Proceedings of the 34th European conference on Advances in Information Retrieval*, ECIR'12, pages 146–157, Berlin, Heidelberg, 2012. Springer-Verlag.

Ying Chen and James Martin. 2007. CU-COMSEM: Exploring Rich Features for Unsupervised Web Personal Named Disambiguation. In *Proceedings of SemEval 2007, Assocciation for Computational Linguistics*, pages 125–128, 2007.

Ying Chen, Sophia Yat Mei Lee and Chu-Ren Huang. 2009. PolyUHK: A Robust Information Extraction System for Web Personal Names. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

Ergin Elmacioglu, Yee Fan Tan, Su Yan, Min-Yen Kan, and Dongwon Lee. 2007. PSNUS: Web People Name Disambiguation by Simple Clustering with Rich Features. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 268–271, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

Yasuhiro Fujiwara, Go Irie and Tomoe Kitahara. 2011. Fast Algorithm for Affinity Propagation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence(IJCAI)- Volume Three*, 2238–2243, Barcelona, Catalonia, Spain.

Sara Lana-Serrano , Julio Villena-Román , José Carlos González-Cristóbal. 2010. Daedalus at WebPS-3 2010: k-Medoids Clustering using a Cost Function Minimization. In *Third Web People Search Evaluation Forum (WePS-3), CLEF 2010*.

Zhengzhong Liu, Qin Lu, and Jian Xu. 2011. High Performance Clustering for Web Person Name Disambiguation using Topic Capturing. In *International Workshop on Entity-Oriented Search (EOS)*, 2011.

Chong Long and Lei Shi. 2010. Web Person Name Disambiguation by Relevance Weighting of Extended Feature Sets. In *Third Web People Search Evaluation Forum (WePS-3), CLEF 2010*.

Gideon S. Mann. 2006. *Multi-Document Statistical Fact Extraction and Fusion*. PhD thesis, Johns Hopkins University, Baltimore, MD, USA, 2006. AAI3213760.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

Octavian Popescu and Bernardo Magnini. 2007. IRST-BP: Web People Search Using Name Entities In *In Proceedings of SemEval 2007, Assocciation for Computational Linguistics*, pages 195–198, 2007.

Frank Wilcoxon. 1945. *Individual Comparisons by Ranking Methods*, volume 1 (6). Biometrics Bulletin, December 1945.

Picking the Amateur's Mind – Predicting Chess Player Strength from Game Annotations

Christian Scheible
Institute for Natural Language Processing
University of Stuttgart, Germany
scheibcn@ims.uni-stuttgart.de

Hinrich Schütze
Center for Information
and Language Processing
University of Munich, Germany

Abstract

Results from psychology show a connection between a speaker's expertise in a task and the language he uses to talk about it. In this paper, we present an empirical study on using linguistic evidence to predict the expertise of a speaker in a task: playing chess. Instructional chess literature claims that the mindsets of amateur and expert players differ fundamentally (Silman, 1999); psychological science has empirically arrived at similar results (e.g., Pfau and Murphy (1988)). We conduct experiments on automatically predicting chess player skill based on their natural language game commentary. We make use of annotated chess games, in which players provide their own interpretation of game in prose. Based on a dataset collected from an online chess forum, we predict player strength through SVM classification and ranking. We show that using textual and chess-specific features achieves both high classification accuracy and significant correlation. Finally, we compare our findings to claims from the chess literature and results from psychology.

1 Introduction

It has been recognized that the language used when describing a certain topic or activity may differ strongly depending on the speaker's level of expertise. As shown in empirical experiments in psychology (e.g., Solomon (1990), Pfau and Murphy (1988)), a speaker's linguistic choices are influenced by the way he thinks about the topic. While writer expertise has been addressed previously, we know of no work that uses linguistic indicators to rank experts.

We present a study on predicting chess expertise from written commentary. Chess is a particularly interesting task for predicting expertise: First, using data from competitive online chess, we can compare and rank players within a well-defined ranking system. Second, we can collect textual data for experimental evaluation from web resources, eliminating the need for manual annotation. Third, there is a large amount of terminology associated with chess, which we can exploit for n-gram based classification.

Chess is difficult for humans because it requires long-term foresight (*strategy*) as well as the capacity for internally simulating complicated move sequences (*calculation* and *tactics*). For these reasons, the game for a long time remained challenging even for computers. Players have thus developed general principles of chess strategy on which many expert players agree. The dominant expert view is that the understanding of fundamental strategical notions, supplemented by the ability of calculation, is the most important skill of a chess player. A good player develops a long-term *plan* for the course of the game. This view is the foundation of many introductory works to chess (e.g., Capablanca (1921), one of the earliest works).

Silman (1999) presents games he played with chess students, analyzing their commentary about the progress of the game. He claims that players who fail to adhere to the aforementioned basic principles tend to perform worse and argues that the students' thought processes reflect their playing strength directly. Lack of strategical understanding marks the difference between amateur and expert players. Experts are mostly concerned with *positional* aspects, i.e., the optimal placement of pieces that offers a

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 311–321, Dublin, Ireland, August 23-29 2014.

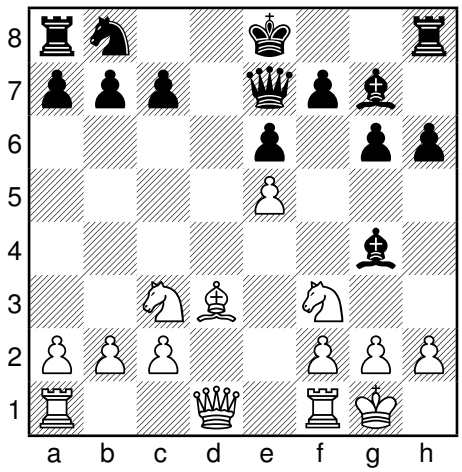


Figure 1: Example chess position, white to play

long-lasting advantage. Amateurs often have *tactical* aspects in mind, i.e., short-term attacking opportunities and exploits that potentially lead to loss of material for their opponents. A correlation between chess strength and verbalization skills has been shown empirically by Pfau and Murphy (1988), who used experts to assess the quality of the subjects' writing.

In this paper, we investigate the differences between the mindset of amateurs and experts expressed in written game commentary, also referred to as *annotated games*. When studying chess, it is best practice to review one's own games to further one's understanding of the game (Heisman, 1995). Students are encouraged to *annotate* the games, i.e., writing down their thought process at each move. We address the problem of predicting the player's strength from the text of these annotations. Specifically, we want to predict the rank of the player at the point when a given game was played. In competitive play, the rank is determined through a numerical rating system – such as the Elo rating system (Elo, 1978) used in this paper – that measures the players' relative strength using pairwise win expectations.

This paper makes the following contributions. First, we introduce a novel training dataset of games annotated by the players themselves – collected from online chess forum. We then formulate the task of playing strength prediction. For each annotated game, each game viewed as a document, we predict the rating class or overall rank of the player. We show that (i) an SVM model with n-gram features succeeds at partitioning the players into two rating classes (above and below the mean rating); and (ii) that ranking SVMs achieve significant correlation between the true and predicted ranking of the players. In addition, we introduce novel chess-specific features that significantly improve the results. Finally, we compare the predictions made by our model to claims from instructional chess literature and results from psychology research.

We next give an overview of basic chess concepts (Section 2). Then, we introduce the dataset (Section 3) and task (Section 4). We present our experimental results in Section 5. Section 6 contains an overview of related work.

2 Basic Chess Concepts

2.1 Chess Terminology

We assume that the reader has basic familiarity with chess, its rules, and the value of individual pieces. For clarity, we review some basic concepts of chess terminology, particularly elementary concepts related to tactics and strategy in an example position (Figure 1).[1]

From a *positional* point of view, white is ahead in *development*: all his *minor pieces* (bishops and knights) have moved from their starting point while black's knight remains on b8. White has also *castled* (a move where the rook and king move simultaneously to get the king to a safer spot on either side of the board) while black has not. White has a *space* advantage as he occupies the e5-square (which is in black's

[1]Modified from the game Dzindzichashvili – Yermolinsky (1993) which is the first position discussed in (Silman, 1999)

1.e4 e5 2.Nf3 Nc6 3.Bc4 Nh6 4.Nc3 Bd6 Trying to follow basic opening principals, control center, develop. etc **5.d3 Na5 6.Bb5** Moved bishop not wanting to trade, but realized after the move that my bishop would be harassed by the pawn on c7 **6...c6 7.Ba4** Moved bishop to safety, losing tempo **7...Qf6 8.Bg5 Qg6 9.O-O b5** Realized my bishop was done, might as well get 2 pawns **10.Nxb5 cxb5 11.Bxb5 Ng4 12.Nxe5** Flat out blunder, gave up a knight, at least I had a knight I could capture back **12...Bxe5 13.Qxg4 Bxb2 14.Rab1 Bd4 15.Rfe1** Moved rook to E file hoping to eventually attack the king. **15...h6 16.c3** Poor attempt to move the bishop, I realized it after I made the move **16...Bxc3 17.Rec1 Be5 18.d4** Another crappy attempt to move that bishop **18...Bxd4 19.Rd1 O-O 20.Rxd4 d6 21.Qd1** I don't remember why I made this move. **21...Qxg5 22.Rxd6 Bh3 23.Bf1** Protecting g2 **23...Nc4 24.Rd5 Qg6 25.Rc1 Qxe4 26.f3 Qe3+ 27.Kh1 Nb2 28.Qc2 Rac8 29.Qe2 Qxc1 30.gxh3 Nc4 31.Qe4 Qxf1#**

Figure 2: Example of an annotated game from the dataset (by user aevans410, rated 974)

half of the board) with a pawn. This pawn is potentially *weak* as it cannot easily be defended by another pawn. Black has both of his bishops (the *bishop pair*) which is considered advantageous as bishops are often superior to knights in open positions. Black's light-square bishop is *bad* as it is obstructed by black's own pawns (although it is outside the *pawn chain* and thus flexible). *Strategically*, black might want to improve the position of the light-square bishop, make use of his superior dark-square bishop, and try to exploit the weak e5 pawn. Conversely, white should try create posts for his knights in black's territory. *Tactically*, white has an opportunity to move his knight to b5 (written Nb5 in algebraic chess notation), from where it would attack the pawn on c7. If the knight could reach c7 (currently defended by black's queen), it would *fork* (*double attack*) black's king and rook, which could lead to the trade of the knight for the rook on the next move (which is referred to as winning the *exchange*). White's knight on f3 is *pinned*, i.e., the queen would be lost if the knight moved. Black can win a pawn by *removing the defender* of e5, the knight on f3, by capturing it with the bishop.

This brief analysis of the position shows the complex theory and terminology that has developed around chess. The paragraph also shows an example of game annotation (although not every move in the game will be covered as elaborately in practice in amateur analyses).

2.2 Elo Rating System

Our goal in this paper is to predict the ranking of chess players based on their game annotations. We will give a brief overview of the Elo system (Elo, 1978) that is commonly used to rank players. Each player is assigned a score that is changed after each game depending on the expected and actual outcome. On `chess.com`, a new player starts with an initial rating of 1200 (an arbitrary number chosen for historical reasons, which has since become a wide-spread convention in chess). Assuming the current ratings R_a and R_b of two players a and b, the expected outcome of the game is defined as

$$E_a = \frac{1}{1 + 10^{-\frac{R_a - R_b}{400}}}.$$

E_a is then used to conduct a (weighted) update of R_a and R_b given the actual outcome of the game. Thus, Elo ratings make pairwise adjustments to the scores. The differences between the ratings of two players predict the probability of one winning against the other. However, the absolute ratings do not carry any meaning by themselves.

3 Annotated Chess Game Data

For supervised training, we require a collection of chess games annotated by players of various strengths. An annotated chess game is a sequence of chess moves with natural language text commentary associated to specific moves. While many chess game collections are available, some of them containing millions of games, the majority are unannotated. The small fraction of annotated games mostly features commentary by masters rather than amateurs, which is not interesting for a contrastive study.

The game analysis forum on `chess.com` encourages players to post their annotated games for review through the community. While several games are posted each day, we can only use a small subset of them.

Parameter	Value
# games	182
# different players	130
mean # moves by game	42
mean # annotated moves by game	16
mean # words by game	114

Table 1: Dataset statistics

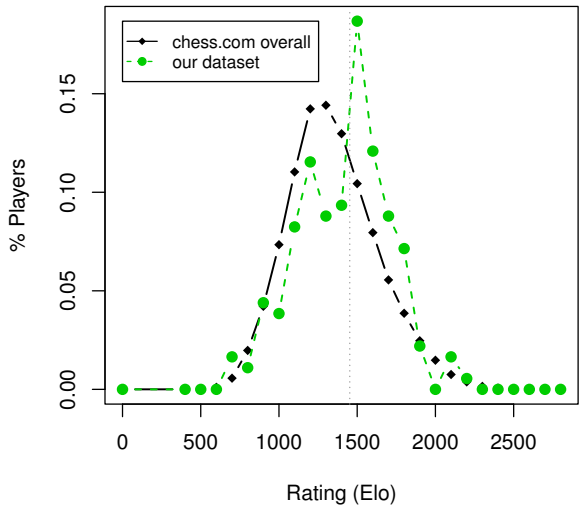


Figure 3: Rating distribution on `chess.com` and our dataset.[4] Each point shows the percentage of players in a bin of width 50 around the value. Dotted line: Median on our dataset used for binning.

Many games are posted without annotations, instead soliciting annotation from the community. Others are missing the rating of the player at the time the game was played – the user profile shows only the current rating for the player which may differ strongly from their historical one.

We first downloaded all available games from the forum archive. The games are stored in portable game notation (PGN, Edwards (1994)). Next, we manually removed games where the annotation had been conducted automatically by a chess program. We also removed games that had annotations at fewer than three moves. The final dataset consists of 182 games with annotations in English and known player rating.[2] We reproduce an example game from the data in Figure 2. This game is typical as the first couple of moves are not commented (as opening moves are typically well-known). Then, the annotator comments on select moves that he believes are key to the progress of the game. Table 1 shows some statistics about the dataset.

The distribution of the ratings in our dataset is shown in Figure 3 in comparison to the overall standard chess rating distribution on `chess.com`.[3] Elo ratings assume a normal distribution of players. We see that overall, the distributions are quite similar, although we have a higher peak and our sample mean is shifted towards higher ratings (1347 overall vs 1462 on our dataset). It is more common for mid-level players to request annotation advice than it is for low-rated players (who might not know about this practice) or high-rated players (who do not look for support by the lower-rated community).

The dataset is still somewhat noisy as players may obtain different ratings depending on the type of venue (over-the-board tournament vs online chess) or the amount of time the players had available (*time control*). Differences in these parameters lead to different rating distributions.[4] For this reason, the total ordering given through the ratings may be difficult to predict. Thus, we will conduct experiments both

[2]Available at `http://www.ims.uni-stuttgart.de/data/chess`
[3]Data from `http://www.chess.com/echess/players`
[4]cf. `http://www.chess.com/article/view/chesscom-rating-comparisons`

on ranking and on classification where the rating range is binned into two rating classes.

4 Predicting Chess Strength from Annotations

4.1 Classification and Ranking

The task addressed in this paper is prediction on the game level, i.e., predicting the strength of the player of each game at the time when the game was played. We view a game as a document – the concatenation of the annotations at each move – and extract feature vectors as described in Section 4.2. We pursue two different machine learning approaches based on support vector machines (SVMs) to predicting chess strength: classification and ranking.

The simplest way to approach the problem is classification. For this purpose, we divide the range of observed rating into two evenly spaced rating classes at the median of the overall rating range (henceforth *amateur* and *expert*). The classification view has obvious disadvantages. At the boundaries of the bins, the distinction between them becomes difficult.

To predict a total ordering of all players, we use a ranking SVM (Herbrich et al., 1999). This model casts ranking as learning a binary classification function that decides whether $\mathit{rank}(\mathbf{x_1}) > \mathit{rank}(\mathbf{x_2})$ over all possible pairs of example feature vectors $\mathbf{x_1}$ and $\mathbf{x_2}$ with differing *rank*.

Note that since Elo ratings are continuous real numbers, it would be conceivable to fit a regression model. However, Elo is designed as a pairwise ranking measure. While a relative difference in Elo represents the probability of one player beating the other, the absolute Elo rating is not directly interpretable.[5]

4.2 Features

We extract unigrams (UG) and bigrams (BG) from the texts. In addition, we propose the following two chess-specific feature sets derived from the text:[6]

Notation (NOT). We introduce two indicators for whether the annotations contain certain types of formal chess notation. The feature SQUARE is added if the annotation contains a reference to a specific square on the chess board (e.g., *d4*). If the annotation contains a move in algebraic notation (e.g., *Nxb4+*, meaning that a knight moved to b4, captured a piece there and put the enemy king in check), the feature MOVE is added.

Similarity to master annotations (MS). This feature is intended to compensate for the lack of training data. We used a master-annotated database consisting of 500 games annotated by chess masters which is available online.[7] As we do not know the exact rating of the annotators, and to avoid strong class imbalances, we cannot make use of the games directly through supervision. Instead, we calculate the cosine similarity between the centroid[8] of the n-gram feature vectors of the master games and each game in the `chess.com` dataset. The cosine similarity between each game and the master centroid is added as a numerical feature.

Additionally, the master similarity scores can be used on their own to rank the games. This can be viewed distant supervision as strength is learned from an external database. We will evaluate this ranking in comparison with our trained models.

5 Experiments

This section, contains experimental results on classifying and ranking chess players. We first present quantitative evaluation of the classification and ranking models and discuss the effect of chess-specific

[5]Preliminary experiments with SVM regression showed little improvements over a baseline of assigning the mean rating to all games. This suggests that the distribution of rankings is difficult to model – possibly due to the low number of annotated games on which the model can be trained.

[6]We also tried using the length of the annotation as well as the number of annotated moves as a feature, which did not contribute any improvements.

[7]`http://www.angelfire.com/games3/smartbridge/famous_games.zip`

[8]We also tried a k-NN approach where we computed the mean similarity of a game from our dataset to its k nearest neighbors among the master games ($k \in 1, 2, 5, \infty$), but found that this approach performed worse.

	Model	Features	$F_1^{(\downarrow)}$	$F_1^{(\uparrow)}$	$F_1^{(\varnothing)}$
1	*Majority BL*		67.2	0.0	33.6
2	SVM (linear)	UG	73.4	71.6	72.5
3	SVM (linear)	UG, BG	74.1	72.0	73.1
4	SVM (linear)	UG, BG, NOT	75.7	74.9	75.3
5	SVM (linear)	UG, BG, NOT, MS	74.2	73.0	73.6

(a) Results (F_1 in %)

	1	2	3	4	5
1					
2	**				
3	**				
4	**	○			
5	**				

(b) Statistical significance of differences in F_1. **: $p < 0.01$, *: $p < 0.05$, ○: $p < 0.1$

Table 2: Classification results

Class	Features
Amateur (↓)	bishop, d4, opening, instead, trying, should, did, where, do, even, rook, get, good, he, coming, point i, exchange, thought, did not, his, clock, too, or, on clock, knight for
Expert (↑)	this, game, can, will, winning, NOT:move, time, draw, because, white, back, black, mate, that, but, moves, can't, very, on, won, really, so, i know, now, only

Table 3: Top 25 features with most negative (amateur) and positive (expert) weights (mean over all folds) in the best setup (UG, BG, NOT)

features. Second, we qualitatively compare the predictions of our models with findings and claims from the literature about the connection between a player's mindset and strength.

5.1 Experimental Setup

To generate feature vectors, we first concatenate all the annotations for a game, tokenize and lowercase the texts, and remove punctuation as well as a small number of stopwords. We exclude rare words to avoid overfitting: We remove all n-grams that occur fewer than 5 times, and add the chess-specific features proposed above. Finally, we L_2-normalize each vector.

We use linear SVMs from LIBLINEAR and SVMs with RBF kernel from LIBSVM (Chang and Lin, 2011). We run all experiments in a 10-fold cross-validation setup.

We measure macro-averaged F_1 for our classification results. We evaluate the ranking model using two measures: pairwise ranking accuracy (Acc_r), i.e., the accuracy over the binary ranking decision for each player pair; and Spearman's rank correlation coefficient ρ for the overall ranking. To test whether differences between results are statistical significant, we apply approximate randomization (Noreen, 1989) for F_1, and the test by Steiger (1980) for correlations, which is applicable to ρ.

5.2 Classification

We first investigate the classification case, i.e., whether we can distinguish players below and above the rating mean. Table 2 shows the results for this experiment. We show F_1 scores for the lower and higher half of the players ($F_1^{(\downarrow)}$ and $F_1^{(\uparrow)}$, respectively), and the macro average of these two scores ($F_1^{(\varnothing)}$). We first note that all SVM classifiers (lines 2–5) score significantly higher than the majority baseline (line 1). When adding bigrams (line 3) and chess-specific notation features (line 4), F_1 increases. However, these improvements are not statistically significant. The master similarity feature (line 5) leads to a drop in F_1 from the previous line. The relatively low rank correlation between the master similarity scores and the two classes ($\rho = 0.334$) leads to this effect. The low correlation itself may occur because the master games were annotated by a third party (instead of the players), leading to strong differences in style.

There are several reasons for misclassification. Many errors occur in the dense region around the class boundary. Also, shorter game annotations are more difficult to classify than longer ones. For detailed error analysis, we first examine the most positively and negatively weighted features of the trained models (Table 3). We will provide a more detailed look into the features in Section 5.4. We

	Model	Features	Acc_r	ρ	sig
1	MS (standalone)	–	–	0.279	✓
2	SVM (linear)	UG	58.7	0.266	✓
3	SVM (linear)	UG, BG	58.8	0.286	✓
4	SVM (linear)	UG, BG, NOT	60.0	0.307	✓
5	SVM (linear)	UG, BG, NOT, MS	59.8	0.310	✓
6	SVM (RBF)	UG	64.0	0.389	✓
7	SVM (RBF)	UG, BG	63.9	0.395	✓
8	SVM (RBF)	UG, BG, NOT	63.8	0.400	✓
9	SVM (RBF)	UG, BG, NOT, MS	63.5	0.397	✓

(a) Ranking results (accuracy in % and ρ)

	1	2	3	4	5	6	7	8	9
1									
2									
3									
4			*						
5			*						
6		○	○						
7	○	*	○						
8	○	*	*	○			○		
9	○	○	○						

(b) Statistical significance of differences in ρ. **: $p < 0.01$, *: $p < 0.05$, ○: $p < 0.1$

Table 4: Ranking results for standalone master similarity and SVM (linear and RBF kernel). Check in sig column denote significance of correlation with true ranking ($p < 0.05$). Numbers in sigdiff column denote a significant improvement ($p < 0.05$) in ρ over the respective line.

find that there are noticeable differences in the writing styles of amateurs and experts. According to the model, one of the most prominent distinctions is that amateurs tend to refer to the opponent as *he*, whereas experts use *white* and *black* more frequently. However, it is of course not universally true, which leads to the misclassification of some experts as amateurs. Another difference in style is that amateur players tend to write about the game in the past tense. This is a manifestation of an important distinction: Amateurs often state the obvious developments of the game (e.g., *Flat out blunder, gave up a knight* in Figure 2) or speculate about options (e.g., *hoping to eventually attack*), while experts provide more thorough positional analysis at key points.

5.3 Ranking

We now turn to ranking experiments (Table 4). We first evaluate the ranking produced by ordering the games by their similarity to the master centroid (line 1). We find that the resulting rank correlation is low but significant.

The results for the linear SVM ranker are shown in lines 2–5. Total ranking is considerably more difficult than binary classification of rating classes. Using a linear SVM, we again achieve low but significant correlations. The linear classifiers (lines 2–5) do not significantly outperform the standalone master similarity (MS) baseline (line 1). Chess-specific features (lines 4 and 5) boost the results, outperforming the bigram models (line 3) significantly. The improvement from adding the MS centroid score feature is not significant.

We again perform error analysis by examining the feature weights (Table 5). We find an overall picture similar to the classification setup (cf. Table 3). The notation feature serves as a good indicator for the upper rating range (cf. Table 3) as experienced players find it easier to express themselves through notation. We observed that lower players tend to express moves in words (e.g., "move my knight to d5") rather than through notation (Nd5), which could serve as an explanation for why pieces (*bishop*, *knight*, *rook*) appear among the top features for amateur players.

However, some features change signs between the two experiments (e.g., *king*, *square*). This effect may indicate that the binary ranking problem is not linearly separable, which is plausible; mid-rated players may use terms that neither low-rated nor high-rated players use. Examining correlations at different ranking ranges confirms this suggestion. In top and bottom thirds of the rating scale, the true and predicted ranks are not correlated significantly. This means that the ranking SVM only succeeds at ranking players in middle third of the rating scale. To introduce non-linearity, we conduct further experiments with an SVM with a radial basis function (RBF) kernel.

The results of this experiment are shown in lines 6–9 of Table 4. All RBF models perform better than

Class	Features
Weaker	instead, king, thinking, one my, fight, d4, even, should, should i, bishop, decided, did, i didn't, opening, feel, put, defense, knight on, black king, been, with my, where, get, cover, pin
Stronger	NOT:move, moves, game, time, won, i know, already, will, stop, way, winning, line, can't, can, black has, this, MS, king side, computer, threaten, first, back, any way, my knight, win pawn, d

Table 5: Top 25 features with most negative (lower rating) and positive (higher rating) weights, mean over all folds ($rank(\mathbf{x_1}) > rank(\mathbf{x_2})$ or vice versa) in the best ranking setup (linear SVM, UG, BG, NOT)

Feature	Coefficient
capture	-0.29
take	-0.21
bishop	-1.06
knight	-0.19
rook	-0.54
king	0.19
queen	0.08
pawn	0.44
pin	-0.26
fork	-0.27

Feature	Coefficient
threat	0.13
danger	0.25
stop	0.50
weakness	0.34
light	0.21
dark	0.37
variation	0.41
winning	0.87
losing	0.08
like	-0.16
hate	-0.05
good	-0.27
bad	0.52

Feature	Coefficient
white	0.74
black	0.71
he	-0.51
fight	-0.17
know	0.41
will	0.88
thinking	-0.44
believe	-0.02
maybe	-0.19
hoping	-0.30

Feature	Coefficient
time	0.81
clock	-0.47
time pressure	-0.12
blunder	-0.31
tempo	-0.36
checkmate	-0.24
mate	0.69
opening	-0.63
castle	-0.33
fall	-0.22
eat	-0.28

Table 6: Selected SVM weights in the best 2-class setup, mean over all folds

the unigram and bigram linear models; all except for the unigram model (lines 7–9) also yield weakly significant improvements over the MS baseline. Adding the notation features (line 8 improves the results and leads to improvements with stronger significance. The RBF kernel makes feature weight analysis impossible, so we cannot perform further error analysis.

5.4 Comparing the Learned Models and Strength Indicators from the Chess Literature

There are many conjectures from instructional chess literature and results from psychological research about various aspects of player behavior. In this section, we compare these to the predictions made by our supervised expertise model. In Table 6, we list selected weights from the best classification model (line 3 in Table 2). We opt for analzying the classifier rather than the ranker as we find the former more directly interpretable.

Long-Term vs Short-Term Planning. The SVM model reflect the short-term nature of the amateurs' thoughts in several ways: (i) Amateurs focus on specific moves rather than long-term plans, and thus, terms like *capture* and *take* are deemed predictive for lower ratings. (ii) Amateurs often think piece-specific (Silman, 1999), particularly about moves with minor pieces (*bishop* or *knight*), and these terms receive high negative weights, pointing to lower ratings. Related to this, Reynolds (1982) observed that amateurs often focus on the current location of a piece, whereas experts mostly consider possible future locations. The SVM model learns this by weighting bigrams of the form * *on*, where * is a piece, as indicators for low ratings. (iii) Many terms related to elementary tactics (e.g., *pin*, *fork*) indicate lower-rated players, whereas terms relating to tactical foresight (e.g., *threat*, *danger*, *stop*) as well as positional terms (e.g., *weakness*, *light* and *dark* squares, *variation*) indicate higher-rated players.

Emotions. A popular and wide-spread claim is that weaker chess players often lose because they are too emotionally invested in the game and thus get carried away (e.g., Cleveland (1907), Silman (1999)). We experimented with a sentiment feature, counting polar terms in the annotations using a polarity lexicon (Wilson et al., 2005). However, this feature did not improve our results.

Manual examination of features expressing sentiment reveals that both amateurs and experts use subjective terms. We note that the vocabulary of subjective expressions is very constrained for stronger

players while it is open for weaker ones. Expert players tend to assess positions as *winning* or *losing* for a side, whereas weaker players tend to use terms such as *like* and *hate*. Both terms are identified as indicators of the respective strength class in our models. Other subjective assessments (e.g., *good* and *bad*) are divided among the classes. Emotional tendencies of amateurs can also be observed through objective indicators. As discussed above, stronger players talk about the game with a more distanced view, often referring to their opponent by their color (*white* or *black*) rather than using the pronoun *he*. Lower-rated players appear to use terms indicating competitions more frequently, such as *fight*.

Confidence. Silman (1999) argues that weaker players lack confidence, which leads to them losing track of their own plans and to eventually follow their opponent's will (often called *losing the initiative*). This process is indeed captured by our trained models. Terms of high confidence (such as *know*, *will*) are weighted towards the stronger class, whereas terms with higher uncertainty (such as *thinking*, *believe*, *maybe*, *hoping*) indicate the weaker class. This observation is in line with findings on self-assigned confidence judgments of chess players (Reynolds, 1992). The sets of terms expressing certainty and uncertainty, respectively, are small in our dataset, so weights for most terms can be learned directly on the n-grams.

Time Management. It has been suggested that deficiencies in time management are responsible for many losses at the amateur level, particularly in fast games (e.g., blitz chess, where each player has 5 minutes to complete the game), for example due to poor pattern recognition skills of beginners (Calderwood et al., 1988). In the trained models, we see that the term *time* itself is actually considered a good indicator for stronger players. *Time* is often used to signify number of moves. So, when used on its own, *time* is referring to efficient play, which is indicative of strong players. Conversely, the terms *clock* and *time pressure* are deemed good features to identify weaker players.

Chess Terminology. As shown in Section 2.1 and throughout this paper, there is a vast amount of chess terminology. We observe that frequent usage of such terms (e.g., *blunder* – a grave mistake, *tempo*, *checkmate* – experts use *mate*, *opening*, *castle*) actually indicate a weaker player. This seems counterintuitive at first, as we may expect lower-rated players to be less familiar with such terms. However, it appears that they are frequently overused by weaker players. This also holds for metaphorical terms, such as *fall* or *eat* instead of *capture*.

6 Related Work

The treatment of writer expertise in extralinguistic tasks in NLP has mostly focused on two problems: (i) retrieval of experts for specific areas – i.e., predicting the area of expertise of a writer (e.g., Tu et al. (2010; Kivimäki et al. (2013)); and (ii) using expert status in different downstream applications such as sentiment analysis (e.g., Liu et al. (2008)) or dialog systems (e.g., Komatani et al. (2003)). Conversely, our work is concerned with predicting a ranking by expertise within a single task.

Several publications have dealt with natural language processing related to games. Chen and Mooney (2008) investigate grounded language learning where commentary describing the specific course of a game is automatically generated. Commentator expertise is not taken into account in this study. Branavan et al. (2012) introduced a model for using game manuals to increase the strength of a computer playing the strategy video game *Civilization II*. Cadilhac et al. (2013) investigated the prediction of player actions in the strategy board game *The Settlers of Catan*. Our approach differs conceptually from theirs as their main focus lies on modeling concrete *actions* in the game (either predicting or learning them); our goal is to predict *player strength*, i.e., to learn to compare players among each other. Rather than explicitly modeling the game, commentary analysis aims to provide insight into specific thought processes.

Work in psychology research by Pfau and Murphy (1988) showed the quality of chess players' verbalization about positions is correlated significantly with their rating. While they use manual assessments by chess masters to determine the quality of a player's writing, our approach is to learn this distinction is automatically given the ratings.

7 Conclusion

In this paper, we presented experiments on predicting the expertise of speakers in a task using linguistic evidence. We introduced a classification and a ranking task for automatically ranking chess players by playing strength using their natural language commentary. SVM models succeed at predicting either a rating class or an overall ranking. In the ranking case, we could significantly boost the results by using chess-specific features extracted from the text. Finally, we compared the predictions of the SVM with popular claims from instructional chess literature as well as results from psychology research. We found that many of the traditional findings are reflected in the features learned by our models.

Acknowledgements

We thank Daniel Quernheim for providing his chess expertise, Kyle Richardson and Jason Utt for helpful suggestions, and the anonymous reviewers for their comments.

References

SRK Branavan, David Silver, and Regina Barzilay. 2012. Learning to win by reading manuals in a monte-carlo framework. *Journal of Artificial Intelligence Research*, 43(1):661–704.

Anais Cadilhac, Nicholas Asher, Farah Benamara, and Alex Lascarides. 2013. Grounding strategic conversation: Using negotiation dialogues to predict trades in a win-lose game. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 357–368.

Roberta Calderwood, Gary A Klein, and Beth W Crandall. 1988. Time pressure, skill, and move quality in chess. *The American Journal of Psychology*, 101(4):481–493.

José R Capablanca. 1921. *Chess Fundamentals*. Harcourt.

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

David L Chen and Raymond J Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th International Conference on Machine learning (ICML)*, pages 128–135.

Alfred A Cleveland. 1907. The psychology of chess and of learning to play it. *The American Journal of Psychology*, 18(3):269–308.

Steven J Edwards. 1994. Portable game notation specification and implementation guide.

Arpad E Elo. 1978. *The Rating of Chessplayers, Past and Present*. Batsford.

Dan Heisman. 1995. *The Improving Annotator – From Beginner to Master*. Chess Enterprises.

Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 1999. Large margin rank boundaries for ordinal regression. *Advances in Neural Information Processing Systems (NIPS)*, pages 115–132.

Ilkka Kivimäki, Alexander Panchenko, Adrien Dessy, Dries Verdegem, Pascal Francq, Hugues Bersini, and Marco Saerens. 2013. A graph-based approach to skill extraction from text. In *Proceedings of TextGraphs-8*, pages 79–87.

Kazunori Komatani, Shinichi Ueno, Tatsuya Kawahara, and Hiroshi G. Okuno. 2003. Flexible guidance generation using user model in spoken dialogue systems. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 256–263.

Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. 2008. Modeling and predicting the helpfulness of online reviews. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM)*, pages 443–452.

Eric W Noreen. 1989. *Computer Intensive Methods for Hypothesis Testing: An Introduction*. Wiley.

H Douglas Pfau and Martin D Murphy. 1988. Role of verbal knowledge in chess skill. *The American Journal of Psychology*, 101(1):73–86.

Robert I Reynolds. 1982. Search heuristics of chess players of different calibers. *The American journal of psychology*, 95(3):383–392.

Robert I Reynolds. 1992. Recognition of expertise in chess players. *The American journal of psychology*, 105(3):409–415.

Jeremy Silman. 1999. *The Amateur's Mind: Turning Chess Misconceptions into Chess Mastery*. Siles Press.

Gregg E A Solomon. 1990. Psychology of novice and expert wine talk. *The American Journal of Psychology*, 103(4):495–517.

James H Steiger. 1980. Tests for comparing elements of a correlation matrix. *Psychological Bulletin*, 87(2):245.

Yuancheng Tu, Nikhil Johri, Dan Roth, and Julia Hockenmaier. 2010. Citation author topic model in expert search. In *Proceedings of the 2010 Conference on Computational Linguistics (Coling): Posters*, pages 1265–1273.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology (HLT) and Empirical Methods in Natural Language Processing (EMNLP)*, pages 347–354.

Zipf's Law and Statistical Data on Modern Tibetan

Huidan Liu
Institute of Software,
Chinese Academy of
Sciences, Beijing,
China, 100190
huidan@iscas.ac.cn

Minghua Nuo
Institute of Software,
Chinese Academy of
Sciences, Beijing,
China, 100190
minghua@iscas.ac.cn

Jian Wu
Institute of Software,
Chinese Academy of
Sciences, Beijing,
China, 100190
wujian@iscas.ac.cn

Abstract

In this paper, a large scale modern Tibetan text corpus is built, which includes about 190 thousands documents, 67.21 million words, 93.66 million syllables in total. Based on the corpus, statistics are made in several language units in different granularities. Statistical data show that : a syllable has 3.26 letters or 2.20 super characters in average, while a sentence has 75.40 letters or 63.14 super characters. The top 10 super characters, syllables, words take up 66.3156%, 16.5556%, 24.6415% of the corpus respectively. Curves for the n-gram frequency-rank list of super chars, syllables and words are plotted. It shows that when all the n-gram phrases for $n = 1, 2, \ldots, 5$ are put together and sorted by frequency in descending order, the frequency-rank curves in log-log axes can be fitted well by a straight line for the unit of syllable and word respectively. But for the unit of super character, we didn't find a curve that can be fitted well enough by a straight line even if we combine all the n-grams for $n = 1, 2, \ldots, 10$.

1 Introduction

The statistical property is the natural property of a language. In recent tens of years, people made statistical analysis on Tibetan characters or syllables. But it's difficult to make statistics on larger language units such as word and n-gram word phrases, especially on a large scale corpus. There are two reasons resulting in the difficulty. First, as Tibetan is a resource poor language, it's hard to build a large scale Tibetan text corpus. Second, Tibetan word segmentation technology is not well developed even until now.

In this paper, we report our word on the statistics on Tibetan based on the language units such as character, syllable, word and their n-gram pairs on a large scale corpus. The remainder of the paper is organized as follow. Tibetan language units are introduced in Section 2. We recall the related work in Section 3. In Section 4, the methods which is used to build the corpus and to segment Tibetan text into language units are described in detail. We make statistics on the corpus and list the most frequency Tibetan language units and test Zipf's law respectively in Section 5. Section 6 concludes this paper.

2 Language Units of Tibetan

Generally speaking, Tibetan is a alphabetic writing system. But there is a unit larger than letter but smaller than syllable, which is different from other language such as English and Chinese. Meanwhile, people have used different terms (in English) to express the same unit or the same term to express different units. So we must make a clarification in this Section.

2.1 Letter, Character and Super Character

There are 30 consonants and 4 vowel signs in modern Tibetan. Several other consonants and vowel signs are also used in Tibetan text to transliterate Sanskrit script. There are only 4 vowel signs (writing) for the vowels (reading) /e/, /i/, /o/ and /u/, but there isn't any signs for the vowel /a/, so every consonant has

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 322–333, Dublin, Ireland, August 23-29 2014.

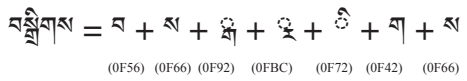


Figure 1: Tibetan encoding schema with small unit used in ISO/IEC 10646.

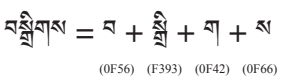


Figure 2: Tibetan encoding schema with large unit used in GB/T 20542.

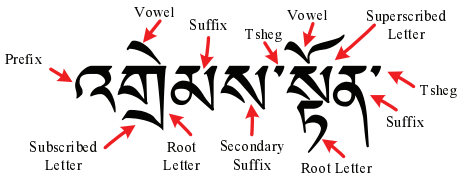


Figure 3: Structure of a Tibetan word.

ཁ་སང་མི་སྡུག་པོ་འདིའི་ཁང་པ་གོང་ཆེན་པོ་ཞིག་གིས་ཉོས་སོང་།									
ཁ་སང་	མི་	སྡུག་པོ་	འདིའི་	ཁང་པ	གོང་ཆེན་པོ་	ཞིག་	གིས་	ཉོས་སོང་	།
Yesterday	man	rich	this	house	expensive	an	bought	did	.
Yesterday this rich man bought an expensive house.									

Figure 4: A Tibetan sentence.

an inherent vowel /a/. Other vowels can be indicated using a variety of diacritics which appear above or below the main letter. Each of these consonants and vowel signs is called a "letter".

In Tibetan encoding schema used in ISO/IEC 10646 and Unicode standard (Consortium, 2013), each Tibetan consonant has two or even more code points to denote its normal form or subjoined form, or other variant forms which are only used in very special context. Each variant form is called a "character" corresponding to a code point in ISO/IEC 10646. In Figure 1, seven characters form a Tibetan syllable. Note that three consonants and a vowel sign are clustered.

Different from the encoding schema with small unit used in ISO/IEC 10646, in Chinese notional standard GB/T 20542 and GB/T 22238 on Tibetan coded character set and some legacy Tibetan encodings, another encoding schema is used. In this schema, the cluster of consonants and vowel sign in Figure 1 is assigned only one code point. Figure 2 shows the schema. The encoding unit shown in Figure 2 is called "字丁 " (Zi Ding) in Chinese but the Chinese term doesn't have an exact English translation, and we call it "super character" or "super char" briefly in this paper.

2.2 Syllable, Word and Sentence

A syllable contains one or up to seven character(s). Syllables are separated by a marker known as "tsheg", which is simply a superscripted dot. People sometime use the Chinese term "字 " (Zi, exactly a character in Chinese script) to denote "syllable". The term "字 " is often translated to "word" in English. But "word" mainly used to express a larger language unit as an item in the vocabulary. So we use the term "syllable" in this paper, and take "word" as a larger language unit which is made up of one or more syllables and has meanings.

Note that in Tibetan "tsheg" is used as the delimiter between two syllables. But there is no another delimiter to mark the boundary between two words. Thus there is a lack of word boundaries in Tibetan. Figure 3 shows the structure of a Tibetan word which is made up of two syllables and means "show" or "exhibition".

In Tibetan text, some monosyllable words, including "འི", "ས", "ར", "འང", "འམ", "འོ" (We call them abbreviation markers (AM) in this paper), can glue to the previous word without a syllable delimiter "tsheg", which produce many abbreviated syllables. For example, when the genitive case word "འི" follows the word "རྒྱལ་པོ" (king), we don't put a "tsheg" between them and get the fused form "རྒྱལ་པོའི" (king[+genitive]). The existence of abbreviated syllables contributes to the difficulty to segment Tibetan sentence into words.

Tibetan sentence contains one or more phrase(s), which contain one or more words. Another marker known as "shed" indicates the sentence boundary, which looks like a vertical pipe. Figure 4 shows a Tibetan sentence and its translation in English.

3 Related Work

In the early 1930s, G. K. Zipf pointed out a statistical feature of large language corpora (both written texts and speech streams) which, remarkably, is observed in many languages, and for different authors and styles (Zipf, 1935). He noticed that the number of words $w(n)$ which occur exactly n times in a language corpus varies with n as $w(n) \sim 1/n^{\gamma}$, where the exponent is close to 2, which results in the well known Zipf's law. The general form of Zipf's law states that:

$$y = f(r) = \frac{C}{r^{\alpha}} \tag{1}$$

where α is a positive parameter close to 1.

Zipf showed that, by and large, his law held for words, syllables and morphemes. Consequently, it is natural to ask if the law also holds for pairs of words. Egghe devised a mathematical argument that it, in fact, does not, but that the exact relation can be approximated by a power law (Egghe, 1999). He extended his investigations to parts of words, namely to the study of N-grams (Egghe, 2000).

Zipfs law was the source of a lively debate related to the structure of DNA. It was claimed (Mantegna et al., 1994) that Zipf's law shows the difference between coding and non-coding DNA as non-coding (so-called junk) DNA fits Zipf's law much better than coding DNA. This would mean, according to the authors, that non-coding regions of DNA may carry new biological information. Yet, this does not mean that junk DNA is a kind of language. Other scientists (Chatzidimitriou-Dreismann et al., 1996), however, have shown that this distinction is not universal and lacks all biological basis.

Zipf's law has been tested on the Internet. It turned out that popularity of Internet pages is described according to Zipf's law. This fact can be used to design better cache tables (Masaki and Takahashi, 1998; Breslau et al., 1998; Adamic and Huberman, 2002). Zipf's studies on city sizes still lead to new developments in geographical and economical studies (Gabaix, 1999a; Gabaix, 1999b; Okuyama et al., 1999; Ioannides and Overman, 2003; Soo, 2005; Soo, 2007).

Back to text, Li (1992) found that the distribution of word frequencies for randomly generated texts is very similar to Zipf's law observed in natural languages such as English (Li, 1992). Ha et al. (2002) investigated the law for two languages English and Mandarin and for n-gram word phrases as well as for single words. The law for single words is shown to be valid only for high frequency words. However, when single word and n-gram phrases are combined together in one list and put in order of frequency the combined list follows Zipf's law accurately for all words and phrases, down to the lowest frequencies in both languages. The Zipf curves for the two languages are then almost identical (Ha et al., 2002).

In recent years, researchers also made statistics on Tibetan. Jiang and Dong (1994) made statistics on the length and different structural mode of Tibetan syllables, and counted up the number of initial clusters and finals of Tibetan syllables, as well as the number of Tibetan letters at different positions in syllables (Jiang and Dong, 1994; Jiang and Dong, 1995). In a further research Jiang (1998; Jiang and Kong (2006; Jiang and Long (2010) , they made statistics on Tibetan letters, and found that the 1th order and 2nd order entropy of Tibetan is 3.9913 bits and 1.2531 bits resplectively (Jiang, 1998), while on super character they are 4.82 and 3.12 (Jiang and Kong, 2006; Jiang and Long, 2010). Wang and Chen (2004) made similar research to calculate the frequency and information entropy of Tibetan character and syllable based on a corpus of $20,000,000$ characters, and discovered that the most frequent 703 Tibetan syllables cover 90% of the corpus (Wang and Chen, 2004). She also presented the research on the frequency-rank relation of Tibetan super character and syllable, and found that the distributions follow Zipf's law too (Wang, 2004). But no further research is reported on whether she tests Zipf's law on larger language units of Tibetan. Other researchers also made statistics on Tibetan syllable's structural mode based a static corpus such as syllable list or dictionary (Gao and Gong, 2005; Ai et al., 2009). Lu et al. (2003) presented the theories and approaches to calculate the frequencies of Tibetan characters, pieces, syllables and words based on a large scale Tibetan corpus including about $40,000,000$ syllables (Lu et al., 2003). However, a large part of the corpus they used are Buddhist literatures and the work can't be done well without a pragmatic Tibetan word segmentation tool (Chen et al., 2003a; Chen et al., 2003b; Jiang, 2006; Jiang and Kong, 2006; Sun et al., 2009; Sun et al., 2010; Lu and Shi, 2011; Liu et al., 2012a).

At present, people already find methods to build a large scale corpus from Tibetan web sites with low cost. Liu et al. (2012b) presented their method to extract the title, content, author and other useful information of articles from several news and broadcasting web sites (Liu et al., 2012b). It's not a difficult work to implement a pragmatic Tibetan word segmentation tool based on the former researches (Chen et al., 2003a; Chen et al., 2003b; Sun et al., 2009; Sun et al., 2010; Liu et al., 2012a) So it's time to make statistics on the frequency distribution of larger language units such as word and n-gram word phrase for Tibetan to see whether they follow Zipf's law.

4 Methods and Corpus

We present our methods to build the corpus and to segment Tibetan text into different units mentioned above.

4.1 Building a Large Scale Tibetan Text Corpus

Previously Liu et al. (2012b) proposed an approach to build a large scale text corpus for Tibetan natural language processing. We adopt the method to build our corpus. we crawled eight Tibetan websites which mainly focus on news and broadcastings. Topic pages but hub pages are selected with a rule based method by checking the url. We analysed the layout structure mode of each web site and built templates to extract topic title, publishing date, author, topic content and some other topic related informations.

Consequently, a large scale Tibetan text corpus is built, which includes about 190 thousands documents, 67.21 million words, 93.66 million syllables and 265 million super characters in total. The sources and scales in different units are shown in Table 1.

♯	source	♯document	♯sentence	♯word	♯syllable	♯super character
1	http://tb.chinatibetnews.com	74,632	1,419,967	26,648,803	37,633,467	108,010,715
2	http://tb.tibet.cn	13,348	331,022	4,288,187	5,872,524	16,388,242
3	http://ti.gzznews.com	8,084	281,405	3,518,918	4,763,097	13,301,408
4	http://ti.tibet3.com	26,631	725,669	9,186,980	12,634,804	35,595,345
5	http://tibet.people.com.cn	29,797	833,221	9,323,838	12,908,542	35,328,443
6	http://www.qhtb.cn	20,616	575,242	7,908,508	10,913,097	31,200,465
7	http://www.tibetcnr.com	9,559	278,681	3,272,274	4,624,878	13,114,130
8	http://xizang.news.cn	7,707	187,423	3,062,419	4,307,175	12,258,911
Total		190,374	4,632,630	67,209,927	93,657,584	265,197,659

Table 1: the sources and scales of the corpus.

It's a heavy task to manually classify those document into domains. However, we still can get the domain information for a certain subsets of the corpus. For some web sites listed above, we can get the domain information from the URL of each web page. For instance, the URL "http://tb.chinatibetnews.com/xzmeishi/2011-12/05/content_831210.htm" shows it belongs to a column called "xzmeishi". so it must be a page about Tibetan foods, because "xz" is the abbreviated form of Chinese word "xizang" (西藏), which means the Tibetan Autonomous Region, while "meishi" means "delicious food". So we can classify the documents in the corpus into domains. Table 2 and 3 list the domains of subsets of the documents from two web sites named "China Tibet News" and "Tibetan's web of China" respectively. Obviously, a large part of the documents in the corpus are news as expected, because nearly all of the 8 web sites are hold by news agencies or radio stations.

4.2 Methods to Segment Tibetan Text

As described in section 2, there is a delimiter between two Tibetan syllables. So we can segment the text into syllables by adding segmentation mark after the delimiter. The encoding schema can be used to segment text into smaller language units.

The challenge lies in the word segmentation. With a similar method to those methods proposed by other researchers (Chen et al., 2003a; Chen et al., 2003b; Jiang and Kong, 2006; Sun et al., 2009; Sun et al., 2010; Liu et al., 2012a), we implemented a segmenter. As mentioned in Section 2.2, some monosyllable words can glue to the previous word without a syllable delimiter "tsheg", which produce many

Order	Domain	♯document	(%)	♯sentence	(%)	♯syllable	(%)
1	Art	3,240	4.76	112,642	8.71	1,265,914	4.40
2	Finance & Economy	712	1.05	12,477	0.96	314,698	1.09
3	History & Geometry	2,897	4.25	19,627	1.52	283,621	0.98
4	News	25,247	37.08	576,842	44.59	14,753,178	51.23
5	Picture	12,732	18.70	51,088	3.95	766,895	2.66
6	Politics & Law	3,230	4.74	63,437	4.90	1,708,839	5.93
7	Rural Life	2,402	3.53	35,535	2.75	871,406	3.03
8	Social Life	1,153	1.69	9,881	0.76	233,454	0.81
9	Special Issues	9,986	14.67	268,003	20.72	6,499,488	22.57
10	Technology & Education	1,988	2.92	38,321	2.96	825,395	2.87
11	Tibetan Buddhism	1,983	2.91	48,832	3.77	569,756	1.98
12	Tibetan Food	215	0.32	2,963	0.23	35,365	0.12
13	Tibetan Medicine	720	1.06	36,676	2.84	303,012	1.05
14	Tour	1,588	2.33	17,296	1.34	367,226	1.28
15	Total	68,093	100.00	1,293,620	100.00	28,798,247	100.00
Total		68,093	100.00	1,293,620	100.00	28,798,247	100.00

Table 2: Domains of a subset of the documents from ”China Tibet News”.

Order	Domain	♯document	(%)	♯sentence	(%)	♯syllable	(%)
1	Art	92	0.35	3,021	0.45	44,727	0.43
2	Culture	885	3.40	109,749	16.18	980,554	9.32
3	Economy	78	0.30	7,749	1.14	124,101	1.18
4	Education	15	0.06	695	0.10	13,919	0.13
5	Music	323	1.24	3,169	0.47	31,791	0.30
6	News	24,055	92.45	519,576	76.61	8,783,626	83.50
7	Photo	80	0.31	2,548	0.38	35,982	0.34
8	Policy	116	0.45	7,062	1.04	121,930	1.16
9	Politics	124	0.48	7,668	1.13	137,538	1.31
10	Tibetan Medicine	107	0.41	11,417	1.68	162,557	1.55
11	Tour	145	0.56	5,563	0.82	82,443	0.78
Total		26,020	100.00	678,217	100.00	10,519,168	100.00

Table 3: Domains of a subset of the documents from ”Tibetan’s web of China”.

abbreviated syllables. So Tibetan has a significant number of complex words where the sounds have been synthesized due to internal sandhi something like Sanskrit. As some of those abbreviated syllables can also be used as normal syllables, they lead to considerable problem in Tibetan word segmentation. So in the first step, we analyse the structure of each syllable in the sentence, and break them into normal syllables and abbreviated mark candidates and take them as the basic units (unbreakable units). Then, in the second step, some special case-auxiliary words (which are all monosyllable words) are used as separators to break the sentence into blocks. Consequently, both the forward maximum matching method and backward maximum matching method are used to segment each block into words. Mean while, it detects ambiguities by bidirectional segmentation, and makes disambiguation with word frequency. A previous research shows that the precision of this method reaches 96.98% (Liu et al., 2012a). The following example shows the main procedure of the method.

Input: དཚོས་སྤྱི་ཚོགས་རིང་ལུགས་ཀྱི་སྤྱི་ལ་དབང་བའི་ལམ་ལུགས་དང་ཇོག་བསྟུན་ཐོག་སྤྲོད་ཀྱི་ཐ་དོན་མཐའ་འཇྱོངས་བྱས་ཡོད།

Translation: We have always followed the principles of socialist public ownership and distribution according to work.

Step 1: ད་ (ཚོ་ས་) སྤྱི་ ཚོགས་ རིང་ ལུགས་ ཀྱི་ སྤྱི་ ལ་ དབང་ (བ་ཞི་) ལམ་ ལུགས་ དང་ ཇོག་ བསྟུན་ ཐོག་ སྤྲོད་ ཀྱི་ ཐ་ དོན་ མཐའ་ འཇྱོངས་ བྱས་ ཡོད །

Step 2: (ད་ ཚོ་ ས་ སྤྱི་ ཚོགས་ རིང་ ལུགས་) ཀྱི་ (སྤྱི་ ལ་ དབང་ བ་ ཞི་ ལམ་ ལུགས་ དང་ ཇོག་ བསྟུན་ ཐོག་ སྤྲོད་) ཀྱི་ (ཐ་ དོན་ མཐའ་ འཇྱོངས་ བྱས་ ཡོད་) །

Step 3: (ད་ ཚོ་) (ས་) (སྤྱི་ ཚོགས་ རིང་ ལུགས་) ཀྱི་ (སྤྱི་ ལ་ དབང་ བ་ ཞི་ ལམ་) (ལུགས་) (དང་) (ཇོག་) (བསྟུན་) (ཐོག་) (སྤྲོད་) ཀྱི་ (ཐ་ དོན་) (མཐའ་ འཇྱོངས་) (བྱས་ ཡོད་) ।

Output: དཚོ་ ས་ སྤྱི་ཚོགས་རིང་ལུགས་ ཀྱི་ སྤྱི་ལ་དབང་བའི་ལམ་ ལུགས་ དང་ ཇོག་ བསྟུན་ ཐོག་ སྤྲོད་ ཀྱི་ ཐ་དོན་ མཐའ་འཇྱོངས་ བྱས་ཡོད ।

Note that, in step 1, two abbreviated syllable candidates are found and given in parentheses. In step 2, the two occurrences of the case-auxiliary word ཀྱི་ break the sentence into several blocks, and each block is segmented into words consequently in step 3. In this paper, as we mainly focus on Tibetan text, so in the segmentation, all Latin words, Latin numbers, Chinese phrases, Tibetan alphabetic numbers such as " ༦༣༣༡༨༩༩ " (6331089) and so on are all replaced by place-holders.

4.3 Counting and Calculation

The SRI Language Modelling Toolkit (SRILM) (Stolcke and others, 2002) is used to count the frequencies in our work.

5 Statistical Data and Analysis

In this section, we show the statistical data and check whether the frequency-rank on the units of super character, syllable and word follows Zipf's law respectively. As there isn't many enough items in the frequency list, we won't check it on the units of letter and character. For the other units, the number of occurrences of each n-gram is listed in Table 4.

unit	super char	syllable	word
unigram	265,197,659	93,657,584	67,209,927
bigram	260,565,029	89,024,954	62,577,297
trigram	256,022,772	84,521,746	58,085,930
4-gram	251,638,774	80,194,075	54,051,877
5-gram	247,285,278	76,155,897	50,242,999
Total	1,280,709,512	423,554,256	292,168,030

Table 4: Number of occurrences of each Tibetan n-gram in different units in the corpus.

5.1 Letter Frequency

In total, Tibetan 83 letters are used in the corpus, of which 39 letters are consonants and 8 letters are vowel signs. Letter ཹ and ཿ didn't occur in the corpus, which means people might prefer to use two letters to spell each of them. The other are Tibetan punctuations and signs. There are also 200 non Tibetan characters used in the corpus. The 47 letters and the two delimiters are listed in Table 5. The character "P", "C" and "V" in the table denote "Punctuation", "Consonant" and "Vowel" respectively. The "theg" shares 23.45% of the corpus while the "thed" shares 1.33%, which shows that a syllable has 3.26 letters (not including the "theg" itself), while a sentence has 75.40 letters in average. 4 of the 8 vowels occur frequently while the other 4 vowels are rarely used. The 2 punctuations share 24.7762% of the corpus while 4 vowels in modern Tibetan share 16.0805%, and the 30 consonants in modern Tibetan share 58.3918%. All these 36 letters share 99.2485% of the corpus in total. The other 4 vowels and 9 consonants which is used to transliterate Sanskrit script are rarely used. They share only 0.0437%. Other Tibetan signs and non Tibetan characters share 0.7078%.

5.2 Character Frequency

There are 119 Tibetan characters used in the corpus in total, including Tibetan punctuations and signs, but Tibetan number is replaced with a place-holder. As there are 83 letters as described in the former subsection, the other 36 characters are the second or third forms of Tibetan consonants. As the frequency of Tibetan character is seldom a concerned issue, we don't make any further remarks on it.

5.3 Super Character Frequency

There are 1,466 super characters used in the corpus in total. The topmost frequently occurred super characters and n-gram super char phrases for $n = 2, 3$ are listed in Table 6. As expected, the "theg" is the most frequently occurred one when we take it as a super character, which shares 31.22%. It indicates that a syllable is formed by 2.20 super characters in average. The "theg" shares 1.5837%, which indicates that a sentence has 63.14 super characters in average.

♯	letter	♯ occur	rate(%)	cum.rate(%)	♯	letter	♯ occur	rate(%)	cum.rate(%)
P01	་	82,818,775	23.4500	23.4500	C20	ཚ	1,788,360	0.5064	96.3522
C01	ས	25,967,545	7.3527	30.8027	C21	ཛ	1,688,121	0.4780	96.8302
C02	ག	21,589,015	6.1129	36.9155	C22	ཀྵ	1,442,603	0.4085	97.2386
C03	ར	18,211,934	5.1567	42.0722	C23	ཞ	1,193,175	0.3378	97.5765
V01	ེ	17,755,843	5.0275	47.0998	C24	ཤ	1,110,743	0.3145	97.8910
V02	ི	17,663,843	5.0015	52.1012	C25	ཟ	1,106,366	0.3133	98.2043
C04	ད	16,796,414	4.7559	56.8571	C26	ཥ	1,074,769	0.3043	98.5086
C05	ན	14,320,083	4.0547	60.9118	C27	ཉ	962,770	0.2726	98.7812
C06	བ	14,311,260	4.0522	64.9640	C28	ཨ	932,081	0.2639	99.0451
C07	ཡ	13,969,128	3.9553	68.9194	C29	ཪ	411,338	0.1165	99.1616
V03	ུ	12,067,901	3.4170	72.3364	C30	ཧ	306,863	0.0869	99.2485
C08	ོ	11,706,716	3.3147	75.6511	V05	ཱ	112,528	0.0319	99.2803
C09	ལ	10,652,623	3.0163	78.6674	C31	ཀྵ	13,398	0.0038	99.2841
C10	ད	10,252,221	2.9029	81.5703	C32	ཫ	11,482	0.0033	99.2874
V04	ཻ	9,304,303	2.6345	84.2048	C33	ཬ	7,979	0.0023	99.2896
C11	མ	8,003,478	2.2662	86.4709	C34	ཋ	6,617	0.0019	99.2915
C12	ཏ	6,484,634	1.8361	88.3070	C35	ཀྱ	945	0.0003	99.2918
C13	ཀ	4,788,018	1.3557	89.6628	V06	ཹ	689	0.0002	99.2920
P02	།	4,683,745	1.3262	90.9890	V07	ཷ	288	0.0001	99.2920
C14	ཤ	3,621,446	1.0254	92.0144	C36	ཀྲ	159	0.0000	99.2921
C15	ཅ	3,220,311	0.9118	92.9262	C37	ཀྱི	128	0.0000	99.2921
C16	ཆ	3,082,972	0.8729	93.7991	V08	ཻ	64	0.0000	99.2921
C17	ཞ	2,696,958	0.7636	94.5628	C38	ཁྱ	55	0.0000	99.2922
C18	ཐ	2,307,810	0.6535	95.2162	C39	ཨ	38	0.0000	99.2922
C19	ཆ	2,223,533	0.6296	95.8458	Total		353,171,951		100.00

Table 5: Frequency of Tibetan letters used in the corpus.

♯	Unigram	♯occur	rate(%)	Bigram	♯occur	rate(%)	Trigram	♯occur	rate(%)
1	་	82,794,773	31.2200	ས་	15,628,059	5.9978	གས་	3,507,130	1.3699
2	ས	17,136,507	6.4618	ད་	11,178,538	4.2901	དུ་	2,344,655	0.9158
3	ད	13,283,592	5.0089	ི་	7,913,908	3.0372	་དུ་	2,311,335	0.9028
4	ག	12,677,652	4.7805	་ཐ	6,137,354	2.3554	་ཡ་	1,829,894	0.7147
5	་	11,999,418	4.5247	་	6,108,586	2.3444	དས་	1,699,226	0.6637
6	ི	10,582,545	3.9904	་ཅ	4,972,884	1.9085	་ཀྱི་	1,353,391	0.5286
7	བ	9,387,744	3.5399	ར་	4,892,373	1.8776	་ཡ་ཡི	1,284,801	0.5018
8	ཡ	6,335,567	2.3890	་ད	4,705,897	1.8060	ཡ་ཡི་	1,282,074	0.5008
9	མ	5,899,668	2.2246	ལ་	4,596,516	1.7641	ི་ས་	1,251,081	0.4887
10	ཅ	5,769,971	2.1757	་ག	4,268,149	1.6380	ས་ཡ	1,244,612	0.4861
Total		175,867,437	66.3156	Total	70,402,264	27.0191	Total	18,108,199	7.0729

Table 6: The topmost frequently occurred super characters and n-gram super char phrases.

The frequency-rank curves of the n-gram for $n = 1, 2, 3, 4, 5$ are plotted with log-log axes in Figure 5. A straight line with $slop = -1.0$ is also plotted in the figure. It's obvious that the curves don't follow Zipf's law so exactly. The high frequency parts of the curves follow Zipf's law at large, but as the rank increases the curves have more rapid decreases than a linear curve with slop $= -1.0$ when the rank > 100. However, we still found that the curve becomes more straight when the n increases.

Similar to Ha et al. (2002), we also combine the frequency list of the n-grams for all $n = 1, 2, \ldots, 5$ together in one list and put in order of frequency. The frequency-rank curve is plotted in Figure 6. A straight line with slope $= -1.0$ is also plotted in the figure, which shows that there are large gaps between

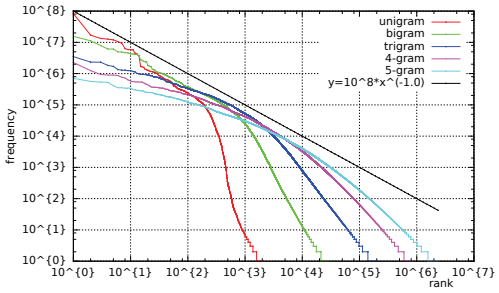


Figure 5: Frequency-rank of super chars and their n-grams.

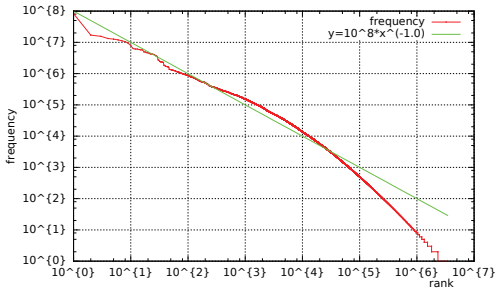


Figure 6: Frequency-rank of combined super char n-gram list.

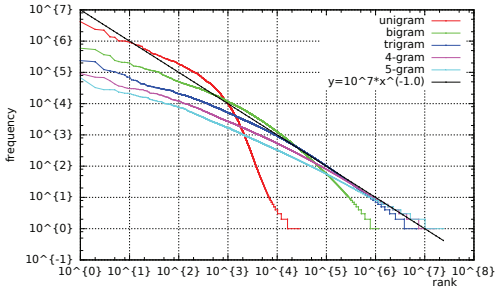


Figure 7: Frequency-rank of syllables and syllable n-grams.

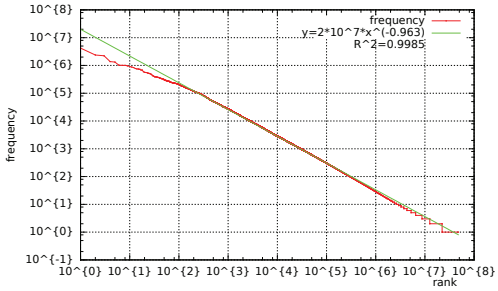


Figure 8: Frequency-rank of combined syllable n-gram list.

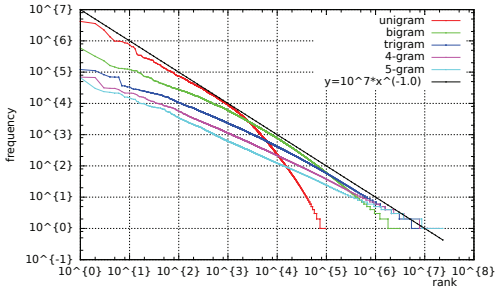


Figure 9: Frequency-rank of words and word n-grams.

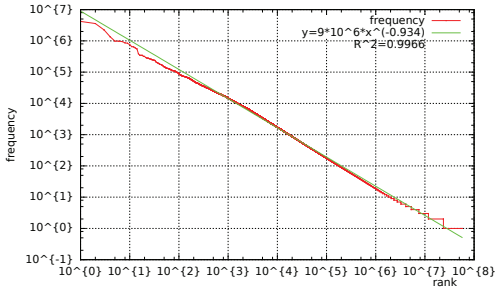


Figure 10: Frequency-rank of combined word n-gram list.

the curve and the line. Obviously it doesn't follow Zipf's law well.

5.4 Syllable Frequency

There are 27,546 syllables and 200 other characters occurred in the corpus in total. The topmost frequently occurred syllables and n-gram syllable phrases for $n = 2, 3$ are listed in Table 7. As expected, the "thed" is the most frequently used unigram when we take is as a syllable. It shares 4.4843% of the corpus. Most of the top 15 unigrams are case auxiliary words (monosyllable word), including གིས་ , ལ་ , ལས་ , ནས་ , གྱི་ and གི་ . The conjunction དང་ , the two nominalization markers པ་ and བ་ are also in the top 10 list. The top 10 syllables take up 16.5556% of the corpus.

The frequency-rank curves of the n-gram for $n = 1, 2, 3, 4, 5$ are plotted with log-log axes in Figure 7. A straight line with $slop = -1.0$ is also plotted in the figure , which shows that the curves don't follow Zipf's law very exactly. The high frequency parts of the curves when $n = 1, 2$ follow Zipf's law at large, but as the rank increases the curves have more rapid decreases than a linear curve with slop $= -1.0$ when the rank > 1000 and the rank > 10000 respectively. The curve becomes more straight when the n increases, and becomes almost straight lines when $n = 3, 4, 5$.

We also combine the frequency list of the n-grams for all $n = 1, 2, \ldots, 5$ together in one list and

♯	Unigram	♯occur	rate(%)	Bigram	♯occur	rate(%)	Trigram	♯occur	rate(%)
1	།	4,199,896	4.4843	དང་ །	593,668	0.6669	ས་ དང་ །	237,068	0.2805
2	དང་	2,370,981	2.5315	ས་ དང་	537,089	0.6033	ས་ རེད་ །	219,016	0.2591
3	ས་	2,233,002	2.3842	ས་ །	367,725	0.4131	རང་ སྐྱོང་ ལྟོ་རས་	104,505	0.1236
4	ཀྱི་	1,377,206	1.4705	རེད་ །	345,059	0.3876	ཡོད་ ས་ རེད་	92,475	0.1094
5	བའི་	1,319,052	1.4084	ཡོད་ །	241,152	0.2709	བྱེད་ ས་ དང་	86,964	0.1029
6	ལ་	1,023,287	1.0926	ས་ རེད་	222,771	0.2502	བ་ དང་ །	82,532	0.0976
7	བ་	1,008,926	1.0772	བ་ དང་	209,657	0.2355	བྱེད་ ས་ །	58,133	0.0688
8	ལས་	1,007,539	1.0758	ཡོད་ ས་	209,095	0.2349	ཡོད་ ས་ དང་	56,153	0.0664
9	ནས་	965,728	1.0311	བྱེད་ ས་	205,235	0.2305	ཞུ་ ཡོན་ ལྷན་	55,173	0.0653
10	བྱེད་	915,663	0.9777	བ་ །	198,682	0.2232	ཡོན་ ལྷན་ ཁང	54,168	0.0641
Total		15,505,617	16.5556	Total	2,931,451	3.2928	Total	992,019	1.1737

Table 7: The topmost frequently occurred syllables and n-gram syllable phrases.

put in order of frequency. The frequency-rank curve is plotted in Figure 8. A fitting straight line $y = 2 \times 10^7 \times x^{-0.963}$ with $R^2 = 0.9985$ is also plotted in the figure, which shows that the curve can be well fitted by the line. Thus, it follows Zipf's law.

5.5 Word Frequency

♯	Unigram	♯occur	rate(%)	Bigram	♯occur	rate(%)	Trigram	♯occur	rate(%)
1	།	4,199,896	6.2489	དང་ །	593,286	0.8258	ཡོད་ས་ རེད་ །	83,110	0.1237
2	ནི་	3,580,891	5.3279	རེད་ །	319,479	0.4447	ལོ་ ནི་ ཟླ་	36,087	0.0537
3	དང་	2,241,125	3.3345	ཡོད་ །	232,155	0.3231	བྱེད་ས་ དང་ །	33,574	0.0500
4	ཀྱི་	1,357,874	2.0203	ས་ ནི་	163,093	0.2270	ཡོད་ས་ དང་ །	28,562	0.0425
5	ལ་	982,161	1.4613	༄༅ །	145,563	0.2026	བྱས་ ཏེ་ །	28,387	0.0422
6	ར་	977,484	1.4544	བྱེད་ས་ ནི་	132,517	0.1845	རྩོལ་སྤྱིག་ འགན་ཁྲ་བ་ །	27,707	0.0412
7	ནས་	949,750	1.4131	བོད་ ཀྱི་	131,176	0.1826	གཞིགས་ ན་ །	26,523	0.0395
8	གྱི་	863,061	1.2841	བྱས་ ཏེ་	116,202	0.1617	བྱས་ས་ རེད་ །	25,829	0.0384
9	གི་	754,281	1.1223	བྱས་ ནས་	110,635	0.1540	ལྟོ་རས་ ཡོངས་ ཀྱི་	25,418	0.0378
10	ས་	654,987	0.9745	གྲུབ་མཐོ་ ནི་	96,640	0.1345	བྱས་ ཡོད་ །	24,786	0.0369
Total		16,561,510	24.6415	Total	2,040,746	2.8406	Total	339,983	0.5058

Table 8: The topmost frequently occurred words and n-gram word phrases.

There are 96,296 words(including Tibetan punctuations, signs) used in the corpus in total. The topmost frequently occurred words and n-gram word phrases for $n = 2, 3$ are listed in Table 8. As expected, the "thed" is the most frequently used unigram when we take is as a word. It shares 6.2489% of the corpus. Almost all of the top 10 unigrams are auxiliary case words (monosyllable word), including ནི་ , ཀྱི་ , ལ་ , ར་ , ནས་ , གྱི་ , གི་ and ས་ . The top 10 words take up 24.6415% of the corpus.

The frequency-rank curves of the n-gram for $n = 1, 2, 3, 4, 5$ are plotted with log-log axes in Figure 9. A straight line with $slop = -1.0$ is also plotted in the figure , which shows that the curves don't follow Zipf's law very exactly. The high frequency part of the curve when $n = 1$ follows Zipf's law at large, but as the rank increases the curve has more rapid decreases than a linear curve with slop $= -1.0$ when the rank > 1000. The curve becomes more straight when the n increases, and becomes almost straight lines when $n = 3, 4, 5$.

We also combine the frequency list of the n-grams for all $n = 1, 2, \ldots, 5$ together in one list and put in order of frequency. The frequency-rank curve is plotted in Figure 10. A fitting straight line

$y = 2 \times 10^7 \times x^{-0.934}$ with $R^2 = 0.9966$ is also plotted in the figure, which shows that the curve can be well fitted by the line. Thus, it follows Zipf's law.

5.6 Further Discussion

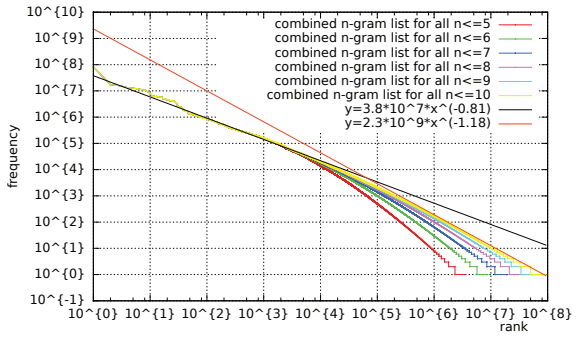


Figure 11: Frequency-rank of combined Tibetan word n-gram lists for $n <= 5, 6, 7, 8, 9, 10$.

Comparing the curves in Figure 5, 7 and 9, we find that the curves with the same n for all $n = 1, 2, 3, 4, 5$ become more straight when the granularity becomes larger. It's similar in the combined n-gram curves in Figure 6, 8 and 10. As it's shown that the two combined n-gram frequency lists for all $n <= 5$ on syllable and word follow Zipf's law well. So, the question is that whether we can find a larger M, which for the combined n-gram list for all all $n < m$, the frequency-rank curve in log-log axes is straight enough. To find the M, the frequency-rank curves for the combined n-gram super character lists for $m = 5, 6, 7, 8, 9, 10$ are plotted respectively in Figure 11. From the figure, we see that the head parts of the curves are overlapped, which correspond to the high frequency parts of the combined n-gram lists, while the tail parts of the curves are divergent. As the m increases, the tail part of the curve becomes closer to the straight line $y = 3.8 \times 10^7 \times x^{-0.81}$. This mainly results from that the frequency of the n-gram decreases when the n increase, and the low frequency part of the combined n-gram list includes more n-grams. However, the two straight lines of $y = 3.8 \times 10^7 \times x^{-0.81}$ and $y = 2.3 \times 10^9 \times x^{-1.18}$ in the figure show that any one of those curves can't be fitted well by a straight line. The reason leading to this somewhat unusual result is an issue to be made further research and analysis.

6 Conclusion

In the former section, we make statistics on different Tibetan language units : letter, super character, syllable and word, and their n-gram phrases. It shows that when we put all the n-gram phrases for $n = 1, 2, \ldots, 5$ together and sort all of them by frequency in descending order, then the frequency-rank curves in log-log axes can be fitted well for the unit of syllable and word respectively. But for the unit of super character, we didn't find a curve which can be fitted well enough by a straight line when we combine all the n-grams for $n <= m$ even if m is up to 10.

Acknowledgements

We thank the reviewers for their critical and constructive comments and suggestions that helped us improve the quality of the paper. The research is partially supported by National Science and Technology Major Project (No.2012ZX01039-004), National Science Foundation (No.61202219, No.61202220, No.61303165), Major Science and Technology Projects in Press and Publishing (No.0610-1041BJNF2328/23, No.0610-1041BJNF2328/26), and Informationization Project of the Chinese Academy of Sciences (No.XXH12504-1-10).

References

Lada A. Adamic and Bernardo A. Huberman. 2002. Zipfs law and the internet. *Glottometrics*, 3(1):143–150.

Jinyong Ai, Hongzhi Yu, and Yonghong Li. 2009. Statistical analysis on tibetan shaped structure. *Journal of Computer Applications*, 29(7):2029–2031.

MG Boroda and AA Polikarpov. 1988. The zipf-mandelbrot law and units of different text levels. *Musikometrika*, 1:127–158.

Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. 1998. On the implications of zipfs law for web caching. Technical report, Citeseer.

CA Chatzidimitriou-Dreismann, RMF Streffer, and Dan Larhammar. 1996. Lack of biological significance in the linguistic features of noncoding dnaa quantitative analysis. *Nucleic acids research*, 24(9):1676–1681.

Yuzhong Chen, Baoli Li, and Shiwen Yu. 2003a. The design and implementation of a tibetan word segmentation system. *Journal of Chinese Information Processing*, 17(3):15–20.

Yuzhong Chen, Baoli Li, Shiwen Yu, and Lancuoji. 2003b. An automatic tibetan segmentation scheme based on case auxiliary words and continuous features. *Applied Linguistics*, 2003(01):75–82.

Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. 2009. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703.

The Unicode Consortium. 2013. *The Unicode Standard, Version 6.3.0*. The Unicode Consortium, ISBN 978-1-936213-08-5, Mountain View, CA.

Leo Egghe. 1999. On the law of zipf-mandelbrot for multi-world phrases.

Leo Egghe. 2000. The distribution of n-grams. *Scientometrics*, 47(2):237–252.

Xavier Gabaix. 1999a. Zipf's law and the growth of cities. *The American Economic Review*, 89(2):129–132.

Xavier Gabaix. 1999b. Zipf's law for cities: an explanation. *The Quarterly Journal of Economics*, 114(3):739–767.

Dingguo Gao and Yuchang Gong. 2005. A statistically study on the qualities of all modern tibetan character set. *Journal of Chinese Information Processing*, 19(1):71–75.

Le Quan Ha, Elvira I Sicilia-Garcia, Ji Ming, and F Jack Smith. 2002. Extension of zipf's law to words and phrases. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–6. Association for Computational Linguistics.

Yannis M Ioannides and Henry G Overman. 2003. Zipfs law for cities: an empirical examination. *Regional science and urban economics*, 33(2):127–137.

Di Jiang and Yinghong Dong. 1994. Statistical analysis on linear processing of tibetan clustered structures. *Chinese Information Processing*, (4):44–46.

Di Jiang and Yinghong Dong. 1995. Research on property of tibetan characters as information processing. *Journal of Chinese Information Processing*, 9(2):37–44.

Di Jiang and Jiangping Kong. 2006. *Advances on the Minority Language Processing of China*. Social Sciences Academic Press, Beijing, China.

Di Jiang and Congjun Long. 2010. *On Characters of Tibetan Writing System: Alpabetic Characters, Pronunciations, ISO Codes, Frequencies, Sorting Orders, Picture Symbols and Transliterations*. Social Sciences Academic Press, Beijing, China.

Di Jiang. 1998. An entropy value of classical tibetan language and some other questions. In *Proceedings of International Conference on Chinese Information Processing*, pages 377–381. Chinese Information Processing Society of China.

Di Jiang. 2006. History and development of tibetan text information processing. In *Frontiers of Chinese Information Processing - Proceedings of the 25th Anniversary Conference of Chinese Information Processing Society of China*, pages 83–97. Tsinghua University Press, Beijing, China.

Wentian Li. 1992. Random texts exhibit zipf's-law-like word frequency distribution. *IEEE Transactions on Information Theory*, 38(6):1842–1845.

Huidan Liu, Minghua Nuo, Longlong Ma, and et al. 2011. Tibetan Word Segmentation as Syllable Tagging Using Conditional Random Fields. In *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation (PACLIC 2011)*, pages 168–177.

Huidan Liu, Minghua Nuo, Longlong Ma, and et al. 2012a. SegT: A pragmatic tibetan word segmentation system. *Journal of Chinese Information Processing*, 26(1):97–103.

Huidan Liu, Minghua Nuo, Jian Wu, and Yeping He. 2012b. Building large scale text corpus for tibetan by extracting text from web pages. In *Proceedings of the 10th asian language resources at COLING 2012*, pages 8–17.

Yajun Lu and Xiaodong Shi. 2011. Random texts exhibit zipf's-law-like word frequency distribution. *Journal of Chinese Information Processing*, 25(4):54–56.

Yajun Lu, Shaoping Ma, Min Zhang, and Guang Luo. 2003. Researches of calculations of tibetan characters, pieces, syllables, vocabulary and universal frequency and its applications. *Journal of Northwest Minorities University(Natural Science)*, 24(48):32–42.

R. N. Mantegna, S. V. Buldyrev, A. L. Goldberger, S. Havlin, C. K. Peng, M. Simons, and H. E. Stanley. 1994. Linguistic features of noncoding dna sequences. *Phys. Rev. Lett.*, 73:3169–3172, Dec.

AIDA Masaki and Noriyuki Takahashi. 1998. A proposal of dual zipfian model for describing http access trends and its application to address cache design. *IEICE transactions on communications*, 81(7):1475–1485.

Marcelo A Montemurro and Damian H Zanette. 2002. New perspectives on zipfs law in linguistics: from single texts to large corpora. *Glottometrics*, 4:87–99.

S Naranan and VK Balasubrahmanyan. 1998. Models for power law relations in linguistics and information science. *Journal of Quantitative Linguistics*, 5(1-2):35–61.

Kazumi Okuyama, Misako Takayasu, and Hideki Takayasu. 1999. Zipf's law in income distribution of companies. *Physica A: Statistical Mechanics and its Applications*, 269(1):125–131.

Ronald Rousseau. 2002. George kingsley zipf: life, ideas, his law and informetrics. *Glottometrics*, 3:11–18.

Herbert A Simon. 1955. On a class of skew distribution functions. *Biometrika*, 42(3/4):425–440.

Kwok Tong Soo. 2005. Zipf's law for cities: a cross-country investigation. *Regional science and urban Economics*, 35(3):239–263.

Kwok Tong Soo. 2007. Zipf's law and urban growth in malaysia. *Urban Studies*, 44(1):1–14.

Andreas Stolcke et al. 2002. Srilm-an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*, volume 2, pages 901–904. Denver.

Yuan Sun, Luosangqiangba, Rui Yang, and Xiaobing Zhao. 2009. Design of a tibetan automatic segmentation scheme. In *the 12th Symposium on Chinese Minority Information Processing*.

Yuan Sun, Xiaodong Yan, , Xiaobing Zhao, and Guosheng Yang. 2010. A resolution of overlapping ambiguity in tibetan word segmentation. In *Proceedings of the 3rd International Conference on Computer Science and Information Technology*, pages 222–225.

Weilan Wang and Wanjun Chen. 2004. The frequency and information entropy of tibetan character and syllabie. *Terminology Standardization and Information Technology*, (2):27–31.

Weilan Wang. 2004. The frequency-rank of language unit in modern tibetan. *Science Technology and Engineering*, 4(5):413–417.

Damián Zanette and Marcelo Montemurro. 2005. Dynamics of text generation with realistic zipf's distribution. *Journal of quantitative Linguistics*, 12(1):29–40.

Damiáan H Zanette. 2006. Zipf's law and the creation of musical context. *Musicae Scientiae*, 10(1):3–18.

George Kingsley Zipf. 1935. The psycho-biology of language.

George Kingsley Zipf. 1949. Human behavior and the principle of least effort.

Simple or Complex? Assessing the readability of Basque Texts

Itziar Gonzalez-Dios, María Jesús Aranzabe, Arantza Díaz de Ilarraza, Haritz Salaberri
IXA NLP Group
University of the Basque Country (UPV/EHU)
itziar.gonzalezd@ehu.es

Abstract

In this paper we present a readability assessment system for Basque, *ErreXail*, which is going to be the preprocessing module of a Text Simplification system. To that end we compile two corpora, one of simple texts and another one of complex texts. To analyse those texts, we implement global, lexical, morphological, morpho-syntactic, syntactic and pragmatic features based on other languages and specially considered for Basque. We combine these feature types and we train our classifiers. After testing the classifiers, we detect the features that perform best and the most predictive ones.

1 Introduction

Readability assessment is a research line that aims to grade the difficulty or the ease of the texts. It has been a remarkable question in the educational domain during the last century and is of great importance in Natural Language Processing (NLP) during the last decade. Classical readability formulae like Flesh formula (Flesch, 1948), Dale-Chall formula (Chall and Dale, 1995) and The Gunning FOG index (Gunning, 1968) take into account raw and lexical features and frequency counts. NLP techniques, on the other hand, make possible the consideration of more complex features.

Recent research in NLP (Si and Callan, 2001; Petersen and Ostendorf, 2009; Feng, 2009) has demonstrated that classical readability formulae are unreliable. Moreover, those metrics are language specific.

Readability assessment is also used as a preprocess or evaluation in Text Simplification (TS) systems e.g. for English (Feng et al., 2010), Portuguese (Aluísio et al., 2010), Italian (Dell'Orletta et al., 2011), German (Hancke et al., 2012) and Spanish (Štajner and Saggion, 2013). Given a text the aim of these systems is to decide whether a text is complex or not. So, in case of being difficult, the given text should be simplified.

As far as we know no specific metric has been used to calculate the complexity of Basque texts. The only exception we find is a system for the auto-evaluation of essays *Idazlanen Autoebaluaziorako Sistema* (IAS) (Aldabe et al., 2012) which includes metrics similar to those used in readability assessment. IAS analyses Basque texts after several criteria focused on educational correction such as the clause number in a sentence, types of sentences, word types and lemma number among others. It was foreseen to use this tool in the Basque TS system (Aranzabe et al., 2012). The present work means to add to IAS the capacity of evaluating the complexity of texts by means of new linguistic features and criteria.

In this paper we present *ErreXail*, a readability assessment system for Basque, a Pre-Indo-European agglutinative head-final pro-drop language, which displays a rich inflectional morphology and whose orthography is phonemic. *ErreXail* classifies the texts and decides if they should be simplified or not. This work has two objectives: to build a classifier which will be the preprocess of the TS system and to know which are the most predictive features that differ in complex and simple texts. The study of the most predictive features will help in the linguistic analysis of the complex structures of Basque as well.

This paper is organised as follows: In section 2 we offer an overview about this topic. We present the corpora we gathered and its processing in section 3. In section 4 we summarise the linguistic features we

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 334–344, Dublin, Ireland, August 23-29 2014.

implemented and we present the experiments and their results in section 5. The present system, *ErreXail*, is described in section 6 and in section 7 we compare our work with other studies. Finally, we conclude and outline the future work (section 8).

2 Related work

In the last years new methods have been proposed to assess the readability in NLP. For English, Si and Callan (2001) use statistical models, exactly unigram language models, combined with traditional readability features like sentence length and number of syllables per word. Coh-Metrix (Graesser et al., 2004) is a tool that analyses multiple characteristics and levels of language-discourse such us narrativity, word concreteness or noun overlap. In the 3.0 version[1] 108 indices are available. Pitler and Nenkova (2008) use lexical, syntactic, and discourse features emphasising the importance of discourse features as well. Schwarm and Ostendorf (2005) combine features from statistical language models, parse features, and other traditional features using support vector machines.

It is very interesting to take a look at readability systems for other languages as well. Some readability metrics take them into account special characteristics linked to languages. For example, in Chinese the number of strokes is considered (Pang, 2006), in Japanese the different characters (Sato et al., 2008), in German the word formation (vor der Brück et al., 2008), in French the *passé simple* (François and Fairon, 2012) and the orthographic neighbourhood (Gala et al., 2013) and in Swedish vocabulary resources (Sjöholm, 2012; Falkenjack et al., 2013) among many other features. For Portuguese, Coh-metrix has been adapted (Scarton and Aluísio, 2010) and in Arabic language-specific formulae have been used (Al-Ajlan et al., 2008; Daud et al., 2013). Looking at free word order, head final and rich morphology languages, Sinha et al. (2012) propose two new measures for Hindi and for Bangla based on English formulae. Other systems use only machine learning techniques, e.g. for Chinese (Chen et al., 2011).

The systems whose motivation is Text Simplification analyse linguistic features of the text and then they use machine learning techniques to build the classifiers. These systems have been created for English (Feng et al., 2010), Portuguese (Aluísio et al., 2010), Italian (Dell'Orletta et al., 2011) and German (Hancke et al., 2012). We follow the similar methodology for Basque since we share the same aim.

Readability assessment can be focused on different domains such as legal, medical, education and so on. Interesting points about readability are presented in DuBay (2004) and an analysis of the methods and a review of the systems is presented in Benjamin (2012) and Zamanian and Heydari (2012).

3 Corpora

Being our aim to build a model to distinguish simple and complex texts and to know which are the most predictive features based on NLP techniques, we needed to collect the corpora. We gathered texts from the web and compiled two corpora. The first corpus, henceforth *T-comp*, is composed by 200 texts (100 articles and 100 analysis) from the *Elhuyar aldizkaria*[2], a monthly journal about science and technology in Basque. *T-comp* is meant to be the complex corpus. The second corpus, henceforth *T-simp*, is composed by 200 texts from *ZerNola*[3], a website to popularise science among children up to 12 years and the texts we collected are articles. To find texts specially written for children was really challenging. Main statistics about both corpora are presented in Table 1.

Corpus	Docs.	Sentences	Tokens	Verbs	Nouns
T-comp	200	8593	161161	52229	59510
T-simp	200	2363	39565	12203	13447

Table 1: Corpora statistics

Both corpora were analysed at various levels:

1. Morpho-syntactic analysis by *Morpheus* (Alegria et al., 2002)

[1] `http://cohmetrix.memphis.edu/cohmetrixpr/cohmetrix3.html` (accessed January, 2014)
[2] `http://aldizkaria.elhuyar.org/` (accessed January, 2014)
[3] `http://www.zernola.net/` (accessed January, 2014)

2. Lemmatisation and syntactic function identification by *Eustagger* (Aduriz et al., 2003)

3. Multi-words item identification (Alegria et al., 2004a)

4. Named entities recognition and classification by *Eihera* (Alegria et al., 2004b)

5. Shallow parsing by *Ixati* (Aduriz et al., 2004)

6. Sentence and clause boundaries determination by *MuGak* (Aranzabe et al., 2013)

7. Apposition identification (Gonzalez-Dios et al., 2013)

This preprocess is necessary to perform the analysis of the features presented in section 4.

4 Linguistic features

In this section we summarise the linguistic features implemented to analyse the complexity of the texts. We distinguish different groups of features: global, lexical, morphological, morpho-syntactic, syntactic and pragmatic features. There are in total 94 features. Most of the features we present have already been included in systems for other languages but others have been specially considered for Basque.

4.1 Global features

Global features take into account the document as whole and serve to give an overview of the texts. They are presented in Table 2.

Averages
Average of words per sentence
Average of clauses per sentence
Average of letters per word

Table 2: Global features

These features are based on classical readability formulae and in the criteria taken on the simplification study (Gonzalez-Dios, 2011), namely the sentence length and the clause number per sentence. They are also included in IAS (Aldabe et al., 2012).

4.2 Lexical features

Lexical features are based on lemmas. We calculate the ratios of all the POS tags and different kinds of abbreviations and symbols. We concentrate on particular types of substantives and verbs as well. Part of theses ratios are shown in Table 3. In total there are 39 ratios in this group.

Ratios
Unique lemmas / all the lemmas
Each POS / all the words
Proper Nouns / all the nouns
Named entities / all the nouns
Verbal nouns / all the verbs
Modal verbs / all the verbs
Causative verbs / all the verbs
Intransitive verbs with one arg. (*Nor* verbs) / all the verbs
Intransitive verbs with two arg. (*Nor-Nori* verbs) / all the verbs
Transitive verbs with two arg. (*Nor-Nork* verbs) / all the verbs
Transitive verbs with three arg. (*Nor-Nori-Nork*) verbs / all the verbs
Acronyms / all the words
Abbreviations / all the words
Symbols / all the words

Table 3: Lexical features

Among those features, we want to point out the causative verbs and the intransitive or transitive verbs with one, two or three arguments (arg.) as features related to Basque. Causative verbs are verbs with the

suffix *-arazi* and they are usually translated as "to make someone + *verb*", e.g. *edanarazi*, that stands for "to make someone drink". Other factitive verbs are translated without using that paraphrase like *jakinarazi* that means "to notify", lit. "to make know". The transitivity classification is due to the fact that Basque verb agrees with three grammatical cases (ergative *Nork*, absolutive *Nor* and dative *Nori*) and therefore verbs are grouped according to the arguments they take in Basque grammars.

4.3 Morphological features

Morphological features analyse the different ways lemmas can be realised. These features are summarised in Table 4 and there are 24 ratios in total.

Ratios
Each case ending / all the case endings
Each verb aspect / all the verbs
Each verb tense / all the verbs
Each verb mood / all the verbs
Words with ellipsis / all the words
Each type of words with ellipsis / all the words with ellipsis

Table 4: Morphological features

Basque has 18 case endings (absolutive, ergative, inessive, allative, genitive...), that is, 18 different endings can be attached to the end of the noun phrases. For example, if we attach the inessive *-n* to the noun phrase *etxea* "the house", we get *etxean* "at home". The verb features considered the forms obtained with the inflection.

Verb morphology is very rich in Basque as well. The aspect is attached to the part of the verb which contains the lexical information. There are 4 aspects: puntual (aoristic), perfective, imperfective and future aspect. Verb tenses are usually marked in the auxiliary verb and there are four tenses: present, past, irreal and archaic future[4]. The verbal moods are indicative, subjunctive, imperative and potential. The latter is used to express permissibility or possible circumstances.

Due to the typology of Basque, ellipsis[5] is a normal phenomenon and ellipsis can be even found within a word (verbs, nouns, adjective...); for instance, *dioguna* which means "what we say". This kind of ellipsis occurs e.g. in English, Spanish, French and German as well but in these languages it is realised as a sentence; but it is expressed only by a word in Basque.

4.4 Morpho-syntactic features

Morpho-syntactic features are based on the shallow parsing (chunks[6]) and in the apposition detection (appositions). These features are presented in Table 5.

Ratios
Noun phrases (chunks) / all the phrases
Noun phrases (chunks) / all the sentences
Verb phrases / all the phrases
Appositions / all the phrases
Appositions / all the noun phrases (chunks)

Table 5: Morpho-syntactic features

Contrary to the features so far presented, the morpho-syntactic features take into account mainly more than a word. About apposition, there are 2 types in Basque (Gonzalez-Dios et al., 2013) but we consider all the instances together in this work.

[4]The archaic future we also take into account is not used anymore, but it can be found in old texts. Nowadays, the aspect is used to express actions in the future.

[5]Basque is a pro-drop language and it is very normal to omit the subject, the object and the indirect object because they are marked in the verb. We do not treat this kind of ellipsis in the present work.

[6]Chunks are a continuum of elements with a head and syntactic sense that do not overlap (Abney, 1991).

4.5 Syntactic features

Syntactic features consider average of the subordinate clauses and types of subordinate clauses. They are outlined in Table 6 and there are 10 ratios in total. The types of adverbial clauses are temporal, causal, conditional, modal, concessive, consecutive and modal-temporal. The latter is a clause type which expresses manner and simultaneity of the action in reference to the main clause.

Ratios
Subordinate clauses / all the clauses
Relative clauses / subordinate clauses
Completive clauses / subordinate clauses
Adverbial clauses / subordinate clauses
Each type of adverbial clause / subordinate clauses

Table 6: Syntactic features

In this first approach we decided not to use dependency based features like dependency depth or distance from dependent to head because dependency parsing is time consuming and slows down the preprocessing. Moreover, the importance of syntax is under discussion: Petersen and Ostendorf (2009) find that syntax does not have too much influence while Sjöholm (2012) shows that dependencies are not necessary. Pitler and Nenkova (2008) pointed out the importance of syntax. but Dell'Orletta et al. (2011) demonstrate that for document classification reliable results can be found without syntax. Anyway, syntax is necessary for sentence classification.

4.6 Pragmatic features

In our cases, the pragmatic features we examine are the cohesive devices. These features are summed up in Table 7. There are 12 ratios in total.

Ratios
Each type of conjunction / all the conjunctions
Each type of sentence connector / all the sentence connectors

Table 7: Pragmatic features

Conjunction types are additive, adversative and disjuntive. Sentence connector types are additive, adversative, disjuntive, clarificative, causal, consecutive, concessive and modal.

5 Experiments

We performed two experiments, the first one to build a classifier and the second one to know which are the most predictive features. For both tasks we used the WEKA tool (Hall et al., 2009).

In the first experiment we ran 5 classifiers and evaluated their performance. Those classifiers were Random Forest (Breiman, 2001), the J48 decision tree (Quinlan, 1993), K-Nearest Neighbour, IBk (Aha et al., 1991), Naïve Bayes (John and Langley, 1995) and Support Vector Machine with SMO algorithm (Platt, 1998). We used 10 fold cross-validation, similar to what has been done in other studies.

Taking into account all the features presented in section 4, the best results were obtained using SMO. This way, 89.50 % of the instances were correctly classified. The F-measure for complex text was 0.899 %, for simple texts was 0.891 % and the MAE was 0.105 %. The results using all the features are shown in Table 8.

Random Forest	**J48**	**IBk**	**Naïve Bayes**	**SMO**
88.50	84.75	72.00	84.50	**89.50**

Table 8: Classification results using all the features

We classified each feature type on their own as well and the best results were obtained using only lexical features, 90.75 %. The classification results according to their feature group are presented in Table 9. We only present the classifiers with the best results and these are remarked in bold.

Classifier	Random Forest	J48	SMO
Global	74.25	73.50	**74.75**
Lex.	88.00	85.00	**90.75**
Morph.	**82.00**	71.75	75.00
Morpho-synt.	**78.25**	76.25	72.75
Synt.	71.25	**73.75**	67.75
Prag.	67.50	**70.50**	65.75

Table 9: Classification results of each feature type

We also made different combinations of feature types and the accuracy was improved. The best combination group was the one formed by lexical, morphological, morpho-syntactic and syntactic features and they obtain 93.50 % with SMO. Best results are show in Table 10.

Feature Group	Random Forest	SMO
Global+Lex	87.50	**89.50**
Global+Lex+Morph	87.75	**89.00**
Global+Lex+Morph+Morf-sint	89.25	**89.50**
Global+Lex+Morph+Morph-sint+Sintax	87.25	**90.25**
Morph+Morph-sint	**84.25**	82.25
Morph+Morph-sint+Sintax	**83.25**	80.75
Morph+Morof-sint+Sintax+Prag	**83.75**	82.00
Lex+Morph	88.75	**92.75**
Lex+Morph+Morph-sint	**89.25**	**89.25**
Lex+Morph+Morph-sint+Sintax	89.75	**93.50**
Lex+Morph+Morph-sint+Sintax+Prag	88.50	**90.25**
Sintax+Prag	**78.25**	73.50

Table 10: Classification results using different feature combinations

Combining the feature types, SMO is the best classifier in most of the cases but Random Forest outperforms the results when there are no lexical features.

In the second experiment, we analysed which were the most predictive linguistic features in each group. We used Weka's Information Gain (InfoGain AttributeEval) to create the ranking and we ran it for each feature group. In Table 11 we present the 10 most predictive features taking all the features groups into account.

The results of this experiment are interesting for the linguistic studies on Text Simplification. It shows us indeed which phenomena we should work on next. In these experiment we notice as well the relevance of the lexical features and that syntactic features are not so decisive in document classification.

The features with relevance 0 have been analysed as well. Some of them are e.g. the ratio of the inessive among all the case endings, the ratio of the indicative mood among all the verbal moods, the ratio of the adjectives among all the words and the ratio of the ratio of the present tense among all the verbal tenses.

We also performed a classification experiment with the top 10 features and J48 is the best classifier (its best performance as well). These results are presented in Table 12.

To sum up, our best results are obtained using a combination of features (Lex+Morph+Morph-sint+Sintax). We want to remark the importance of lexical features as well, since they alone outperform all the features and 5 of them are among the top ten features.

6 System overview

The readability system for Basque *ErreXail* has a three-stage architecture (Figure 1).

So, given a Basque written text, we follow next steps:

1. The linguistic analysis will be carried out, that is, morpho-syntactic tagging, lemmatisation, syntactic function identification, named entity recognition, shallow parsing, sentence and clause boundaries determination and apposition identification will be performed. We will use the tools presented in section 3.

Feature and group	Relevance
Proper nouns / common nouns ratio (Lex.)	0.2744
Appositions / noun phrases ratio (Morpho-synt.)	0.2529
Appositions / all phrases ratio (Morpho-synt.)	0.2529
Named entities / common nouns ratio (Lex.)	0.2436
Unique lemmas / all the lemmas ratio (Lex.)	0.2394
Acronyms / all the words ratio (Lex.)	0.2376
Causative verbs / all the verbs ratio (Lex.)	0.2099
Modal-temporal clauses / subordinate clauses ratio (Synt.)	0.2056
Destinative case endings / all the case endings ratio (Morph.)	0.1968
Connectors of clarification / all the connectors ratio (Prag.)	0.1957

Table 11: Most predictive features

Random Forest	J48	IBk	Naïve Bayes	SMO
87.75	**88.25**	72.00	83.25	87.00

Table 12: Classification results using the top 10 features

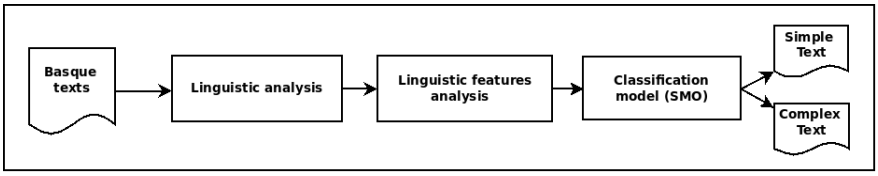


Figure 1: The architecture of system

2. Texts will be analysed according to the features and measures presented in section 4.

3. We will use the SMO Support Vector Machine as classification model, since that was the best classifier in the experiments exposed in section 5. To speed up the process for Text Simplification, we will analyse only the combination of lexical, morphological, morpho-syntactic and syntactic (Lex+Morph+Morph-sint+Sintax) features.

Although the first application of this system will be the preprocessing of texts for the Basque TS system, the system we present in this paper is independent and can be used for any other application. We want to remark that this study, as it is based on other languages, could be applied to any other language as well provided that the text could be analysed similar to us.

7 Discussion

The task of text classification has been carried out by several studies before. Due to our small corpus we were only able to discriminate between complex and simple texts like Dell'Orletta et al. (2011) and Hancke et al. (2012), other studies have classified more complexity levels (Schwarm and Ostendorf, 2005; Aluísio et al., 2010; François and Fairon, 2012). In this section we are going to compare our system with other systems that share our same goal, namely to know which texts should be simplified.

Comparing our experiment with studies that classify two grades and use SMO, Hancke et al. (2012) obtain an accuracy of 89.7 % with a 10 fold cross-validation. These results are very close to ours, although their data compiles 4603 documents and ours 400. According to the feature type, their best type is the morphological, obtaining 85.4 % of accuracy. Combining lexical, language model and morphological features they obtain 89.4 % of accuracy. To analyse their 10 most predictive features, they use Information Gain as well but we do not share any feature in common.

Dell'Orletta et al. (2011) perform three different experiments but only their first experiment is similar to our work. For that classification experiment they use 638 documents and follow a 5 fold cross-validation process of the Euclidian distance between vectors. Taking into account all the features the accuracy of their system is 97.02 %. However, their best performance is 98.12 % when they only use the combination of raw, lexical and morpho-syntactic features.

Aluísio et al. (2010) assess the readability of the texts according to three levels: rudimentary, basic and advanced. In total they compile 592 texts. Using SMO, 10 fold cross-validation and standard classification, they obtain 0.276 MAE taking into account all the features. The F-measure for original texts is 0.913, for natural simplification 0.483 and for strong simplification 0.732. They experiment with feature types as well but they obtain their best results using all the features. Among their highly correlated features they present the incidence of apposition in second place as we do here. We do not have any other feature in common.

Among other readability assessment whose motivation is TS, Feng et al. (2010) use LIBSVM (Chang and Lin, 2001) and Logistic Regression from WEKA and 10 fold cross-validation. They assess the readability of grade texts and obtain as best results 59.63 % with LIBSVM and 57.59 % with Logistic Regression. Since they assess different grades and use other classifiers it is impossible to compare with our results but we find that we share predictive features. They found out that named entity density and and nouns have predictive power as well.

8 Conclusion and perspectives

In this paper we have presented the first readability assessment system for the Basque language. We have implemented 94 ratios based on linguistic features similar to those used in other languages and specially defined for Basque and we have built a classifier which is able to discriminate between difficult and easy texts. We have also determined which are the most predictive features. From our experiments we conclude that using only lexical features or a combination of features types we obtain better results than using all the features. Moreover, we deduce that we do not need to use time consuming resources like dependency parsing or big corpora to obtain good results.

For the future, we could implement new features like word formation or word ordering both based in other languages and in neurolinguistic studies that are being carried out for Basque. Other machine learning techniques can be used, e.g. language models and in the case of getting a bigger corpora or a graded one, we could even try to differentiate more reading levels. We also envisage readability assessment at sentence level in near future.

Acknowledgements

Itziar Gonzalez-Dios's work is funded by a PhD grant from the Basque Government. We thank Lorea Arakistain and Iñaki San Vicente from *Elhuyar Fundazioa* for providing the corpora. We also want to thank Olatz Arregi for her comments. This research was supported by the the Basque Government (IT344-10), and the Spanish Ministry of Science and Innovation, Hibrido Sint project (MICINN, TIN2010-202181).

References

Steven P. Abney. 1991. Parsing by Chunks. In Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, *Principle-Based Parsing: Computation and Psycholinguistics*. Kluwer Academic.

Itziar Aduriz, Izaskun Aldezabal, Iñaki Alegria, Jose Mari Arriola, Arantza Díaz de Ilarraza, Nerea Ezeiza, and Koldo Gojenola. 2003. Finite State Applications for Basque. In *EACL'2003 Workshop on Finite-State Methods in Natural Language Processing.*, pages 3–11.

Itziar Aduriz, María Jesús Aranzabe, Jose Mari Arriola, Arantza Díaz de Ilarraza, Koldo Gojenola, Maite Oronoz, and Larraitz Uria. 2004. A cascaded syntactic analyser for Basque. *Computational Linguistics and Intelligent Text Processing*, pages 124–134.

David W. Aha, Dennis Kibler, and Marc C. Albert. 1991. Instance-based learning algorithms. *Machine Learning*, 6:37–66.

Amani A Al-Ajlan, Hend S Al-Khalifa, and A Al-Salman. 2008. Towards the development of an automatic readability measurements for Arabic language. In *Digital Information Management, 2008. ICDIM 2008. Third International Conference on*, pages 506–511. IEEE.

Itziar Aldabe, Montse Maritxalar, Olatz Perez de Viaspre, and Uria Larraitz. 2012. Automatic Exercise Generation in an Essay Scoring System. In *Proceedings of the 20th International Conference on Computers in Education*, pages 671–673.

Iñaki Alegria, María Jesús Aranzabe, Aitzol Ezeiza, Nerea Ezeiza, and Ruben Urizar. 2002. Robustness and customisation in an analyser/lemmatiser for Basque. In *LREC-2002 Customizing knowledge in NLP applications workshop*, pages 1–6, Las Palmas de Gran Canaria, May.

Iñaki Alegria, Olatz Ansa, Xabier Artola, Nerea Ezeiza, Koldo Gojenola, and Ruben Urizar. 2004a. Representation and treatment of multiword expressions in Basque. In *Proceedings of the Workshop on Multiword Expressions: Integrating Processing*, pages 48–55. Association for Computational Linguistics.

Iñaki Alegria, Olatz Arregi, Irene Balza, Nerea Ezeiza, Izaskun Fernandez, and Ruben Urizar. 2004b. Design and development of a named entity recognizer for an agglutinative language. In *First International Joint Conference on NLP (IJCNLP-04). Workshop on Named Entity Recognition.*

Sandra Aluísio, Lucia Specia, Caroline Gasperin, and Carolina Scarton. 2010. Readability assessment for text simplification. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9. Association for Computational Linguistics.

María Jesús Aranzabe, Arantza Díaz de Ilarraza, and Itziar Gonzalez-Dios. 2012. First Approach to Automatic Text Simplification in Basque. In Luz Rello and Horacio Saggion, editors, *Proceedings of the Natural Language Processing for Improving Textual Accessibility (NLP4ITA) workshop (LREC 2012)*, pages 1–8.

María Jesús Aranzabe, Arantza Díaz de Ilarraza, and Itziar Gonzalez-Dios. 2013. Transforming Complex Sentences using Dependency Trees for Automatic Text Simplification in Basque. *Procesamiento de Lenguaje Natural*, 50:61–68.

Rebekah George Benjamin. 2012. Reconstructing readability: Recent developments and recommendations in the analysis of text difficulty. *Educational Psychology Review*, 24(1):63–88.

Leo Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32.

Jeanne Sternlicht Chall and Edgar Dale. 1995. *Readability Revisited: The New DaleChall Readability Formula*. Brookline Books, Cambridge, MA.

Chih-Chung Chang and Chih-Jen Lin. 2001. Libsvm - a library for support vector machines. The Weka classifier works with version 2.82 of LIBSVM.

Yaw-Huei Chen, Yi-Han Tsai, and Yu-Ta Chen. 2011. Chinese readability assessment using TF-IDF and SVM. In *Machine Learning and Cybernetics (ICMLC), 2011 International Conference on*, volume 2, pages 705–710. IEEE.

Nuraihan Mat Daud, Haslina Hassan, and Normaziah Abdul Aziz. 2013. A Corpus-Based Readability Formula for Estimate of Arabic Texts Reading Difficulty. *World Applied Sciences Journal*, 21:168–173.

Felice Dell'Orletta, Simonetta Montemagni, and Giulia Venturi. 2011. READ-IT: assessing readability of Italian texts with a view to text simplification. In *Proceedings of the Second Workshop on Speech and Language Processing for Assistive Technologies*, SLPAT '11, pages 73–83, Stroudsburg, PA, USA. Association for Computational Linguistics.

William H. DuBay. 2004. The Principles of Readability. *Impact Information*, pages 1–76.

Johan Falkenjack, Katarina Heimann Mühlenbock, and Arne Jönsson. 2013. Features indicating readability in Swedish text. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013)*, pages 27–40.

Lijun Feng, Martin Jansche, Matt Huenerfauth, and Noémie Elhadad. 2010. A comparison of features for automatic readability assessment. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 276–284. Association for Computational Linguistics.

Lijun Feng. 2009. Automatic Readability Assessment for People with Intellectual Disabilities. *SIGACCESS Access. Comput.*, (93):84–91, January.

Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Thomas François and Cédrick Fairon. 2012. An AI readability formula for French as a foreign language. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 466–477. Association for Computational Linguistics.

Núria Gala, Thomas François, and Cédrick Fairon. 2013. Towards a French lexicon with difficulty measures: NLP helpnig to bridge the gap between traditional dictionaries and specialized lexicons. In I. Kosem, J. Kallas, P. Gantar, S. Krek, M. Langemets, and M. Tuulik, editors, *Electronic lexicography in the 21st century: thinking outside the paper. Proceedings of the eLex 2013 conference, 17-19 October 2013, Tallinn, Estonia.*, pages 132–151, Ljubljana/Tallinn. Trojina, Institute for Applied Slovene Studies/Eesti Keele Instituut.

Itziar Gonzalez-Dios, María Jesús Aranzabe, Arantza Díaz de Ilarraza, and Ander Soraluze. 2013. Detecting Apposition for Text Simplification in Basque. In *Computational Linguistics and Intelligent Text Processing*, pages 513–524. Springer.

Itziar Gonzalez-Dios. 2011. Euskarazko egitura sintaktikoen azterketa testuen sinplifikazio automatikorako: Aposizioak, erlatibozko perpausak eta denborazko perpausak. Master's thesis, University of the Basque Country (UPV/EHU).

Arthur C. Graesser, Danielle S. McNamara, Max M. Louwerse, and Zhiqiang Cai. 2004. Coh-Metrix: Analysis of text on cohesion and language. *Behavior Research Methods*, 36(2):193–202.

Robert Gunning. 1968. *The technique of clear writing*. McGraw-Hill New York.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Julia Hancke, Sowmya Vajjala, and Detmar Meurers. 2012. Readability Classification for German using lexical, syntactic, and morphological features. In *COLING 2012: Technical Papers*, page 10631080.

George H. John and Pat Langley. 1995. Estimating Continuous Distributions in Bayesian Classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo. Morgan Kaufmann.

Lau Tak Pang. 2006. *Chinese Readability Analysis and its Applications on the Internet*. Ph.D. thesis, The Chinese University of Hong Kong.

Sarah E. Petersen and Mari Ostendorf. 2009. A machine learning approach to reading level assessment. *Computer Speech & Language*, 23(1):89–106.

Emily Pitler and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 186–195. Association for Computational Linguistics.

John C. Platt. 1998. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In Bernhard Schlkopf, Christopher J. C Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods-Support Vector Learning*. MIT Press.

Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.

Satoshi Sato, Suguru Matsuyoshi, and Yohsuke Kondoh. 2008. Automatic Assessment of Japanese Text Readability Based on a Textbook Corpus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Joseph Maegaard, Benteand Mariani, Jan Odijk, Stelios Piperidis, and Daniel Tapias, editors, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA).

Carolina Evaristo Scarton and Sandra Maria Aluísio. 2010. Análise da Inteligibilidade de textos via ferramentas de Processamento de Língua Natural: adaptando as métricas do Coh-Metrix para o Português. *Linguamática*, 2(1):45–61.

Sarah E Schwarm and Mari Ostendorf. 2005. Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 523–530. Association for Computational Linguistics.

Luo Si and Jamie Callan. 2001. A statistical model for scientific readability. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 574–576. ACM.

Manjira Sinha, Sakshi Sharma, Tirthankar Dasgupta, and Anupam Basu. 2012. New Readability Measures for Bangla and Hindi Texts. In *Proceedings of COLING 2012: Posters*, pages 1141–1150, Mumbai, India, December. The COLING 2012 Organizing Committee.

Johan Sjöholm. 2012. Probability as readability: A new machine learning approach to readability assessment for written Swedish. Master's thesis, Linköping.

Tim vor der Brück, Sven Hartrumpf, and Hermann Helbig. 2008. A readability checker with supervised learning using deep indicators. *Informatica*, 32(4):429–435.

Sanja Štajner and Horacio Saggion. 2013. Readability Indices for Automatic Evaluation of Text Simplification Systems: A Feasibility Study for Spanish. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 374–382, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Mostafa Zamanian and Pooneh Heydari. 2012. Readability of texts: State of the art. *Theory and Practice in Language Studies*, 2(1):43–53.

Influence of Target Reader Background and Text Features on Text Readability in Bangla: A Computational Approach

Manjira Sinha
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
West Bengal, India
manjira@cse.iitkgp.ernet.in

Tirthankar Dasgupta
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
West Bengal, India
tirtha@cse.iitkgp.ernet.in

Anupam Basu
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
West Bengal, India
anupam@cse.iitkgp.ernet.in

Abstract

In this paper, we have studied the effect of two important factors influencing text readability in Bangla: the target reader and text properties. Accordingly, at first we have built a novel Bangla readability dataset of 135 documents annotated by 50 readers from two different backgrounds. We have identified 20 different features that can affect the readability of Bangla texts; the features were divided in two groups, namely, 'classic' and 'non-classic'. Preliminary correlation analysis reveals that text features have varying influence on the text hardness stated by the two groups. We have employed support vector machine (SVM) and support vector regression (SVR) techniques to model the reading difficulties of Bangla texts. In addition to developing different models targeted towards different type of readers, separate combinations of features were tested to evaluate their comparative contributions. Our study establishes that the perception of text difficulty varies largely with the background of the reader. To the best of our knowledge, no such work on text readability has been recorded earlier in Bangla.

1 Introduction

Readability of a text generally refers to how well a reader is able to comprehend the content of a text, through reading (Dale and Chall, 1948). Readability is a complex cognitive phenomenon where, the cognitive load of a text for a reader depends on both the characteristics of a text like, lexical choice, syntactic complexity, semantic complexity, discourse level complexity and on the background of the user. Several experiments have already established that readability of texts are quite language dependent and existing readability measures in English cannot directly be used to compute readability of other languages like, Bangla and Hindi (Sinha et al., 2012). Yet, compared to the numerous readability measures in English and other European languages(Benjamin, 2012), few initiatives have been taken to compute text readability in a Eastern Indo-Aryan language like Bangla or any other Indian languages which are structurally very different from many of their Indo-European cousins such as English, which is of West-Germanic descent (Sinha et al., 2012). One important factor that affects the readability of a text is the background of the respective reader. According to Dale (Dale, 1949), "The interpretation of the expressed thought is related more to the reader's informational background and motivations than to the internal evidences of the expressional facility of the author". Reader's background is a complex derivative of one's educational and socio-economic state. As per one of the pioneering works in readability by Dale and Chall (1949), the outcome of reading depends on many characteristics of the prospective readers including "reading abilities, interests, age, sex, intellectual

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 345–354, Dublin, Ireland, August 23-29 2014.

maturity, background of information etc." However, we do not know of any such investigations for Bangla text readability that have investigate the way background of a reader affect the readability of text. Such language specific study is needed as Bangla as a language is very different from English and the inapplicability of English readability formulae for Bangla text has already been established.

Considering the above issues as our motivation, in this paper we have developed models to predict reading difficulty of a Bangla document perceived according to different target reader groups. To categorize among different reader groups, we have considered age, education and socio-economic data as indicators of comprehension ability. In addition, we have also explored the impact of different types of text features on text comprehensibility in Bangla. However, development and evaluation of such model requires availability of well-annotated resources. To the best of our knowledge, no automatically accessible data annotated according to the reading difficulty level is available for Bangla. Therefore, we have developed a digital resource pool of Bangla text documents in Unicode encoding that can be used for various NLP tasks such as feature extraction, document analysis etc. Such a dataset is essential to analyze readability of text documents based on the target reader. Next, we have visualized the text readability problem from a machine learning perspective as a classification problem using support vector machines (SVM) and an estimation problem using support vector regression (SVR). Our study is based on a wide range of textual features, from the syntactic and lexical features of a text like, its average sentence length, average word length in terms of visual units, to discourse level features like, number of jukta-akshars (consonant conjuncts) , number of different parts of speeches, named entity and lexical relations (refer to section 3). Although regression analysis has been previously used to model the text readability in Bangla, reader group specific analysis and machine learning techniques like support vectors have not been used so far. We have considered two target reader groups namely Group-1(or Adult group) with average age of 23 Yrs and Group-2 (or minor's group) with average age of 15 Yrs.

The organization of the paper is as follows: section 2 presents a brief literature survey on existing readability metrics for English and Bangla; section 3 defines the features of a text considered in this study, and empirical data collection, section 4 discusses the experiment observations, the prediction techniques and presents the results and validations for the two techniques. Finally, section 5 offers conclusion and perspective.

2 Related Works

The quantitative analysis of text readability started with L.A. Sherman in 1880 (Sherman, 1893). Till date, English and other languages have got over 200 readability metrics (DuBay, 2004; Rabin et al., 1988).The existing quantitative approaches towards predicting readability of a text can be broadly classified into three categories (Benjamin, 2012):

Classical methods: they analyze the syntactic features of a text like sentence length, paragraph length etc. The examples are Flesch Reading Ease Score (Flesch, 1948), FOG index (Gunning, 1968), Fry graph (Fry, 1968), SMOG (McLaughlin, 1969) etc. The formulae do not take into account the background of the reader and the semantic features of the text such as whether the actual contents are making sense or not. Despite their shortcomings, these simple metrics are easy to calculate and provide a rough estimation of reading difficulty of a text provided.

Cognitively motivated methods: texts are analyzed based on the cognitive features like, cohesion, organization and users' background. Proposition and inference model (Kintsch and Van Dijk, 1978), prototype theory (Rosch, 1978), latent semantic analysis (Landauer et al., 1998), Coh-metrix (Graesser et al., 2004) are some prominent members of this group. This group of models moves beyond the surface features of a text and try to measure objectively the different cognitive indicators associated with text and the reader. However, it has been observed that, many situations, some traditional indicators perform as well as the newer and more difficult versions (Crossley et al., 2007).

Statistical language modeling: This class of approaches incorporates the power machine learning methods to the field of readability. They are particularly useful in determining readability of web texts (Collins-Thompson and Callan, 2005; Collins-Thompson and Callan, 2004; Si and Callan, 2003) (Liu et al., 2004). SVM has been used to identify grammatical patterns within a text and classification based on it (Schwarm and Ostendorf, 2005; Heilman et al., 2008; Petersen and Ostendorf, 2009). Although, these methods sound promising, the problem is that they cannot act as standalone measure:

they need an amount of training data for classifiers appropriate to a particular user group and often these measures takes into account complex text features which for resource poor languages need manual effort to annotate.

In Bangla, only a couple of works have been executed on text readability. Das and Roychoudhury (Das and Roychoudhury, 2006) studied a miniature model with respect to one parametric and two parametric fits. They have used seven paragraphs from seven literary texts. They considered two structural features of a text: average sentence length and number of syllables per 100 words. They found the two-parametric fit as better performer. Sinha et al. (Sinha et al., 2012) has developed two readability formulae for Bangla texts using regression analysis. For their study sixteen texts of length, about 100 words were used. They have considered six structural or syntactic features of a text for the work. They have demonstrated that the English readability formulae such as Flesch Reading Ease Index, SMOG Index do not perform appropriately while being applied to Bangla documents. They have found the textual features like average word length, number of polysyllabic words and number of jukta-akshars in a text to be the most influential ones. Both the works mentioned have taken into account a small subset of potentially important text features; none them have considered feature such as the extent of text cohesion. Moreover, their study did not explore the influence of readers' background on text readability. In our study, we have addressed the issue of readers' background as well as the effect of features at different textual level.

3 Empirical Data Collection

As mentioned, there is no annotated data present in Bangla, which can provide a direct classification of text difficulty for Bangla readers. Therefore, we have undertaken an effort to annotate the experiment texts with the target readers of Bangla.

3.1 Participants

Our objective in this study is to investigate how readability varies with the background of the reader. Therefore, two different target reader groups have been considered to study the relationship of effect of text parameters on comprehension and user background. SEC[1] or socio-economic classification has been stated according to the standards of Market Research Society of India (MRSI). MRSI has defined 12 socio-economic strata: A1 to E3, in the decreasing order. These strata have been designed based on the education level of the chief wage earner of the family and the number of "consumer durables" (as per a predefined list including agricultural land) owned by the family. It has been seen that this way of grading reflect the social and economic position of a household in terms of fields such as education, awareness etc. As can be inferred from the chart, the participants range from classes C2 to E1 (C2, D1, D2, E1), which represents the medium to low social-economic classes.

Type	Background	Mean age (Standard deviation)
Group 1 (adult): 25 native speakers of Bangla	Education: pursuing graduation	22.8 (1.74)
	SEC: C2-E1	
Group 2 (minors): 25 native speakers of Bangla	Education: pursuing secondary or higher secondary	15 (1.24)
	SEC: C2-E2	

Table1: User Statistics

3.2 Readability corpus preparation

We have stated in the introduction about the scarcity of annotated digital resource pool in Bangla useful for automatic processing. Although there are a few works on text readability in Bangla, the data is not available in accessible formats. To address the problem, we have developed a corpus of Bangla documents. The current size of the resource is about 250 documents of length about 2000 words spanning over broad categories such as News, literature, blogs, articles etc. A number of different text

[1] http://imrbint.com/research/The-New-SEC-system-3rdMay2011.pdf

features were computed against each document. The descriptions of the features and the justification for them have been stated below.

3.3 Feature selection:

Inferring from the cognitive load theory (Paas et al., 2003), we have assumed that the cognitive load exerted by a text on a reader depends on syntactic and lexical properties of a text like, average sentence length, average word length, number of polysyllabic words and as well as discourse features like the counts of the different parts of speeches and the number of co-references one has to resolve in order to comprehend the text. The logic behind such assumptions is as follows: while processing a text a user has to parse the sentences in it and extract semantically relevant meaning from those sentences and the words. In order to process a sentence, one has to take into account the length of the sentence and types of words contained in it; in addition, to infer the meaning of a sentence, it is important to establish the connections or the nature of dependencies among the different words in a sentence. The role of a word is determined by its parts of speech and its way of use in that context; apart from it, the words can have varied complexity based on factors like their length, count of syllables. Similarly, at the discourse level, a reader not only has to comprehend each sentence or paragraph, but also has to infer the necessary co-references among them to understand the message conveyed by the text. The complexity of this task depends on the number of entities (noun, proper nouns) in the text, how one entity is connected with other, relationships like synonymy, polysemy, and hyponymy. To capture the effects of all these parameters in our readability models, we have considered text features over a broad range. The details of the features are presented in Table 2. The word features like average word length, average syllable per word, sentence features like average sentence length and discourse features like number of polysyllabic words, number of jukta-akshars (consonant conjuncts) have been calculated as stated by Sinha et al. (Sinha et al., 2012), as the features need customizations for Bangla. The calculations based on lexical chains have been followed from Galley and McKeown (Galley and McKeown, 2003).

Feature	Description
word features	
average word length	Bangla orthographic word consists of a combination of four types of graphemes[2], each of them is considered as a single visual unit. Average word length is total word length in terms of visual units divided by number of words.
average syllable per word	Total word length in terms of syllable divided by total number of words.
sentence features	
average sentence length	Total sentence length in terms of words divided by number of sentence.
$(noun phrase)	Average number of NP per sentence
$(verb phrase)	Average number of VP per sentence
$(adjective)	Average number of adjectives per sentence
$(postposition)	Average number of postpositions per sentence. Bangla grammar has postpositions, instead of prepositions present in English. Unlike English, postpositions in Bangla do not belong to separate part of speech. The postpositions require their object noun to take possessive, objective or locative case. Suffixes act as the case markers.
$(entity)	average number of named entity per sentence
$(unique entity)	Average number of unique entity per sentence
$(clauses)	Average number of clauses per sentence

[2] http://en.wikipedia.org/wiki/Bengali_alphabet#Characteristics_of_the_orthographic_word

discourse features	
Number of polysyllabic words and normalized measure for 30 sentences	Polysyllabic words are the words whose count of syllable exceeds 2.
number of jukta-akshars (consonant conjuncts)	Total number of jukta-akshars in a text of 2000 words. It is an important feature for Bangla because each of the clusters has separate orthographic and phonemic (in some cases) representation than the constituents consonants.
#(noun phrase)	Total number of NP in the document
#(verb phrase)	Total number of VP in the document
#(adjective)	Total number of adjective in the document.
#(postposition)	Total number of postpositions in the document.
#(entity)	Total number of named entity in the document
#(unique entity)	Total number of unique entity in the document
#(lexical chain)*	Total number of lexical chain in the document
average lexical chain length*	Computed over the document

Table2: Details of text features considered for the study

The features marked with * in the above table have been manually annotated against each text. The other features, though they are computed automatically, a round of manual checking was incorporated for the sake of correctness.

Expert annotations and user annotations:

Since there is no formal ranking of Bangla texts according to their reading levels, therefore, the documents were then annotated by language experts to approximate the suitable reading level for each document. However, to develop any practical readability application, feedbacks from actual users are necessary. From the resource pool mentioned in Introduction, 135 texts were chosen for the present study: two sets of distinct 45 texts were for each group: for the adult group those were the texts annotated by experts to have relatively high reading level and for the minor's group, the texts were annotated as having relatively low reading level; pairwise t-test were performed between the two type of text features to assure that their difference is significant ($p<0.05$).

The rest 45 texts are common to both the groups to account for the difference in comprehension for the same document and the assumption that may in some cases group 2 participants have comparable reading skill as of group 1: consequently, the texts annotated by experts as demanding high reading level were selected for this purpose. These were required to ensure that the experimental data spans over a broad range and is unbiased. The text details are presented in table 2 below.

Source of Texts	Number of texts		
	Gr.1	Gr.2	common
Literary corpora_classical	5	5	5
Literary corpora_contemporay	6	5	6
News corpora_general news	6	6	5
News corpora_interview	5	6	6
Blog corpora_personal	6	5	5
Blog corpora_official	5	5	5
Article corpora_ scholar	6	7	7
Article corpora_general	6	6	6

Table3: Text details

Each participant was asked 2 questions: "How easy was it for you to understand/comprehend the text?" and "How interesting was the reading to you?". Against each question, they were to answer on a 5 point scale (1=easy, 5=very hard). Inter-rater reliability was measured through Krippendorff's alpha[3]

[3] http://en.wikipedia.org/wiki/Krippendorff's_alpha

and $\alpha = 0.81$ was found. Therefore, we concluded that annotators agree more often than would have occurred by chance. We have measured the correlation between the outcomes of two questions corresponding to each of the fifty annotators; and found that in each case the correlation was greater than 0.8 ($p < 0.05$). Therefore, the questions can be considered as equivalent, and subsequently we have considered the rating for the first question as user input for our readability models. Corresponding to each text, the average of the user ratings was considered for further processing.

4 Analysis and Model Development

4.1 Correlation coefficients

We have performed partial spearman correlation between each of the features and user rating. Table 4 presents some of the examples from each type of features due to the space limitation; results corresponding to other features are also described subsequently. The following features have selected as they have been used in the existing literature for Bangla (Sinha et al., 2012). The correlations are presented separately for the distinct texts and the common texts delivered to the two groups of users. This will allow us to investigate is there any significance difference of reading feedbacks between the different target populations.

Feature	Correlation coefficient r (Significance (if p<0.05) p value)			
	Different texts		Common texts	
	Gr. 1	Gr. 2	Gr.1	Gr. 2
Word features				
average sentence length	**0.8 (0.0017)**	**0.33(0.2011)**	**0.75 (0.0013)**	**0.54 (0.08)**
average word length	0.60 (0.0142)	0.73(0.0041)	0.66 (0.0026)	0.8 (0.0032)
Sentence features				
average syllable per word	**0.66 (0.06)**	**0.64(0.0047)**	**0.60(0.07)**	**0.75(0.0043)**
Discourse features				
number of polysyllabic words	0.73 (0.0013)	0.74 (0.0008)	0.67(0.0021)	0.65(0.0006)
normalized measure for 30 sentences	0.76(0.0011)	0.66 (0.0041)	0.65 (0.0015)	0.66(0.0032)
number of jukta-akshars	**0.87 (0.0018)**	**0.39 (0.1228)**	**0.81 (0.0024)**	**0.85 (0.0043)**

Table 4: Correlation coefficients (user rating vs text features)

Some interesting observations can be made from the above table:

- Average sentence length or mean number of words per sentence have been long found to be a strong predictor of text difficulty [1]. In our case, while this holds true for the adult data, the correlation is less for the minors and it is not significant.
- Average syllable per word does not hold significant correlation for the adult data in both cases but it does for the minor's group
- Jukta-akshars or consonant conjuncts have major impact on text readability in Bangla (Sinha et al., 2012). For adult data, it can be seen that this feature has a strong and significant correlation, which not true for the user data of group 2 for separate texts. On the other hand, for the common texts this feature was found to have high significant correlation with both the reader groups. This is may be due to the nature of the common texts.
- Apart from the above two cases, the above table also presents evidence in support of the fact that the reader's perception of text difficulty in relation to text features changes with the target reader background.

The impact of the remaining features has been discussed here with respect to the two different types of text scenarios:

Distinct texts for two groups:

- In case of the readers from the first group, the user ratings have high correlation ($r > 0.65$) with $(clauses), #(verb phrases), #(unique entity), #(lexical chain) and average lexical chain length. The correlations are also significant. However, the correlations with $(noun phrase), $(verb phrase) $(postpositions), #(postpositions), #(adjective) were found to be insignificant. The correlation of user annotation with features such as $(entity), $(unique entity) were found to be low ($r < 0.45$) but significant.

- The group 2 readers were found to show high ($r > 0.65$) and significant correlation with $(verb phrases), $(unique entity), $(clauses), #(entity), #(lexical chain) and average lexical chain span. The correlations with $(postposition), #(postpositions) were not significant. Features like $(noun phrase), $(adjective) and #(adjective) were found to have low ($r < 0.45$) but significant correlations with user ratings.

Common texts for both groups:

- It has been observed that the group 2 user ratings have higher correlation with the sentence level features than the discourse level features. In particular, features such as number of $(noun phrase), $(adjective), $(unique entity) and $(clauses) have high correlation with the text difficulty ratings provided by the minor's group. Among the discourse level features #(entity) and #(unique entity)have a high correlation, but #(verb phrase), #(adjective) were found to have not significant influence.

- On the other hand, the adult data are more inclined towards discourse features such as #(noun phrase) and #(verb phrase), #(unique entity) in a document. This may be due to the ability of the older people to comprehend the text as a whole rather than inferring meaning from individual units at a time. From sentence level feature $(clause) was found to be significant and important in terms of correlation, but $(noun phrase), $(adjective) do not bear significant correlation.

- Properties like lexical chain, which require a reader to establish connections among different attributes of a concept have great significance for both group1 and group2 annotations.

- For both the user groups the influence of average $(postposition and #(postposition) were found to be little and insignificant.

From the above discussions, it is evident that the two different target reader groups show a large difference in their reading pattern and perception of text difficulty. The difference has been observed in both the cases: when they were presented with different type of texts and with same texts. Therefore, it has been established that the target reader background plays an important role in modelling text difficulty. Accordingly, in the following sections, we have developed different models of different reader groups, and in the process we have also shown that the models have different parameter values and configurations.

4.2 Computational modelling

Analyses of correlation coefficients give an estimation of trend in user ratings against text features. The next step is to develop suitable models for automatic readability prediction. To achieve the objective, we have used machine-learning methods such as support vector machine (SVM) and support vector regression (SVR) techniques. In addition, we have also presented a comparative study of performances of different text features in readability model building in this section. The features have been used in three combinations. First they were divided in two categories i) comprising of only the six features mentioned in table 4 as they represent the 'classical' features used extensively to model text readability, and ii) second category consists of the rest 14 features and the group is termed 'non-classical' , this yielded the first two combinations. The third combination consists of all the features. Therefore, we have evaluated six different types of SVM and SVR models for each group.

We have employed a binary SVM classifier here. Given a training set instance-class pairs $(\overline{x_i}, y_i)$, $i = 1 \ldots l$, where $\overline{x_i} \in R^n$ and $\overline{y} \in \{1, -1\}^l$, the general equation of a SVM is (Manning et al., 2008):

$$\frac{1}{2}\overline{w}^T\,\overline{w} + C\sum_{i}^{l}\xi_i \;\; is\; minimized,$$

$$\overline{w} = weight\; vector, C = regularization\; term \qquad \ldots (\text{equation: } 1)$$

$$y_i\left(\overline{w}^T\Phi(\overline{x_i}) + b\right) \geq 1 - \xi_i, \qquad \xi_i(slack\; variable) \geq 0 \qquad \ldots (\text{equation: } 2)$$

In this work, we have taken 90 texts against each group of users by combining the 45 reader group specific texts and 45 common texts (refer to section 3). Then for each category of reader, the texts were shuffled randomly. We have used 70 texts for training and 20 texts for evaluation of the model and performed 2-fold cross validation. The minimum, maximum and median of the rating distribution lie respectively at **(2.33)**, **(8.4)** and **(5.92)** for adult (group1) and at **(1.83)**, **(8.2)** and **(5.5)** for minor (group 2). To train and test the SVM models, we needed to spit the data in two classes (easy and hard), this has been done by assigning the ratings less than the median in to class easy (label '-1') and the rest to the class hard (label '1'), i.e., the user ratings were mapped to the label space $\overline{y}$. In case of SVR, the label space mapping was not required. The text features were mapped to the feature space $\overline{x_i}$. Although we have tested four types of kernel functions: linear, polynomial, radial basis and sigmoid on the data using LIBSVM (Chang and Lin, 2011) software, here only the results corresponding to linear and polynomial kernels have been presented as the other two kernels performed poorly. To evaluate the quality of the classifications for SVM, multiple correlation (R) and percentage of texts accurately classified (Acc) have been used. R denotes the extent to which the predictions are close to the actual classes and its square (R^2) indicates the percentage of dependent variable variation that can be explained by the model. Therefore, while percentage accuracy is an indicator to how well the model has performed to classify, R indicates the extent of explanatory power it posses. A better fit will have large R-value as well as Acc. For SVR, root mean square error (RMSE) has been reported instead of Acc; a good fit will have less RMSE. Below tables present, the SVM and SVR results for adult and minor's data for different kernels and different combination of features. The kernels were evaluated for a number of SVM parameter combinations and only the result corresponding to the most efficient one is presented.

Features	Classic features		Non-classic features		All features	
SVM parameters	C = 10; d = 2; r = 0; $\gamma = 1/6 = 0.1$; $\xi_i = 0.01$ (total support vector = 28)					
Kernel	R	Acc.	R	Acc.	R	Acc.
linear	0.75	76%	0.73	79%	0.80	87%
Polynomial	0.73	75%	0.72	75%	0.75	79.5%

Table 5: SVM for group1 readers

Features	Classic features		Non-classic features		All features	
SVM parameters	C = 1; d = 2; r = 0; $\gamma = 1/6 = 0.1$; $\xi_i = 0.001$ (total support vector = 22)					
Kernel	R	Acc.	R	Acc	R	Acc.
Linear	0.75	75%	0.72	77%	0.83	86%
Polynomial	0.71	70%	0.73	72%	0.78	76%

Table 6: SVM for group2 readers

Features	Classic		Non-classic features		All features	
Kernel	R	RMSE	R	RMSE	R	RMSE
linear	0.56	1.6	0.53	1.7	0.68	1.1
Polynomial	0.43	2.2	0.47	11.2	0.56	23.3

Table 7: SVR for group1 readers

Features	Classic		Non-classic features		All	
Kernel	R	RMSE	R	RMSE	R	RMSE
linear	0.50	1.5	0.54	1.4	0.65	1.2
Polynomial	0.47	3.1	0.45	15.5	0.51	29.7

Table 8: SVR for group2 readers

From table 5 and table 6, it can be seen that the SVM for the two target reader groups differ significantly in term of parameter attributes and their accuracy. It is also evident that incorporating only non-classic features versus classic features improves the accuracy of SVM very slightly and both types of features have similar explanatory power; combining both the classic and non -classic feature improves the accuracy and multiple correlations significantly. The SVR from table 7 and table 8 show the similar trend in terms of feature performances: classic and non-classis features have comparable RMSE and R, but there is significant gain when the two types are taken together. The regression equations for group1 and group2 readers differ in the coefficients of the feature variables; these imply that the two groups require different readability models. Moreover, the linear kernel was found to perform better than the polynomial kernel in all the cases.

5 Conclusion

In this paper, we have studied the effect of two important factors affecting text readability in Bangla: the target reader and text properties. We have found that the perception of text difficulty varies largely with the background of the reader. Accordingly, we have developed computational models to compute readability of Bangla text documents based on the target reader group. In order to achieve our goal we have first developed a novel Bangla dataset annotated in terms of text readability by users with varying age group. A preliminary analysis of the reading pattern of each target group was performed by analysing the correlation of text features with user annotations. Next, we have applied the SVM classifier to classify text documents into two different classes namely, *hard* and *easy*; the SVM for the two reader groups have different properties, implying the difference between two corresponding models. We have also compared the performance of the classifier based on the feature set they use. We observed that in contrast to applying only the classical features or the non-classic features, performance of the classifier improves if both types of features are used. This is true for both the adult as well as the minor's dataset. Overall, we have achieved an accuracy of around 86% for the minor's dataset and 87% for the adult dataset respectively. In addition to classification, support vector regression has been used to model text difficulty from an estimation perspective. The result of the SVR also establishes our previous findings. To the best of our knowledge, no such work on text readability has been recorded earlier in Indian languages, especially in Bangla. The next step of this study is to analyse the performance of the readability formula from one group (say adult) when applied to the other group (say minors) and vice versa. We will also repeat our study with more spread apart user groups spread over less diverse economic strata. In future, we are planning to develop for multi-class text readability models. The work will also be extended to model text comprehensibility for reading disabilities in Bangla.

Reference

Benjamin, R. (2012). Reconstructing readability: Recent developments and recommendations in the analysis of text difficulty. *Educational Psychology Review*, 24:1–26.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Collins-Thompson, K. and Callan, J. (2004). A language modeling approach to predicting reading difficulty. In *Proceedings of HLT/NAACL*, volume 4.

Collins-Thompson, K. and Callan, J. (2005). Predicting reading difficulty with statistical language models. *Journal of the American Society for Information Science and Technology*, 56(13):1448–1462.

Dale, E. (1949). Readability.

Dale, E. and Chall, J. (1948). A formula for predicting readability. *Educational research bulletin*, pages 11–28.

Das, S. and Roychoudhury, R. (2006). Readability modelling and comparison of one and two parametric fit: A case study in bangla*. *Journal of Quantitative Linguistics*, 13(01):17–34.

DuBay, W. (2004). The principles of readability. *Impact Information*, pages 1–76.

Flesch, R. (1948). A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Fry, E. (1968). A readability formula that saves time. *Journal of reading*, 11(7):513–578.

Galley, M. and McKeown, K. (2003). Improving word sense disambiguation in lexical chaining. In *IJCAI*, volume 3, pages 1486–1488.

Graesser, A., McNamara, D., Louwerse, M., and Cai, Z. (2004). Coh-metrix: Analysis of text on cohesion and language. *Behavior Research Methods*, 36(2):193–202.

Gunning, R. (1968). *The technique of clear writing*. McGraw-Hill NewYork, NY.

Heilman, M., Collins-Thompson, K., and Eskenazi, M. (2008). An analysis of statistical models and features for reading difficulty prediction. In *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, pages 71–79. Association for Computational Linguistics.

Kintsch, W. and Van Dijk, T. (1978). Toward a model of text comprehension and production. *Psychological review*, 85(5):363.

Landauer, T., Foltz, P., and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284.

Liu, X., Croft, W., Oh, P., and Hart, D. (2004). Automatic recognition of reading levels from user queries. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 548–549. ACM.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge.

McLaughlin, G. (1969). Smog grading: A new readability formula. *Journal of reading*, 12(8):639–646.

Paas, F., Renkl, A., and Sweller, J. (2003). Cognitive load theory and instructional design: Recent developments. *Educational psychologist*, 38(1):1–4.

Petersen, S. E. and Ostendorf, M. (2009). A machine learning approach to reading level assessment. *Computer Speech & Language*, 23(1):89–106.

Rabin, A., Zakaluk, B., and Samuels, S. (1988). Determining difficulty levels of text written in languages other than english. *Readability: Its past, present & future. Newark DE: International Reading Association*, pages 46–76.

Rosch, E. (1978). Principles of categorization. *Fuzzy grammar: a reader*, pages 91–108.

Schwarm, S. and Ostendorf, M. (2005). Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 523–530. Association for Computational Linguistics.

Sherman, L. (1893). Analytics of literature: A manual for the objective study of english poetry and prose. *Boston: Ginn*.

Si, L. and Callan, J. (2003). A semisupervised learning method to merge search engine results. *ACM Transactions on Information Systems (TOIS)*, 21(4):457–491.

Sinha, M., Sharma, S., Dasgupta, T., and Basu, A. (2012). New readability measures for Bangla and Hindi texts. In *Proceedings of COLING 2012: Posters*, pages 1141–1150, Mumbai, India. The COLING 2012 Organizing Committee.

Inducing Word Sense with Automatically Learned Hidden Concepts

Baobao Chang **Wenzhe Pei** **Miaohong Chen**
Key Laboratory of Computational Linguistics, Ministry of Education
School of Electronics Engineering and Computer Science, Peking University
Beijing, P.R.China, 100871
{chbb,peiwenzhe,miaohong-chen}@pku.edu.cn

Abstract

Word Sense Induction (WSI) aims to automatically induce meanings of a polysemous word from unlabeled corpora. In this paper, we first propose a novel Bayesian parametric model to WSI. Unlike previous work, our research introduces a layer of hidden concepts and view senses as mixtures of concepts. We believe that concepts generalize the contexts, allowing the model to measure the sense similarity at a more general level. The Zipf's law of meaning is used as a way of pre-setting the sense number for the parametric model. We further extend the parametric model to non-parametric model which not only simplifies the problem of model selection but also brings improved performance. We test our model on the benchmark datasets released by Semeval-2010 and Semeval-2007. The test results show that our model outperforms state-of-the-art systems.

1 Introduction

Word Sense Induction (WSI) aims to automatically induce meanings of a polysemous word from unlabeled corpora. It discriminates among meanings of a word by identifying clusters of similar contexts. Unlike the task of Word Sense Disambiguation (WSD), which classifies polysemous words according to a pre-existing and usually hand-crafted inventory of senses, WSI makes it attractive to researchers by eliminating dependence on a particular sense inventory and learning word meaning distinction directly based on the contexts as observed in corpora.

Almost all WSI work relies on the distributional hypothesis, which states that words occurring in similar contexts will have similar meanings. To effectively discriminate among contexts, proper representation of contexts would be a key issue. Basically, context can be represented as a vector of words co-occurring with the target word within a fixed context window. The similarity between two contexts of the target word can then be measured by the geometrical distance between the corresponding vectors. To ease the sparse problem and capture more semantic content, some kinds of generalizations or abstractions are needed. For example, a context of *bank* including *money* may not share similarity with that including *cash* measured at word level. However, given the conceptual relationship between *money* and *cash*, the two contexts actually share high similarity.

One straightforward way of introducing conceptualization is to assign semantic code to context words, where semantic codes could be derived from WordNet or other resources like thesauruses. However, two problems remain to be tackled. The first one concerns ambiguities of context words. Context words may have multiple semantic codes and thus word sense disambiguation to context words or other extra cost is needed. The second one concerns the nature of WSI task. WSI actually is target-word-specific, which means the conceptualization should be done specifically to different target words. A general purpose conceptualization defined by a thesaurus may not well meet this requirement and may not be equally successful in discriminating contexts of different target words.

To address these problems, we first propose a parametric Bayesian model which jointly finds conceptual representations of context words and the sense of the target word. We do this by introducing a layer of target-specific conceptual representation between the target sense layer and the context words layer

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 355–364, Dublin, Ireland, August 23-29 2014.

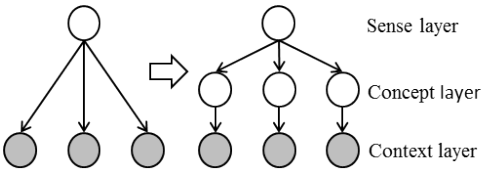


Figure 1: Architecture of our model

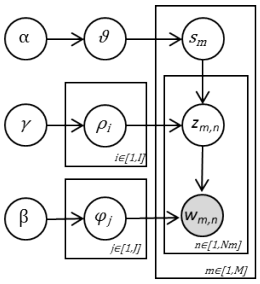


Figure 2: Graphical notation of the Basic Model

through a Bayesian framework as illustrated in Figure 1. From the generative perspective, the sense of the target word is first sampled. Then the sense generates different conceptual configurations which in turn generate different contexts. With a deeper architecture, our model makes it possible to induce word senses at a more abstract level, i.e. the concept level, which is not only less sparse but also more semantically oriented. Both the senses of the target word and the latent concepts are inferred automatically and unsupervisedly with inference procedure given enough contexts involving a target word. The latent concepts inferred with the model share similarities with those defined in thesauruses, as both of them cluster semantically related words. However, since the latent concepts are inferred with regard to individual target words, they are target-word-specific and thus fit the WSI task better than general purpose concepts defined in thesauruses. Context words may still correspond to multiple latent concepts. However, the disambiguation is implicitly done in the process of the word sense induction.

Setting the number of senses that the algorithm should arrive at is another problem frequently exercising the minds of WSI people. Instead of trying different sense numbers on a word-by-word basis, we propose to use Zipf's law of meaning (Zipf, 1945) to guide the selection of the sense numbers in this paper. With the law of meaning, sense numbers could be set on an all-word basis, rather than on a word-by-word basis. This is not only simple but also efficient, especially in the case where there are a large number of target words to be concerned.

We further extend the parametric model into a non-parametric model, as it allows adaptation of model complexity to data. By extending our model to non-parametric model, the need to preset the numbers of senses and latent concepts are totally removed and, moreover, the model performance is also improved.

We evaluate our model on the commonly used benchmark datasets released by both Semeval-2010 (Manandhar et al., 2010) and Semeval-2007 (Agirre and Soroa, 2007). The test results show that our models perform much better than the state-of-the-art systems.

2 The parametric model

2.1 Basic Model

The main point of our work is that different senses are signaled by contexts with different concept configurations, where different concepts are formally defined as different distributions over context words. Formally, we denote by $P(s)$ the global multinomial distribution over senses of an ambiguous word and by $P(w|z)$ the multinomial distributions over context words w given concept z. Context words are generated by a mixture of different concepts whose mixture proportion is defined by $P(z|s)$, such that:

$$P(w_i) = \sum_j P(s=j) \sum_k P(z_i = k|s=j) P(w_i|z_i = k)$$

Following the model, each context word w_i surrounding the target word is generated as follows: First, a sense s is sampled from $P(s)$ for the target word. Then for each context word position i, a concept z_i is sampled according to mixture proportion $P(z|s)$ and w_i is finally sampled from $P(w|z)$.

Figure 2 shows the model with the graphical notation, where M is the number of instances of contexts regarding to a concerned target word and N_m is the number of word tokens in context m. s_m is the sense label for target word in context m. $w_{m,n}$ is the n-th context word in context m. $z_{m,n}$ is the concept label associated with $w_{m,n}$. I is the total number of senses to be induced. J is the total number

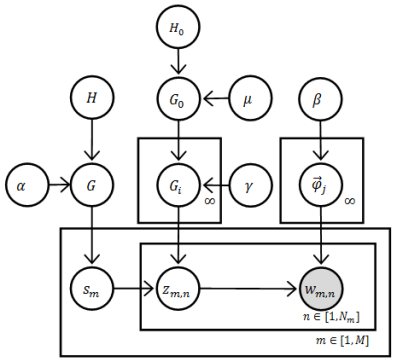


Figure 3: Graphical notation of the non-parametric WSI model

of concepts. $\vec{\theta}$ is the notational shorthand for the sense distribution $P(s)$, $\vec{\rho}_i$ is the shorthand for the i-th sense-concept distribution $P(z|s = i)$, and $\vec{\varphi}_j$ is the j-th concept-word distribution $P(w|z = j)$. Following conventional Bayesian practice, $\vec{\theta}$, $\vec{\rho}_i$ and $\vec{\varphi}_j$ are assumed to be drawn from Dirichlet priors with symmetric parameter α, γ, β respectively. The observed variable is represented with shaded node and hidden variable with unshaded node.

2.2 Zipf's law of meaning

Most of the WSI work requires that the number of senses to be induced be specified ahead of time. One straightforward way to deal with this problem is to repeatedly try different numbers of senses on a development set and select the best performed number. However, this should be done in principle on a word-by-word basis, and thus could be time-consuming and prohibitive when there are lots of target words to be concerned. A more systematic way of setting sense numbers in Bayesian models is extending the parametric model into a non-parametric model, which will be described in detail in section 3.

To work with our parametric model, we propose in this paper that an empirical law, Zipf's law of meaning (Zipf, 1945), could be used to guide the sense number selection. Zipf's law of meaning states that the number of sense of a word is proportional to its frequency as shown in the following equation:

$$I = K * f^b \tag{1}$$

where I is the number of word senses and f is the frequency of the word. K is the coefficient of proportionality which is unknown and b is about 0.404 according to an experimental study done by Edmonds (2006).

Certainly, Zipf's law of meaning is not as strict as a rigorous mathematical law. However, it sketches the distribution of the sense numbers with word frequencies of all words and allows us to estimate the sense numbers on an all-word basis by selecting appropriate coefficient K. This is not only simple but also efficient, especially in the case that there are a large number of target words to be concerned.

3 Non-parametric Model

A limitation of the parametric model is that the sense number I of the target word and the number J of latent concepts need to be fixed beforehand. Bayesian non-parametric (BNP) models offer elegant approach to the problem of model selection and adaption. Rather than comparing models that vary in complexity, the BNP approach is to fit a single model that can adapt its complexity to the data. Unlike the parametric approach, BNP approach assumes an infinite number of clusters, among which only a few are active given the training data. Our basic model can be naturally extended into a BNP model as shown in Figure 3. Instead of assuming a finite number of senses, we place a nonparametric, Dirichlet process (DP) prior on the sense distribution as follows:

$$\begin{aligned} G &\sim DP(\alpha, H) \\ s_m &\sim G, m = 1, 2, \ldots, M \end{aligned}$$

where α is the concentration parameter and H is the base measure of the Dirichlet process.

For each sense s_i of the target words, we place a Hierarchical Dirichlet process (HDP) prior on the mixture proportion to latent concepts shown as follows:

$$\begin{aligned} G_0 &\sim DP(\mu, H_0) \\ G_i &\sim DP(\gamma, G_0), i = 1, 2, \ldots \\ z_{m,n} &\sim G_i, n = 1, 2, \ldots, N_m \\ w_{m,n} &\sim \vec{\varphi}_{z_{m,n}} \end{aligned}$$

where μ and γ are concentration parameters to G_0 and G_i, H_0 is the base measure of G_0.

By using HDP priors, we make sure that the same set of concept-word distributions is shared across all senses and all contexts of a target word, since each random measure G_i inherits its set of concepts from the same G_0.

As in parametric model, $\vec{\varphi}_j$ is the j-th concept-word distribution $P(w|z = j)$, however, there are now an infinite number of such distributions. So is the number of senses. However, with a fixed number of contexts of the target word, only a finite number of senses and concepts are active and they could be inferred automatically by the inference procedure.

4 Model Inference

We use Gibbs sampling (Casella and George, 1992) for inference to both the parametric and nonparametric model. As a particular Markov Chain Monte Carlo (MCMC) method, Gibbs sampling is widely used for inference in various Bayesian models (Teh et al., 2006; Li and Li, 2013; Li and Cardie, 2014).

4.1 The Parametric Model

For the parametric model, we use collapsed Gibbs sampling, in which the sense distribution $\vec{\theta}$, sense-concept distribution $\vec{\rho}_i$ and concept-word distribution $\vec{\varphi}_j$ are integrated out. At each iteration, the sense label s_m of the target word in context m is sampled from conditional distribution $p(s_m|\vec{s}_{\neg m}, \vec{z}, \vec{w})$, and the concept label $z_{m,n}$ for the context word $w_{m,n}$ is sampled from conditional distribution $p(z_{m,n}|\vec{s}, \vec{z}_{\neg(m,n)}, \vec{w})$. Here $\vec{s}_{\neg m}$ refers to all current sense assignments other than s_m and $\vec{z}_{\neg(m,n)}$ refers to all current concept assignment other than $z_{m,n}$.

The conditional distribution $p(s_m|\vec{s}_{\neg m}, \vec{z}, \vec{w})$ and $p(z_{m,n}|\vec{s}, \vec{z}_{\neg(m,n)}, \vec{w})$ can be derived as shown in equation (2) and (3) respectively:

$$p(s_m = i|\vec{s}_{\neg m}, \vec{z}, \vec{w}; \alpha, \beta, \gamma) \propto (c_i^{\neg m} + \alpha) \cdot \frac{\prod_{j=1}^{J} \prod_{x=1}^{f_{m,j}} (c_{i,j}^{\neg m} + \gamma + x - 1)}{\prod_{x=1}^{f_{m,*}} (\sum_{j=1}^{J} c_{i,j}^{\neg m} + J * \gamma + x - 1)} \tag{2}$$

$$p(z_{m,n} = j|\vec{s}, \vec{z}_{\neg(m,n)}, \vec{w}; \alpha, \beta, \gamma) \propto (c_{s_m,j}^{\neg(m,n)} + \gamma) \cdot \frac{(c_{j,w_{m,n}}^{\neg(m,n)} + \beta)}{\sum_{t=1}^{V} c_{j,t}^{\neg(m,n)} + V * \beta} \tag{3}$$

Here, $c_i^{\neg m}$ is the number of instances with sense i. $c_{i,j}^{\neg m}$ is the number of concept j in instances with sense i. Both of them are counted without the m-th instance of the target word. $c_{s_m,j}^{\neg(m,n)}$ is defined in a similar way with $c_{i,j}^{\neg m}$ but without counting the word position (m, n). $c_{j,w_{m,n}}^{\neg(m,n)}$ is the number of times word $w_{m,n}$ is assigned to concept j without counting word position (m, n). $f_{m,j}$ is the number of concept j assigned to context words in instance m and $f_{m,*}$ is the total number of words in contexts of instance m. V stands for the size of the word dictionary, i.e. the number of different words in the data. x is an index which iterates from 1 to $f_{m,*}$.

$\vec{\theta}$, $\vec{\rho}_i$ and $\vec{\varphi}_j$ can be estimated in a similar way, we now only show as example the estimation of $\vec{\rho}_i$, parameters for sense-concept distributions. According to their definitions as multinomial distributions with Dirichlet prior, applying Bayes' rule yields:

$$p(\vec{\rho}_i|\vec{z}; \vec{\gamma}) = \frac{p(\vec{\rho}_i; \vec{\gamma}) * p(\vec{z}|\vec{\rho}_i; \vec{\gamma})}{Z_{\vec{\rho}_i}} = Dir(\vec{\rho}_i|\vec{c}_i + \vec{\gamma})$$

where $\vec{c_i}$ is the vector of concept counts for sense i. Using the expectation of the Dirichlet distribution, values of $\rho_{i,j}$ can be worked out as follows:

$$\rho_{i,j} = \frac{c_{i,j} + \gamma}{\sum_{k=1}^{J} c_{i,k} + J * \gamma}$$

Different read-outs of $\rho_{i,j}$ are then averaged to produce the final estimation.

4.2 The Non-parametric Model

Chinese restaurant process (CRP) and Chinese restaurant franchise (CRF) process (Teh et al., 2006) have been widely used as sampling scheme for DP and HDP respectively. As our non-parametric model involves both DP and HDP, we use both CRP and CRF based sampling for model inference.

In the CRP metaphor to DP, there is one Chinese restaurant with an infinite number of tables, each of which can seat an infinite number of customers. The first customer enters the restaurant and sits at the first table. The second customer enters and decides either to sit with the first customer or by herself at a new table. In general, the $n+1$st customer either joins an already occupied table k with probability proportional to the number n_k of customers already sitting there, or sits at a new table with probability proportional to α. As in our model, when we sample the sense s_m for each context, we assume that tables correspond to senses of target words and customers correspond to whole contexts in which the target word occurs.

In the CRF metaphor to HDP, there are multiple Chinese restaurants, and each one has infinitely many tables. On each table the restaurant serves one of infinitely many dishes that other restaurants may serve as well. At each table of each restaurant one dish is ordered from the menu by the first customer who sits there, and it is shared among all customers who sit at that table. The menu is shared by all the restaurants. To be specific to our model, when we sample the concept $z_{m,n}$ for each context word, we assume each sense s_m of the target word corresponds to a restaurant and each word $w_{m,n}$ corresponds to a customer while concept $z_{m,n}$ corresponds to the dishes served to the customer by the restaurant. Neither the number of restaurant nor the number of dishes is finite in our model.

For model inference, we first sample s_m using CRP-based sampling and then we sample $z_{m,n}$ for each s_m using CRF-based sampling. The sampling of s_m and $z_{m,n}$ are done alternately, but not independently. The sampling of s_m is conditional on the current value of $z_{m,n}$ and vice versa, conforming to the scheme of Gibbs Sampling.

The equation for sampling s_m is derived as in equation (4):

$$p(s_m = i|\vec{s}_{\neg m}, \vec{z}, \vec{w}) \propto \begin{cases} c_i^{\neg m} \cdot p(\vec{z}_m|\vec{z}_{\neg m}, s_m = i) & \textbf{if} \quad i = old \\ \alpha \cdot p(\vec{z}_m|\vec{z}_{\neg m}, s_m = i^{new}) & \textbf{else} \end{cases}$$

$$where$$

$$p(\vec{z}_m|\vec{z}_{\neg m}, s_m = i) = \frac{\prod_{j=1}^{J}\prod_{x=1}^{f_{m,j}}(c_{i,j}^{\neg m} + \gamma * \frac{c_{t,j}^{\neg m}}{c_{t,*}^{\neg m}+\mu} + x - 1)}{\prod_{x=1}^{f_{m,*}}(\sum_{j=1}^{J} c_{i,j}^{\neg m} + \gamma + x - 1)} \tag{4}$$

Here $p(\vec{z}_m|\vec{z}_{\neg m}, s_m = i)$ is estimated block-wise for context m according to the CRF metaphor. $c_i^{\neg m}$ and $c_{i,j}^{\neg m}$ are defined in the same way as that in equation (2). $c_{t,j}^{\neg m}$ is the number of tables with dish j in all restaurants but m and $c_{t,*}^{\neg m}$ means the number of tables in all restaurants but m. x is an index which iterates from 1 to $f_{m,*}$.

Sampling $z_{m,n}$ needs more steps than sampling s_m as we need to record the table assignment for each dish (concept). For each dish $z_{m,n}$ of a customer $w_{m,n}$, we first sample the table at which the customer sits according to the following equations:

$$p(t_{m,n} = t|\vec{t}_{\neg(m,n)}, \vec{z}_{\neg(m,n)}, w_{m,n}, s_m = i) \propto \begin{cases} c_{i,t}^{\neg(m,n)} \cdot p_j^{\neg(m,n)}(w_{m,n}) & \textbf{if} \quad t = old \\ \gamma \cdot p(w_{m,n}|\vec{t}_{\neg(m,n)}, t_{m,n} = t, \vec{z}_{\neg(m,n)}, w_{m,n}) & \textbf{else} \end{cases}$$

$$where$$

$$p_j^{\neg(m,n)}(w_{m,n}) = p(w_{m,n}|z_{m,n} = j, \vec{w}_{\neg(m,n)}) = \frac{c_{j,w_{m,n}}^{\neg(m,n)} + \beta}{\sum_{t=1}^{V} c_{j,t}^{\neg(m,n)} + V\beta}$$

	Basic Model	BNP
α	1.0	0.2
β	0.05	0.01
γ	0.05	0.2
μ	N/A	0.001
K	0.27	N/A
Concept number	20	N/A
Context window	$\pm$ 5 words	$\pm$ 9 words

Table 1: Hyperparamters of our models

Here $c_{i,t}^{\neg(m,n)}$ is the number of customers on table t in restaurant i and $c_{j,w_{m,n}}^{\neg(m,n)}$ has the same meaning as in equation (3). If the sampled table t is previously occupied, then $z_{m,n}$ is set to the dish j assigned to t according to the CRF metaphor. If the sampled table t is new, the probability $p(w_{m,n}|\vec{t}_{\neg(m,n)}, t_{m,n} = t, \vec{z}_{\neg(m,n)}, w_{m,n})$ is calculated using equation (5), which is the sum of the probability of all previously ordered dishes and the newly ordered dish.

$$p(w_{m,n}|\vec{t}_{\neg(m,n)}, t_{m,n} = t, \vec{z}_{\neg(m,n)}, w_{m,n}) = \sum_{j=1}^{J} \frac{c_{t,j}^{\neg(m,n)}}{c_{t,*}^{\neg(m,n)} + \mu} \cdot p_j^{\neg(m,n)}(w_{m,n}) + \frac{\mu}{c_{t,*}^{\neg(m,n)} + \mu} \cdot p_{j^{new}}^{\neg(m,n)} \tag{5}$$

Because a new table is added, we then sample a new dish for this table according to equation (6).

$$p(z_{m,n} = j|\vec{t}, \vec{z}_{\neg(m,n)}) \propto \begin{cases} c_{t,j}^{\neg(m,n)} \cdot p_j^{\neg(m,n)}(w_{m,n}) & \textbf{if} \quad j = old \\ \mu \cdot p_{j^{new}}^{\neg(m,n)}(w_{m,n}) & \textbf{if} \quad j = new \end{cases} \tag{6}$$

After the dish j is sampled, it is assigned to the new table and the number of table serving dish j is added.

Parameters $\vec{\theta}$, $\vec{\rho}_i$ and $\vec{\varphi}_j$ can be estimated in the same way as described in section 4.1.

5 Experiment

5.1 Experiment Setup

Data Our primary WSI evaluation is based on the standard dataset in Semeval-2010 Word sense induction & Disambiguation task (Manandhar et al., 2010). The target word dataset consists of 100 words, 50 nouns and 50 verbs. There are a total number of 879,807 sentences in training set and 8,915 sentences in testing set. The average number of word senses in the data is 3.79.

Model Selection The trail data of Semeval-2010 WSI task is used as development set for parameter tuning, which consists of training and test portions of 4 verbs. The 4 verbs are different words than the 100 target words in the training data. There are only about 138 instances on average for each target word in the training part of the trial data. To make a development set of more reasonable size, the trial data are supplemented with 6K instances of the 4 verbs extracted from the British National Corpus (BNC)[1] corpus. As we use the Zipf's law of meaning to guide the selection of number of senses, BNC was also used to count word frequencies.

The final hyper-parameters are set as in Table 1. In all the following experiments, Gibbs sampler is run for 2000 iterations with burn-in period of 500 iterations. Every 10th sample is read out for parameter estimating after the burn-in period to avoid autocorrelation. Due to the randomized property of Gibbs sampler, all results in the next sections are averaged over 5 runs. The average running time for each target word is about 7 minutes on a computer equipped with an Intel Core i5 processor working at 3.1GHz and 8GB RAM.

Pre-Processing For each instance of the target word in training data and testing data, all words are lemmatized and stop words like '*of*', '*the*', '*a*' which are irrelevant to word sense distinction are filtered. Words occurring less than twice are removed.

Evaluation method Semeval-2010 WSI task presents two evaluation schemes which are supervised evaluation and unsupervised evaluation. In supervised evaluation, the gold standard dataset is split into

[1] www.natcorp.ox.ac.uk/

Model	Supervised Evaluation		Unsupervised Evaluation		Averaged #s
	80-20 split	60-40 split	V-Measure	Paired-Fscore	
Basic Model	64.12	63.68	11.52	**44.42**	5
Basic Model + Zipf	66.4	65.25	15.2	35.12	7.66
BNP	**69.3**	**68.9**	**21.4**	23.1	15.62

Table 2: Test results with different configurations.

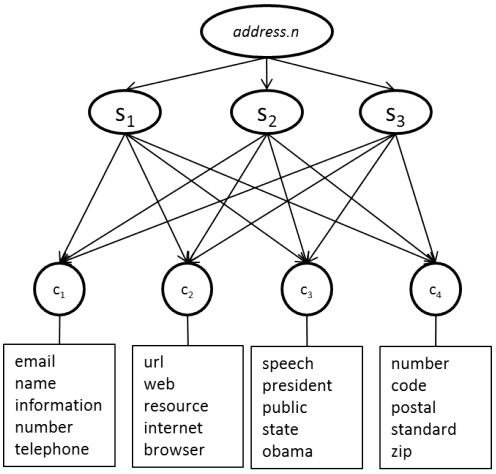


Figure 4: Examples of concepts induced with the BNP model specific to the target word *address.n* (with c_i denoting concept)

a mapping and an evaluation parts. The first part is used to map the automatically induced senses to gold standard senses. The mapping is then used to calculate the system's F-Score on the second part. According to the size of mapping data and evaluation data, the evaluation results are measured on two different splits which are 80-20 splits and 60-40 splits. 80-20 splits means that 80% of the test data are used for mapping and 20% are used for evaluation. In unsupervised evaluation, the system outputs are compared by using metrics V-Measure (Rosenberg and Hirschberg, 2007) and Paired F-Score (Artiles et al., 2009).

5.2 Experiment Results

Table 2 lists all experiment results. The Basic Model stands for the parametric model with fixed number of senses for all target words. The number of senses is set to 5 which gives the best performance on development set. Basic Model + Zipf is the model with the number of sense estimated by Zipf's law of meaning. BNP stands for our non-parametric model. As we can see, compared with the Basic Model with fixed sense number, the model using Zipf's law of meaning achieves improved performance. This means Zipf's law of meaning has positive effect in setting the sense number of the WSI task. BNP achieves the best performance on both supervised evaluation and V-measure evaluation. In terms of Paired F-score, however, the Basic Model gets the best results while BNP performs worst. This is consistent with what claimed by Manandhar et al. (2010), that Paired F-score tends to penalize the model with higher number of clusters.

As stated before, our models not only perform word sense induction but also group the context words into concepts. Figure 4 shows 4 of the concepts induced by BNP with regard to the target word *address.n*. Senses of *address.n* are defined as the mixture of concepts and concepts are defined as distributions over context words. We only list the top five words with the highest probabilities under each concept. As shown in Table 2, the non-parametric model induces much finer granularity of senses than the gold standard, it makes distinction among *email address*, *web address*, and even *ip address*. A possible solution is to further measure the closeness of senses based on the sense representations induced and merge similar senses to produce coarser granularity of senses.

Model	F-score(%)
BNP+position	**69.7**
BNP	69.3
Basic Model + Zipf	66.4
Basic Model	64.1
HDP	65.8
HDP+position (Lau et al., 2012)	68
distNB (Choe and Charniak, 2013)	65.4
UoY (Korkontzelos and Manandhar, 2010)	62.4

Model	F-score(%)
BNP+position	**88.0**
BNP	86.1
HDP (Yao and Van Durme, 2011)	85.7
HDP+position (Lau et al., 2012)	86.9
Feature-LDA (Brody and Lapata, 2009)	85.5
1-layer-LDA (Brody and Lapata, 2009)	84.6
HRG (Klapaftis and Manandhar, 2010)	87.6
I2R (Niu et al., 2007)	86.8

Table 3: Comparison with state-of-the-arts on Semeval-2010 data (left) and Semeval-2007 data (right)

5.3 Comparison with previous work

Much previous work (Brody and Lapata, 2009; Klapaftis and Manandhar, 2010; Yao and Van Durme, 2011) tested their models only on Semeval-2007 dataset (Agirre and Soroa, 2007) which consists of roughly 27K instances of 65 target verbs and 35 target nouns, coming from the Wall Street Journal corpus (WSJ) (Agirre and Soroa, 2007). For a complete comparison, we also test our model on the Semeval-2007 dataset. Since training data was not provided as part of the original Semeval-2007 dataset, we follow the approach of previous work (Brody and Lapata, 2009; Yao and Van Durme, 2011; Lau et al., 2012) to construct training data for each target word by extracting instances from the BNC corpus. Following paractices as much previous work (Brody and Lapata, 2009; Yao and Van Durme, 2011; Lau et al., 2012) did, we compare with previous work with supervised F-score on 80-20 data split in Semeval-2010 and noun data in Semeval-2007.

Table 3 (left) compares our models against the state-of-the-art systems tested on 80-20 data split in Semeval-2010. HDP+position (Lau et al., 2012) improved the HDP model (Yao and Van Durme, 2011) by including a position feature. distNB (Choe and Charniak, 2013) extends the naive Bayes model by reweighting the conditional probability of a context word given the sense by its distance to the target word. UoY (Korkontzelos and Manandhar, 2010) is the best performing system in Semeval-2010 competition which used a graph-based model. We re-implemented and tested the HDP model on the Semeval-2010 dataset since Yao and Van Durme (2011) and Lau et al. (2012) did not report their HDP results on this dataset.

Different with normal practice in WSI work, there is no feature engineering in our model. However, our BNP model outperformed all the systems on supervised evaluation. Even the Basic Model outperformed the best performing Semeval-2010 system. Especially, our BNP model performs much better than the HDP model. Both Lau et al. (2012) and Choe and Charniak (2013) show benefit of using positional information. Since our model does not exclude further feature engineering, we also introduce a position feature[2] into our non-parametric model (**BNP+position**) as in Lau et al. (2012). This contributes to a further 0.4% rise in performance.

Table 3 (right) compares our models with previous work on the nouns dataset in Semeval-2007. We divides systems being compared into two groups. The first group model the WSI task with Bayesian framework, while the second group uses models other than Bayesian model. Feature-LDA is the LDA-based model proposed by Brody and Lapata (2009) which incorporates a large number of features into the model. The 1-layer-LDA is their model with only bag-of-words features. HRG is a hierarchical random graph model. I2R is the best performing system in Semeval-2007. As shown in Table 3 (right), our BNP model with position feature (**BNP+position**) outperforms all systems. If we restrict our attention to the first group in which all models are Bayesian model, our BNP model without feature engineering outperforms the HDP model which is also non-parametric model without feature engineering.

6 Related Work

A large body of previous work is devoted to the task of Word Sense Induction. Almost all work relies on the distributional hypothesis, which states that words occurring in similar contexts will have similar meanings. Different work exploits distributional information in different forms, including context clustering models (Schütze, 1998; Niu et al., 2007; Pedersen, 2010; Elshamy et al., 2010; Kern et al., 2010), graph-based models (Korkontzelos and Manandhar, 2010; Klapaftis and Manandhar, 2010) and Bayesian

[2]Formally, the position feature is the context words with its relative position to the target word.

models. For Bayesian models, Brody and Lapata (2009) firstly introduced a Bayesian model to WSI task. They used the LDA-based model in which contexts of target word were viewed as documents as in the LDA model (Blei et al., 2003) and senses as topics. They trained a separate model for each target word and included a variety of features such as words, part-of-speech and dependency information. Yao and Van Durme (2011) extended LDA-based model into non-parametric HDP model but removed the feature engineering. Lau et al. (2012) showed improved supervised F-score by including position feature to the HDP model. Choe and Charniak (2013) proposed a reweighted naive Bayes model by incorporating the idea that words closer to the target word are more relevant in predicting the sense.

Our model differs from the context clustering models and graph-based models, as it is a Bayesian probabilistic model. Our work also differs from the LDA-based models. LDA topics were actually re-interpreted as senses of target word as Brody and Lapata (2009) applied the LDA to WSI tasks, so did Yao and Van Durme (2011) and Lau et al. (2012). They induced word senses by firstly tagging (sampling) senses (of target words) to context words and selecting the mostly tagged sense as sense of target words. Our model could be viewed as an extension of LDA, but fit the WSI task more naturally and much better. We distinguish senses of target words from concepts of context words and assume that they are separate. Therefore, our model has two hidden layers corresponding to the sense of the target word and the concepts of the context words respectively. Basically, one decide the sense of the target word based on the concept configuration of context words, instead of tagging senses of target word to context words. The separation of senses of target word and concepts of context words is actually not only required by linguistic intuition but also leads to improvement by our experiment. Our model is also different from the naive Bayes model since our model induces senses of the target word at concept level while naive Bayes model works at word level and does not involve conceptualization to context words at all.

7 Conclusion

In this paper, we first proposed a parametric Bayesian generative model to the task of Word Sense Induction. It is distinct from previous work in that it introduces a layer of latent concepts that generalize the context words and thus enable the model to measure the sense similarity at a more general level. We also show in this paper that Zipf's law of meaning can be used to guide the setting of sense numbers on an all-word basis, which is not only simple but also independent of the clustering methods being used. We further extend our parametric model to non-parametric model which not only simplifies the problem of model selection but also bring improved performance. The test results on the benchmark datasets show that our model outperforms the state-of-the-art systems.

Acknowledgments

This work is supported by National Natural Science Foundation of China under Grant No. 61273318 and National Key Basic Research Program of China 2014CB340504.

References

Eneko Agirre and Aitor Soroa. 2007. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 7–12. Association for Computational Linguistics.

Javier Artiles, Enrique Amigó, and Julio Gonzalo. 2009. The role of named entities in web people search. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 534–542. Association for Computational Linguistics.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 103–111. Association for Computational Linguistics.

George Casella and Edward I George. 1992. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174.

Do Kook Choe and Eugene Charniak. 2013. Naive Bayes word sense induction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1433–1437, Seattle, Washington, USA, October. Association for Computational Linguistics.

Phillip Edmonds. 2006. Disambiguation, lexical. *Encyclopedia of Language and Linguistics. Second Edition. Elsevier*.

Wesam Elshamy, Doina Caragea, and William H Hsu. 2010. Ksu kdd: Word sense induction by clustering in topic space. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 367–370. Association for Computational Linguistics.

Roman Kern, Markus Muhr, and Michael Granitzer. 2010. Kcdc: Word sense induction by using grammatical dependencies and sentence phrase structure. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 351–354. Association for Computational Linguistics.

Ioannis P Klapaftis and Suresh Manandhar. 2010. Word sense induction & disambiguation using hierarchical random graphs. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 745–755. Association for Computational Linguistics.

Ioannis Korkontzelos and Suresh Manandhar. 2010. Uoy: Graphs of unambiguous vertices for word sense induction and disambiguation. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 355–358. Association for Computational Linguistics.

Jey Han Lau, Paul Cook, Diana McCarthy, David Newman, and Timothy Baldwin. 2012. Word sense induction for novel sense detection. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 591–601. Association for Computational Linguistics.

Jiwei Li and Claire Cardie. 2014. Timeline generation: Tracking individuals on twitter. In *Proceedings of the 23rd international conference on World wide web*, pages 643–652.

Jiwei Li and Sujian Li. 2013. Evolutionary hierarchical dirichlet process for timeline summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 556–560, Sofia, Bulgaria, August. Association for Computational Linguistics.

Suresh Manandhar, Ioannis P Klapaftis, Dmitriy Dligach, and Sameer S Pradhan. 2010. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68. Association for Computational Linguistics.

Zheng-Yu Niu, Dong-Hong Ji, and Chew-Lim Tan. 2007. I2r: Three systems for word sense discrimination, chinese word sense disambiguation, and english word sense disambiguation. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 177–182. Association for Computational Linguistics.

Ted Pedersen. 2010. Duluth-wsi: Senseclusters applied to the sense induction task of semeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 363–366. Association for Computational Linguistics.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, volume 7, pages 410–420.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123.

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical dirichlet processes. *Journal of the american statistical association*, 101(476).

Xuchen Yao and Benjamin Van Durme. 2011. Nonparametric bayesian word sense induction. In *Graph-based Methods for Natural Language Processing*, pages 10–14.

George Kingsley Zipf. 1945. The meaning-frequency relationship of words. *The Journal of General Psychology*, 33(2):251–256.

Inferring Knowledge with Word Refinements in a Crowdsourced Lexical-Semantic Network

Manel Zarrouk
UM2-LIRMM
161 rue Ada
34095 Montpellier, FRANCE
manel.zarrouk@lirmm.fr

Mathieu Lafourcade
UM2-LIRMM
161 rue Ada
34095 Montpellier, FRANCE
mathieu.lafourcade@lirmm.fr

Abstract

Automatically inferring new relations from already existing ones is a way to improve the quality and coverage of a lexical network and to perform error detection. In this paper, we devise such an approach for the crowdsourced JeuxDeMots lexical network and we focus especially on word refinements. We first present deduction (generic to specific) and induction (specific to generic) which are two inference schemes ontologically founded and then propose a transfer schema devoted to infer relations with and for word refinements.

1 Introduction

Efficiently building useful resources for Computational Linguistics (CL) is of a crucial interest. Most of existing lexical-semantic networks have been built by hand (like for instance WordNet (Miller et al., 1990)) and, despite that assisting tools are generally designed for consistency checking, the task remains time consuming and costly. Fully automated approaches are generally limited to term co-occurrences as extracting precise semantic relations between terms from corpora remains at best difficult. Crowdsourcing approaches are flowering in CL especially with the advent of Amazon Mechanical Turk or in a broader scope Wikipedia, to cite the most well known examples. WordNet is such a lexical network, constructed at great cost, based on synsets which can be roughly considered as concepts (Fellbaum, 1988). EuroWordnet (Vossen., 1998) a multilingual version of WordNet and WOLF (Sagot., 2008) a French version of WordNet, were built by automated crossing of the original Princeton WordNet and other lexical resources along with some more or less manual checking. Navigli (2010) constructed automatically BabelNet a large multilingual lexical network from term co-occurrences in Wikipedia. Although being very large and multilingually connected (which is tremendously usefull for machine translation, for instance) it contains few various lexical-semantic relations.

An *ideal* lexical-semantic network contains interconnected lemmas, word forms and multi-word expressions as entry points (nodes) along with word meanings and concepts. The idea itself of *word senses* as forwarded in the lexicographic tradition may be debatable in the context of resources for semantic analysis, and we generally prefer to consider the psycholinguistic idea of *word usages*. A given polysemous word, as identified by locutors, has several usages that might differ substantially from word senses as classically defined. A given usage can also in turn have several deeper refinements and the whole set of usages can take the form of a decision tree. For a very classical example, *bank* can be related to money or river : *bank* $\gtrdot$ *'bank>money'* and *bank* $\gtrdot$ *'bank>river'*. A *'bank>money'* can be distinguished as the financial institution or the actual building.

In the context of a collaborative construction, such a lexical resource should be considered as being constantly evolving and a general pragmatic rule of thumb is to have no definite certitude about the state of an entry. For a polysemous term, some refinements might be just missing at a given time

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 365–375, Dublin, Ireland, August 23-29 2014.

notwithstanding the evolution of language which might be very fast, especially in technical domains. There is no way (unless by inspection) to know if a given entry refinements are fully completed, and even if this question is really relevant.

Creating collaboratively a lexical-semantic network (or, in all generality, any similar resource) can be devised according to two broad strategies. Firstly, it can be designed as a contributive system like Wikipedia where people willingly add and complete entries (like for Wiktionary). Secondly, contribution can be undertaken indirectly thanks to games (also known as GWAP (vonAhn, 2008)). In this case, players do not need to be aware that while playing they are helping building a lexical and semantic resource. In any case, the built network is not free of errors which are (or should be) corrected along their discovery. Thus, a large number of obvious relations may be missing in the lexical network but are indeed necessary for a high quality resources usable in various NLP applications, or even crucial notably for textual semantic analysis.

For example, contributors seldomly indicate that a particular bird type can fly, as it is considered as an obvious generality. Only notable facts which are not easily deductible are naturally contributed. Conversly, well known exceptions are also generally contributed and take the form of a negative weight and anotated as such (for example, *fly* $\xrightarrow{agent:-100}$ *ostrich* [exception: bird]). In order to consolidate the lexical network, we adopt a strategy based on a simple inference mechanism to propose new relations from those already existing. The approach is strictly endogenous (i.e. self-contained) as it doesn't rely on any other external resources. Inferred relations are submitted either to contributors for voting or to experts for direct validation/invalidation. A large percentage of the inferred relations has been found to be correct However, a non negligible part of them are found to be wrong and understanding why is both interesting and useful. The explanation process can be viewed as a *reconciliation* between the inference engine and contributors who are guided through a dialog to explain why they found the considered relation incorrect. The possible causes for a wrong inferred relation may come from three possible origins: false premises that were used by the inference engine, exception or confusion due to some polysemy.

In (Sajous et al., 2013) an endogenous enrichment of Wiktionary is done thanks to a crowdsourcing tool. A quite similar approach of using crowdsourcing has been considered by (Zeichner, 2012) for evaluating inference rules that are discovered from texts. In (Krachina, 2006), some specific inference methods are conducted on text with the help of an ontology. Similarly, (Besnard, 2008) capture explanation with ontology-based inference. OntoLearn (Velardi, 2006) is a system that automatically build ontologies of specific domains from texts and also makes use of inferences. There have been also researchs on taxonomy induction based on WordNet (see (Snow, 2006)). Although extensive work on inference from texts or handcrafted resources has been done, almost none endogenously on lexical network built by the crowds. In this article, we first present the principles behind the lexical network construction with crowdsourcing and *games with a purpose* (also known as human-based computation games) and illustrated them with the JeuxDeMots (JDM) project. Then, we present the outline of an *elicitation engine* based on an *inference engine* using deduction, induction and especially relation transfer schemes. The reconciliation engine which presents the second part of the elicitation engine is detailed on previous papers (Zarrouk, LREC2014) (Zarrouk, TALN2013). An experimentation with a discussion is then detailed.

2 Crowdsourced lexical networks

For validating our approach, we used the JDM lexical network, which has been made freely available by its authors, and constructed thanks to a set of associatory games (Lafourcade, 2007). There is an increasing trend of using online GWAPs (game with a purpose (Thaler et al., 2011)) method for feeding such resources. Beside manual or automated strategies, contributive approaches are flowering and becoming more and more popular as they are both cheap to set up and efficient in quality.

The network is composed of terms (as vertices) and typed relations (as links between vertices) with weights. It contains terms and possible refinements. There are more than 50 types for relations, that range from ontological (hypernym, hyponym), to lexical-semantic (synonym, antonym) and to se-

mantic role (agent, patient, instrument). The weight of a relation is interpreted as a strength, but not directly as a probability of being valid. The JDM network is not an ontology with some pristine, factorized and well-thought hierarchy of concepts or terms. A given term can have a substantial set of hypernyms that covers a large part of the ontological chain to upper concepts. For example, hypernym(cat) = {feline, mammal, living being, pet, vertebrate, ...}. Heavier weights associated to terms are those felt by users as being the most relevant. On the 1st of January 2014, there are more than 6 800 000 relations and roughly 310 000 lexical items in the JDM lexical network (according to the figures given by the game site: http://jeuxdemots.org). To our knowledge, there is no other, in French at least, existing freely available crowdsourced lexical-network, especially with weighted relations, thus enabling strongly heuristics or psycho-linguistically motivated methods.

3 Inferring Semantic Relations...

Adding new relations to the JDM lexical network may rely on two components: (a) an inference engine and (b) a reconciliator. The inference engine proposes relations as if it was a contributor, to be validated by other human contributors or experts. In case of invalidation of an inferred relation, the reconciliator is invoked to try to assess why the inferred relation was found wrong. Elicitation here should be understood as the process to transform some implicit knowledge of the user into explicit relations in the lexical network. The core ideas about inferences in our engine are the following:

- inferring is to derive new premises (taking the form of relations between terms) from previously known premises, which are existing relations;
- candidate inferences may be logically blocked on the basis of the presence or the absence of some other relations;
- candidate inferences can be filtered out on the basis of a strength evaluation. The strong assumption here is to consider strengh as a confidence level, which is in fact only partially exact. More precisely, high strengh values clearly correlate to confidence, but we cannot say much about low strength values.

Figure 1: On the left, triangular deductive inference scheme where logical blocking based on the polysemy of the central term B which has two distinct meanings B' and B'' is applied. Arrows labelled $\geqslant$ are word meaning/refinements. The relation R? is the conclusion that may be blocked. On the right, (A *is-a* B) and (A R C) are the premises, and (B R C) is the induction proposed for validation. Term A may be polysemous with refinements holding premises, thus inducing a probably wrong relation.

3.1 ... by Deduction and by Induction...

Inferring by deduction (Zarrouk, RANLP2013) is a top-down scheme based on the transitivity of the ontological relation *is-a* (hypernym). If a term A is a kind of B and B holds some relation R with C, then we can expect that A holds the same relation type with C. The scheme can be formally written as follows:

$$\exists\, A \xrightarrow{is\text{-}a} B \quad \wedge \quad \exists\, B \xrightarrow{R} C \quad \Rightarrow \quad A \xrightarrow{R} C$$

For example, *shark* $\xrightarrow{is\text{-}a}$ *fish* and *fish* $\xrightarrow{has\text{-}part}$ *fin*, thus we can expect that *shark* $\xrightarrow{has\text{-}part}$ *fin*. The inference engine is applied on terms having at least one hypernym (the scheme could not be applied otherwise). Of course, this scheme is far too naive, especially considering the resource we are dealing with and may produce wrong relations. Indeed, the central term B is possibly polysemous and ways to avoid probably wrong inferences can be done through a *logical blocking*: if there are two distinct meanings for B that hold respectively the first and the second relation, then most probably

the inferred relation is wrong (see figure 1) and hence should be blocked. Moreover, if one of the premises is tagged by contributors as *true but irrelevant*, then the inference is blocked.

It is possible to evaluate a confidence level (on an open scale) for each produced inference, in a way that dubious inferences can be eliminated out through *statistical filtering*. The weight w of an inferred relation is the geometric mean of the weights of the premises (relations (A *is-a* B) and (B R C) in figure 1). If the second premise has a negative value, the weight is not a number and the proposal is discarded. As the geometric mean is less tolerant to small values than the arithmetic mean, inferences which are not based on two rather strong relations (premises) are unlikely to pass.

$$w(A \xrightarrow{R} C) = (\, w(A \xrightarrow{is-a} B) \times w(B \xrightarrow{R} C)\,)^{1/2} \quad \Rightarrow \quad w3 = (w1 \times w2)^{1/2}$$

Although making a transitive closure over a knowledge base is not new, doing so considering word usages (refinements) over a crowdsourced lexical network is an original approach. As for the deductive inference, induction (Zarrouk, RANLP2013) exploits the transitivity of the relation *is-a*. If a term A is a kind of B and A holds a relation R with C, then we might expect that B could hold the same type of relation with C. More formally we can write: $\exists\, A \xrightarrow{is-a} B \;\wedge\; \exists\, A \xrightarrow{R} C \;\Rightarrow\; B \xrightarrow{R} C$
For example, *shark* $\xrightarrow{is-a}$ *fish* and *shark* $\xrightarrow{has-part}$ *jaw*, thus we might expect that *fish* $\xrightarrow{has-part}$ *jaw*. This scheme is a generalization inference. The principle is similar to the one applied to the deduction scheme and similarly some logical and statistical filtering may be undertaken. The central term here A, is possibly polysemous (as shown in figure 1). In that case, we have the same polysemy issues with the deduction, and the inference may be blocked. The estimated weight for the induced relation is:

$$w(B \xrightarrow{R} C) = (w(A \xrightarrow{R} C))^2 \,/\, w(A \xrightarrow{is-a} B) \quad \Rightarrow \quad w2 = (w_3)^2 / w_1$$

3.2 ... and Performing Reconciliation

Inferred relations are presented to the validator to decide of their status. In case of invalidation, a reconciliation procedure is launched in order to diagnose the reasons: error in one of the premises (previously existing relations are false), exception or confusion due to polysemy (the inference has been made on a polysemous central term). A dialog is initiated with the user. To know in which order to proceed, the reconciliator checks if the weights of the premises are rather strong or weak.

Errors in the premises. We suppose that the relation (A *is-a* B) (in figures 1) has a relatively low weight. The reconciliation process asks the validator if that relation is true. It sets a negative weight to this relation if it is false so that the inference engine blocks further inferences. Else, if the relation (A *is-a* B) is true, we ask about the second relation (B R C or A R C) and proceed as above if the answer is negative. Otherwise, we check the other cases (exception, polysemy).

Errors due to exceptions. For the *deduction*, in case we have two trusted relations, the reconciliation process asks the validators if the inferred relation is a kind of exception relatively to the term B. If it is the case, the relation is stored in the lexical network with a negative weight and annotated as *exception*. Relations that are exceptions do not participate further as premises for deducing. For the *induction*, in case we have two trusted relations, the reconciliator asks the validators if the relation ($A \xrightarrow{R} C$) (which served as premise) is an exception relatively to the term B. If it is the case, in addition to storing the false inferred relation ($B \xrightarrow{R} C$) in the lexical network with a negative weight, the relation ($A \xrightarrow{R} C$) is annotated as *exception*. In the induction case, the exception is a true premise which leads to a false induced relation. In both cases of induction and deduction, the *exception* tag concerns always the relation ($A \xrightarrow{R} C$). Once this relation is annotated as an exception, it will not participate as a premise in inferring generalized relations (bottom-up model) but can still be used in inducing specified relations (top-down model).

Errors due to Polysemy. If the central term (B for deduction and A for induction) presenting a polysemy is mentioned as polysemous in the network, the refinement terms $term_1$, $term_2$, ... $term_n$ are presented to the validator so he can choose the appropriate one. The validator can propose new terms as refinements if he is not satisfied with the listed ones (inducing the creation of new appropriate refinements). If there is no meta information indicating that the term is polysemous, we ask

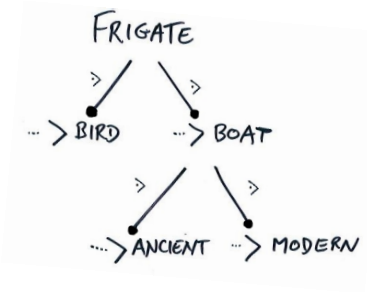


Figure 2: Refinement (noted $\gg$) tree of the term *frigate*. The first level discriminates between *frigate>bird* and *frigate>boat* which itself is refined between *(frigate>boat)>ancient* and *(frigate>boat)>modern*. This tree is a part of the lexical network which makes use of a specific refinement relation. Each refinement is connected to other terms of the network.

first the validator if it is indeed the case. After this procedure, new relations will be included in the network with positive values and the inference engine will use them later on as premises.

3.3 Transferring Relations with Refinements

A given polysemous word, as identified by locutors, has several usages that might differ substantially from word senses as classically defined. A given usage can also in turn have several deeper refinements and the whole set of usages can take the form of a decision tree. For example, *frigate* can be a bird or a ship. A *frigate>boat* can be distinguished as a modern ship with missiles and radar (*frigate>boat>modern*) or an ancient vessel with sails (*frigate>boat>ancient*). Having proper relations between refinements and other terms or refinements is crucial for word sense disambiguation.

The purpose of this scheme is to *enrich refinements and terms that are ontologically connected*. As its name indicates, this scheme requires the term A to have at least a refinement A' and at least one support relation that is ontological. The Relation Inference Scheme with Refinements (RIS_R) scheme, for each synonym, hypernym or hyponym (the support) B of the start term A, tries to share the outgoing relations between A' and B. The relations exchanged are the inferred relations to be validated or rejected latterly. To increase the relevance of the proposed relations, we make sure that some relation exists between the refinement term A' and the term B. For example, suppose we have A: $rose$ which has two refinements at least A': *rose>flower* and *rose>color* and a hypernym B: $plant$. In this example, the terms A': *rose>flower* and B: $plant$ are related (some relation exists between them) unlike the terms A': *rose>color* and B: $plant$. This strategy avoid proposing for example *rose>color* $\xrightarrow{has-part}$ *leaf* (an outgoing relation coming from B).

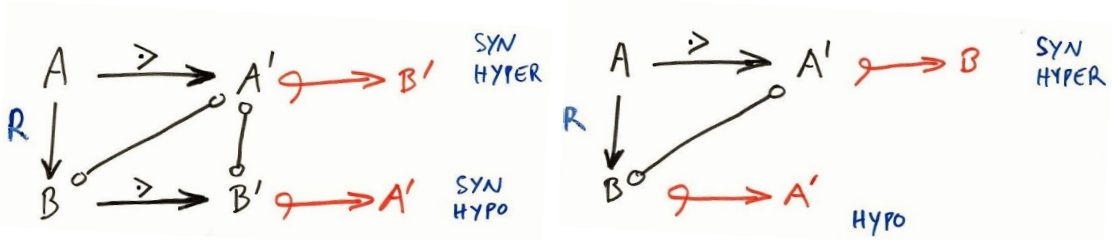


Figure 3: Relation Inference Scheme with Refinements (RIS_R). Above A (resp. B) has a refinement A' (resp. B'). Outgoing relations of A' are copied as outgoing relations of B' and vice-versa, according to the support relation (syn, hyper, hypo). On the right, we are in a *minimal* situation where B has no refinement.

Another strategy is not to propose outgoing relations from an hypernym to its hyponyms. The direction of the transfer is always from the hyponym to the hypernym because generally, outgoing relations of an hypernym are not all valid for its hyponyms. For example, for the term A: *animal* having a refinement A': *animal>zoology* which can have as parts *fin, scale, fang*... Those relations x $\xrightarrow{has-part}$ *(fin, scale, fang)* are not valid for the hyponym *cow*, for example.

This scheme has a behavior subtly different according to the nature of the term B (synonym, hypernym or hyponym) relatively to A. In figure 3, we use the following notations:

- $A \looparrowright B$: propose all the outgoing relations of A as outgoing relations for the term B (other notation as $\mathbb{C}$ to copy relations and $\mathbb{D}$ to displace them are available but not used here);
- $A \circ\!\!-\!\!-\!\!\circ B$: a relation between A and B in any direction exists.

4 Experimentations and Discussion

Our experiments consisted in applying and assessing the schemes presented above on the entire lexical network. This has been once during one run. At the time of writing of this article, the JeuxDeMots consists in more than 6 800 000 relations betweeen 310 000 terms. Specifically, it contains over 150 000 hypernym *is-a* relations, 170 000 syn relations and 27 000 hyponym relations.

Relation type	**Proposed %**
is-a (x is a type of y)	6.2
has-parts (x is composed of y)	25
holonyms (y specific of x)	7.2
typical place (of x)	7.2
charac (x as characteristic y)	13.7
agent-1 (x can do y)	13.3
instr-1 (x instrument of y)	1.7
patient-1 (x can be y)	1
place-1 (x located in the place y)	9.8
place > action (y can be done in place x)	3.4
object > mater (x is made of y)	0.3

Table 1: Percentages of relation proposed per relation type globally for deduction and induction.

4.1 Assessing Deduction and Induction

We applied the inference engine on around **32 000** randomly selected terms having at least one hypernym or one hyponym and thus produced by deduction more than **2 700 000** inferences and produced by induction over **430 000** relation candidates. The threshold for filtering was set to a weight of 25. This value is relevant as when a human contributor proposed relation is validated by experts, it is introduced with a default weight of 25 (the choice of this particular value is arbitrary and could have been different). The transitive *is-a* (Table1) is not very productive which might seem surprising at first glance. In fact, the *is-a* relation is already quite populated in the network, and as such, fewer new relations can be inferred. The figures are inverted for some other relations that are not so well populated in the lexical network but still are potentially valid. The *has-parts* relation and the agent semantic role (the *agent-1* relation) are by far the most productive types.

Deduction	**% valid**		**% error**		
Rel type	**rlvt**	**¬ rlvnt**	**prem**	**excep**	**pol**
isa	76%	13%	2%	0%	9%
has-parts	65%	8%	4%	13%	10%
holonyme	57%	16%	2%	20%	5%
typ place	78%	12%	1%	4%	5%
charac	82%	4%	2%	8%	4%
agent-1	81%	11%	1%	4%	3%
instr-1	62%	21%	1%	10%	6%
patient-1	47%	32%	3%	7%	11%
place-1	72%	12%	2%	10%	6%
place>act	67%	25%	1%	4%	3%
obj>mater	60%	3%	7%	18%	12%

Induction	**% valid**		**% error**		
Rel types	**rlvt**	**¬rlvnt**	**prem**	**excep**	**pol**
isa	-	-	-	-	-
has-parts	78%	10%	3%	2%	7%
holonyme	68%	17%	2%	8%	5%
typ place	81%	13%	1%	2%	3%
charac	87%	6%	2%	2%	3%
agent-1	84%	12%	1%	2%	1%
instr-1	68%	24%	1%	4%	3%
patient-1	57%	36%	3%	2%	2%
place-1	75%	16%	2%	5%	2%
place>act	67%	28%	1%	3%	1%
obj>mater	75%	10%	7%	5%	3%

Table 2: On the left, number of propositions produced by *deduction* and ratio of relations found as true or false. On the right, Number of propositions produced by *induction* and ratio of relations found as true or false.

Table 2 presents some evaluations of the status of the inferences proposed by the inference engine through deduction and induction respectively. Inferences are valid for an overall of 80-90% with around 10% valid but not relevant (like for instance *dog* $\xrightarrow{has-parts}$ *proton*). We observe that error number in premises is quite low, and errors can be easily corrected. Of course, not all possible errors are detected through this process. More interestingly, the reconciliation allows in 5% of the cases to

RIS_R	# existed	# proposed	productivity
syn	38 792	105 288	271.41%
hyper	139 490	101 908	73.05%
hypo	38 756	101 336	261.47%

Table 3: The number of relations existing before application of the scheme and those proposed by the scheme. The statistics were made on the terms on which the scheme has proposed inferences

identify polysemous terms and refinements. Globally false negatives (inferences voted false while being true) and false positives (inferences voted true while being false) are evaluated to less than 0,5%. For the induction process, the relation *is-a* is not obvious (a lexical network is not reductible to an ontology and multiple inheritance is possible). Result seems about 5% better than for the deduction process: inferences are valid for an overall of 80-95%. The error number is quite low. The main difference with the deduction process is on errors due to polysemy which is lower with the induction process. To try to assess a baseline for those results, we compute the full closure of the lexical network, i.e. we produce iteratively all possible candidate relations until no more could be found, each candidate being considered as correct and participating to the process. We got more than 6 million relations out of which 45% were wrong (evaluated on around 1 000 candidates randomly chosen).

4.2 Assessing Relation Transfer

We applied the scheme of refinements relation transfer with three different support relations:
• RIS_R (synonym): the scheme applied with syn as support (in case of existence of B' the terms A' and B' share relations.)
• RIS_R (hyponym): the scheme applied with hypo (relations are shared from B or B' to A')
• RIS_R (hypernym): the scheme applied with R=hyper (relations are shared from A' to B or B').
RIS_R stands for *Relation Inference Schema with Refinements*.

Relation type	**syn**	**hypo**	**hyper**
associated	50 946	39 325	51 960
has-part	13 362	13 120	8 049
is-a	3 711	5 114	5 707
hyponym	6 463	10 186	6 326
holonym	1 927	1 407	3 757
charac	10 378	10 063	7 614
location	5 921	9 251	5 529
agent-1	6 887	9 366	3024
other	5 693	4 076	9 370

Relation type	**syn**	**hypo**	**hyper**
associated	92.4%	65%	60.8%
has-part	93.2%	46.9%	80.8%
is-a	86.2%	56.4%	46%
hyponym	69.7%	60%	65%
holonym	74.4%	60.7%	64.2%
charac	91.5%	73.9%	90.5%
location	91.1%	81%	79.5%
agent-1	92.1%	78.9%	90.9%

Table 4: On the left, relations proposed by type of the support relation and relation type of the conclusion. On the right, percentage of valid relations by type of the support relation and relation type of the conclusion.

Relation Transfer Productivity - Since the schema has a condition to be applied, the propositions (inferred relations) are made for only **6 349** terms fullfilling the constraints. The whole process produced **308 532** inferences presenting totally new relations not existing before in the network which make about **49** new relations per entry. The RIS_R(syn) produced **2.7** times the existing relations which make it the most productive version, followed by the RIS_R(hypo) producing **2.6** times and the RIS_R(hyper) with a productivity of **0.73** (table 3). The inferred relations are detailed by relation type in the left table 4. The different relation types are variously productive, and this is mainly due to the number of existing relations and the distribution of their type. The "associated" type is the most proposed from both three schemes and this is explained by the large semantic spectre of this relation type since it refers to every term associated to the target term. In the network, the most possessed relations of a term are typed with the associated relations. The amount of the relations proposed is related to the one existing in the network. If a relation type is quite populated in the network, fewer new relations can be inferred. The figures are inverted for some other relations that are not so well populated in the lexical network but still are potentially valid.

Relation Transfer Accuracy - The validation process was applied manually on a sample of around 1 000 propositions randomly choosen for each scheme. The synonym version has the highest accuracy with **90.76 %** valid relations, hypernym version with **72.69 %** and **66.24 %** for the hyponym version (table 4). The synonym version of the scheme has systematically the best accuracy for all the relation types. Some accuracy percentages are lower than others for some reasons. In certain cases, some outgoing relations of an hyponym do not suit for the hypernym. For example:
• A: animal • A': animal>animalia • $B(hypo)$: cat
⇒ The inference scheme will propose the outgoing relation of cat (*cat* $\xrightarrow{is-a}$ *pet*) to $animal$>$animalia$ (*animal>animalia* $\xrightarrow{is-a}$ *pet*) which is wrong and this explain the weak percentage of accuracy for example of the relation *is-a* (56.4% by the RIS_R(hypo) and 46% by the RIS_R(hyper)) and $has-part$ (46.9% by the RIS_R(hypo)).
Another reason is that in the network, some terms are not refined (or not completely refined) which can lead to some wrong relations, as for example: • A: cheese • A': cheese>dairy product • $B(hypo)$: goat [1]
⇒ The inference scheme will propose the relation (*cheese>dairy product* $\xrightarrow{has-part}$ *teats*) which is wrong and thus because the term $goat$ is not yet refined into *goat>dairy product* and *goat>animal*.

From the figures, we can make the following observations. First, global results show that produced inferences are strongly valid with synonyms. The results are poorer with hypernyms and hyponyms (table 4) which is obvious regarding that with synonym, the terms exchanging relations are roughly at the same level of the taxonomic hierarchy which is not the case when they are related with an hyponym or hypernym relation.

5 Conclusion

We have presented some issues in inferring new relations from existing ones to consolidate a lexical-semantic network built with games and user contributions. To be able to enhance the network quality and coverage, we proposed an elicitation engine based on inferences (induction, deduction and relation transfer with refinements) and reconciliation. If an inferred relation is proven wrong, a reconciliation process is conducted in order to identify the underlying cause and solve the problem.

We focused our work on the transfer of relations related to word usage (refinements) with help of a support relation being either synonym, hypernym or hyponym. Unlike deduction and induction, the transfer scheme does not rely directly on the relation (*is-a*), but merely on terms that may be ontologicaly connected to the target. Experiments showed that relation transfer for refinements is quite productive (compared to deduction and induction), and is satisfying in correctness especially with synonym as support relation. The most obvisous reason is that in general a (quasi-)synonym is almost at the same level with the target term, and at least much more often than a hypernym or hyponym. User evaluation showed that wrong inferred relations (between around 20-15% of all inferred relations) are still logically sound and could not have been dismissed *a priori*. Relation transfer with refinements can conclusively be considered as a usefull and efficient tool for relation inference, and it may be really crucial as support for building information to be used in word sense disambiguation. In particular, it can help proposing hypernyms for the target term when they are missing, making possible further deductions or inductions. Hence, a virtuous circle may be initiated.

Still, the main difficulty of such approach relies in setting the various parameters in order to achieve an appropriate and fragil tradeoff between an over-restrictive filter (many false negatives, resulting in information losses) and a too lenient engine (many false postive, resulting in more human effort). The elicitation engine we presented through schemes based on deduction, induction and more precisely on relation transfer is an efficient error detector and a polysemy identifier. The actions taken during the reconciliation forbid an inference proven wrong or exceptional to be inferred again. Each inference scheme may be supported by the two others in particular for refinements, and if a given inference has been produced by more than one of these three schemes, it is almost surely correct.

[1] In french, some dairy products are called sometimes by the name of the producer animal, like $chevre$(*goat*) for the cheese made from the goat's milk

An additional inference scheme, *abduction*, reinforced our inference engine and guided it through producing accurate new relations with an interesting accuracy. This scheme can be viewed as an example based strategy. Hence abduction relies on similarity between terms, which may be formalized in our context as sharing some outgoing relations between terms. The abductive inferring layout supposes that relations held by a term can be proposed to similar terms. Abduction first selects a set of similar terms to the target term A which are considered as proper examples. The outgoing relations from the examples which are not common with those of A are proposed as potential relations for A and then presented for validation/invalidation to users. Unlike induction and deduction, abduction can be applied on terms with missing or irrelevant ontological relations, and can generate ontological relations to be used afterward by the inference loop. This scheme was detailed in our paper (M. Zarrouk, EACL2014).

Researches are undertaken on (semi)automating the inference schemes or inference rules (scheme with just one or two unknown terms) discovery by our elicitation system. Enhancements are also considered on our previous schemes as for exemple defining the inference's scope especially in deduction and induction (example: what to do to avoid transferring invalid inferences from the term *animal* as *has-part wings* to its hyponyms like *cat* or *fish*).

We are also modelling a declarative query language that allows users to manipulate the lexical-semantic network and to apply our elicitation engine according to their needs while remaining focused on their request and without drifting in database access or linguistic domain.

References

von Ahn, L. and Dabbish, L. 2008. *Designing games with a purpose.* in Communications of the ACM, number 8, volume 51.p58-67.

Besnard, P. Cordier, M.O., and Moinard, Y. 2008. *Ontology-based inference for causal explanation..* Integrated Computer-Aided Engineering , IOS Press, Amsterdam , Vol. 15 , No. 4 , 351-367 , 2008.

Fellbaum, C. and Miller, G. 1988. *(eds) WordNet..* The MIT Press.

Krachina, O., Raskin, V. 2006. *Ontology-Based Inference Methods.* CERIAS TR 2006-76, 6p.

Lafourcade, M. 2007. *Making people play for Lexical Acquisition..* In Proc. SNLP 2007, 7th Symposium on Natural Language Processing. Pattaya, Thailande, 13-15 December. 8 p.

Lafourcade, M., Joubert, A. 2008. *JeuxDeMots : un prototype ludique pour l'Ã l'mergence de relations entre termes..* In proc of JADT'2008, Ecole normale supÃ l'rieure Lettres et sciences humaines , Lyon, France, 12-14 mars 2008 .

Lafourcade, M., Joubert, A. 2012. *Long Tail in Weighted Lexical Networks..* In proc of Cognitive Aspects of the Lexicon (CogAlex-III), COLING, Mumbai, India, December 2012.

Lieberman, H, Smith, D. A and Teeters, A 2007. *Common consensus: a web-based game for collecting commonsense goals..* In Proc. of IUI, Hawaii,2007.12p .

Marchetti, A and Tesconi, M and Ronzano, F and Mosella, M and Minutoli, S. 2007. *SemKey: A Semantic Collaborative Tagging System..* in Procs of WWW2007, Banff, Canada. 9 p.

Mihalcea, R and Chklovski, T. 2003. *Open MindWord Expert: Creating large annotated data collections with web users help..* In Proceedings of the EACL 2003, Workshop on Linguistically Annotated Corpora (LINC). 10 p.

Miller, G.A. and Beckwith, R. and Fellbaum, C. and Gross, D. and Miller, K.J. 1990. *Introduction to WordNet: an on-line lexical database..* International Journal of Lexicography. Volume 3, p 235-244.

Navigli, R and Ponzetto, S. 2010. *BabelNet: Building a very large multilingual semantic network..* in Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11-16 July 2010.p 216-225.

Sagot, B. and Fier, D. 2010. *Construction d'un wordnet libre du franï£¡ais ï£¡ partir de ressources multilingues..* in Proceedings of TALN 2008, Avignon, France, 2008.12 p.

Thaler, S and Siorpaes, K and Simperl, E. and Hofer, C. 2011. *A Survey on Games for Knowledge Acquisition..* STI Technical Report, May 2011.19 p.

Sajous, F., Navarro, E., Gaume, B,. Prï£¡vot, L. and Chudy, Y. 2013. *Semi-Automatic Enrichment of Crowdsourced Synonymy Networks: The WISIGOTH system applied to Wiktionary..* Language Resources & Evaluation, 47(1), pp. 63-96.

Siorpaes, K. and Hepp, M. 2008. *Games with a Purpose for the Semantic Web..* in IEEE Intelligent Systems, number 3, volume 23.p 50-60.

Snow, R. Jurafsky, D., Y. Ng., A. 2006. *Semantic taxonomy induction from heterogenous evidence.* in Proceedings of COLING/ACL 2006, 8 p.

Velardi, P. Navigli, R. Cucchiarelli, A. Neri, F. 2006. *Evaluation of OntoLearn, a methodology for Automatic Learning of Ontologies.* in Ontology Learning and Population, Paul Buitelaar Philipp Cimmiano and Bernardo Magnini Editors, IOS press 2006).

Vossen, P. 2011. *EuroWordNet: a multilingual database with lexical semantic networks..* Kluwer Academic Publishers.Norwell, MA, USA.200 p.

Zarrouk, M., Lafourcade, M. and Joubert, A. 2013. *Inference and reconciliation in a lexical-semantic network.* 14th International Conference on Intelligent Text Processing and Computational Linguistic (CICLING-2013), 13 p.

Zarrouk, M., Lafourcade, M. and Joubert, A. 2013. *Inductive and deductive inferences in a Crowdsourced Lexical-Semantic Network.* 9th International Conference on Recent Advances in Natural Language Processing (RANLP 2013), 6 p.

Zarrouk, M., Lafourcade, M. and Joubert, A. 2014. *About Inferences in a Crowdsourced Lexical-Semantic Network*. In proc of 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014), 8 p.

Zarrouk, M., Lafourcade, M. and Joubert, A. 2013. *InfÃl'rences dÃl'ductives et rÃl'conciliation dans un rÃl'seau lexico-sÃl'mantique*. 20Ãl'me confÃl'rence du Traitement Automatique du Langage Naturel 2013 (TALN 2013), 14 p.

Zarrouk, M.and Lafourcade, M. 2014. *Relation Inference in Lexical Networks ... with Refinements*. The 9th edition of the Language Resources and Evaluation Conference, 26-31 May, Reykjavik, Iceland, 6 p.

Zeichner, N., Berant J., and Dagan I. 2012. *Crowdsourcing Inference-Rule Evaluation*. in proc of ACL 2012 (short papers).

A Supervised Learning Approach Towards Profiling the Preservation of Authorial Style in Literary Translations

Gerard Lynch
Centre for Applied Data Analytics Research
University College Dublin
Ireland
firstname.lastname@ucd.ie

Abstract

Recently there has been growing interest in the application of approaches from the text classification literature to fine-grained problems of textual stylometry. This paper seeks to answer a question which has concerned the translation studies community: how does a literary translator's style vary across their translations of different authors? This study focuses on the works of Constance Garnett, one of the most prolific English-language translators of Russian literature, and uses supervised learning approaches to analyse her translations of three well-known Russian authors, Ivan Turgenev, Fyodor Dosteyevsky and Anton Chekhov. This analysis seeks to identify common linguistic patterns which hold for all of the translations from the same author. Based on the experimental results, it is ascertained that both document-level metrics and n-gram features prove useful for distinguishing between authorial contributions in our translation corpus and their individual efficacy increases further when these two feature types are combined, resulting in classification accuracy of greater than 90 % on the task of predicting the original author of a textual segment using a Support Vector Machine classifier. The ratio of nouns and pronouns to total tokens are identified as distinguishing features in the document metrics space, along with occurrences of common adverbs and reporting verbs from the collection of n-gram features.

1 Introduction

The application of *supervised learning* technologies to textual data from the humanities in order to shed light on stylometric questions has become more popular of late. In particular, these approaches have been applied to questions from the field of translation studies, which concern the notion of *translationese*[1] detection in Italian and other languages, (Baroni and Bernardini, 2006; Ilisei et al., 2010; Ilisei and Inkpen, 2011; Popescu, 2011; Koppel and Ordan, 2011; Lembersky et al., 2011). Work has also been carried out on source language detection from translation corpora, (van Halteren, 2008; Lynch and Vogel, 2012) and translation direction detection in parallel MT training corpora, (Kurokawa et al., 2009), which can have applications in the domain of machine translation where the direction of bilingual translation corpora has been shown to impact on the accuracy of automated translations using such corpora[2].

This work seeks to apply these methods to the task of identifying authorial style within a corpus of translations by the same translator. Venuti (1995) mentions the concept of the *translator's invisibility*, that the measure of the best translator is that their style is not distinguishable in the translation, that their main concern and focus is to deliver the original text in a faithful manner. Of course, this task is often subject to their own vocabulary choices and as was often the case, cultural or personal bias of the translator or the regime or government in which they were operating. Identifying the former case will

[1] The subset or dialect of language which consists solely of translations from another language.

[2] Translating FR-EN, a smaller bilingual corpus of French translated to English provides similar qualitative results (BLEU score) to a larger corpus consisting of English translated to French.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 376–386, Dublin, Ireland, August 23-29 2014.

be the focus of this work, as choices of vocabulary or sentence construction can be isolated through the application of machine learning methods, although the latter is also a highly interesting question, albeit a more complex one to tackle using the methods at hand.[3]

2 Previous work

Baroni and Bernardini (2006) were among the first to apply advanced machine learning techniques to questions of textual stylometry, although use of linguistic features and metrics was already established in studies such as Borin and Pruetz (2001) who worked on POS distributions in translated Swedish and work by Mikhailov and Villikka (2001) who examined translated Finnish using statistical methods and metrics from authorship attribution. Baroni and Bernardini (2006) investigated a corpus of translated and original text from an Italian current affairs journal using a Support Vector Machine classifier, managing ca. 87% accuracy in distinguishing the two textual classes. Their study also investigated the performance of humans on such a task and found that the machine learning algorithm was more consistent although it was outperformed by one of the expert human analysts. Ilisei et al. (2010) used textual features such as type-token ratio and readability scores in their work on detecting translated text in Spanish and obtained comparable accuracy to Baroni and Bernardini (2006) who mostly used mixed POS and word n-grams. Popescu (2011) employed a different approach using a feature set consisting of n-grams of characters, and maintained reasonable accuracy in classifying translated literary works from originals.

Koppel and Ordan (2011) concerned themselves with the concept of *dialects* of translationese and whether translations from the same source language were more similar to one another than translations from different source languages and to what extent genre affected translationese. In their experiments on the Europarl corpus and a three source-language corpus from the International Herald Tribune, they found that training on one corpus and testing on another reported low accuracy, indicating genre effects, coupled with the fact that training on a corpus of translations from one source language and testing on a corpus translation from another source language obtained poorer results than using a corpus of translations from several source languages.

van Halteren (2008) investigated the predictability of source language from a corpus of Europarl translations and predicted source language with an accuracy of over 90%, using multiple translations of a source text in different languages. Distinguishing features from the Europarl corpus included phrases such as *a certain number* in texts of French origin, *framework conditions* in texts of a German origin and various features that were particular to the nature of the corpus as a collection of parliamentary speeches.[4] More recently, Lynch and Vogel (2012) revisited the source language detection task with a focus on literary translations, and obtained classification accuracy of ca. 80% on a corpus of translations into English from Russian, German and French using a feature set containing a combination of ratios of parts of speech and POS n-grams. Texts translated from French had a higher ratio of nouns to total words than the other two categories, and the frequency of contractions such as *it's* and *that's* varied between the subcorpora.

Focusing on the stylistic variation of individual translators from the point of view of researchers in translation studies, Baker (2000) defined frameworks for performing stylistic analyses of translator's using quantitative methods. Her own examples examined translators of Portuguese and Arabic and focused on the translation of common verbs, such as *say* and *tell*. She found that the frequency of these verbs was a distinguishing metric between translators but was careful to mention that these features might vary depending on the corpora in question. Winters (2007) profiled translator style in two translations of F. Scott Fitzgeralds *The Beautiful and the Damned*, focusing on modal particles and speech act reporting verbs as a distinguishing aspect of translatorial style. Vajn (2009) applied textual metrics such as type-token ratio and relative vocabulary richness to two translations of Plato's *Republic* to investigate the variation between two translations by Benjamin Jowett and Robin Waterfield and developed a theory of co-authorship to explain the complementary stylistic effect of authorial and translatorial style.

[3] See Li et al. (2011) and Wang and Li (2012) for examples of studies of translation from Chinese and English which take the cultural background of translators into account when discussing distinguishable features.

[4] German native speakers addressed the congregation in a different manner to English native speakers, for example.

Ongoing work in translation studies and digital humanities have examined the question of translatorial vs. authorial style using computational analyses. Burrows (2002) investigated the stylistic properties of several English translations of Roman poet Juvenal using his own Delta metric developed for authorship attribution and the frequencies of common words, Lucic and Blake (2011) investigated two translations of German author Rainer Maria Rilke in English using the Stanford Lexical Parser and found differing patterns of syntactic structure such as negation modifiers and adverbial modifiers.[5]

Recently, Forsyth and Lam (2013) analysed two parallel English translations of the French-language correspondence of Theo and Vincent Van Gogh using k-nearest neighbour classifiers and a feature set consisting of the sixty-nine most frequent words and found that a distinct authorial style for each of the brothers was preserved in both translations, with translatorial style also proving distinguishable, albeit to a lesser extent than its authorial counterpart. Lynch (2013) investigated two English translators of Henrik Ibsen's dramas using machine learning methods and found that document metrics and n-gram features similar to those used in this current study proved accurate in distinguishing authorship of parallel translations of the same source, and also that document metrics such as average sentence length distributions learned from translations of different works by the same author could be used to classify the author of a parallel translation, indicating that the translators' styles were learnable across a diverse corpus of works by the same author.

Rybicki (2006) used Burrow's Delta to investigate the stylistic nature of character idiolects in dramatic translation, focusing on Polish drama, and found that the translated idiolects tended to cluster in similar patterns[6] to the idiolects in the original text. Lynch and Vogel (2009) worked on a similar topic, the clustering of character idiolects in English and German translations of Henrik Ibsen's plays using the χ^2 metric. Rybicki and Heydel (2013) used Burrow's Delta, and dendrogram clustering to investigate the case of a Polish translation of Virginia Woolf's *Night and Day* and found that the method identified the point in the novel where one translator had taken over from another[7] Rybicki (2012) had previously used these techniques to distinguish translatorial style in a large corpus of Polish translations and concluded that such style was not to be captured using the methods at hand, which consisted of using Burrow's Delta metric with five thousand of the most frequent words. Although the metric performed well at clustering translations by author, it failed to cluster translations by translator, leading the author to conclude that as Venuti (1995) had claimed, the best translators are in fact invisible.

Although these studies are generally of an exploratory nature and often seek to draw conclusions about particular literary works and figures, the methodologies used are general to textual stylometry and have been successfully applied to emerging tasks in computational linguistics such as MT quality estimation, (Felice and Specia, 2012), personality detection (Mairesse and Walker, 2008), sentiment analysis (Gamon, 2004), fraud detection (Goel and Gangolly, 2012) and many other studies where textual analyses are pertinent.

3 Motivation and background to study

In this study, the translations of a literary translator of a number of different authors are examined in order to measure the extent to which authorial style is preserved by the translator in question. This analysis encompasses features represented by n-grams of words or POS tags and also stylometric metrics based on whole texts, such as type-token ratio, lexical richness and readability scores. Previous work (Rybicki and Heydel, 2013; Burrows, 2002; Rybicki, 2012; Koppel and Ordan, 2011) focused on lists of highly frequent words in their analysis of translations. By using supervised learning techniques, it is possible to investigate exactly which words are discriminating between author's idiolects in translation, regardless of frequency, together with abstract representations of word types and textual metrics, which present an alternative overview of the data in question.

This study examines the translations of British translator Constance Garnett (1861-1946) from the Russian originals written by Fyodor Dosteyevsky, Ivan Turgenev and Anton Chekhov. Moser (1988) and

[5] *not* and *nearly*.

[6] Villians with villians, heroes with heroes and female and male characters formed separate clusters

[7] The original translator passed away before she could finish the translation, hence the completion by another party.

Remnick (2005) write about Garnett's[8] life, describing her early days as a student of Latin and Greek in Cambridge, marriage to publisher and literary figure Edward Garnett and her chance introduction to Russian literature by the chance meeting with a young revolutionary in London. Along with the three aforementioned characters, she also translated works by Leo Tolstoy and Nikolai Gogol, Alexander Ostrovsky and Alexander Herzen, seventy works in all.

According to Moser (1988), her reputation was firmly established with her translations of Turgenev and thereafter Garnett was more or less responsible for igniting the English language-world's love affair with Dosteyevsky. Her translations were not without criticism however, Moser (1988) mentioning that Edmund Wilson believed she caused Russian authors to sound *more or less the same*, a claim echoed later by Joseph Brodsky who remarked that the average Western English-language reader cannot distinguish Tolstoy's voice from Dosteyevsky's, as they are in fact reading Constance Garnett's own voice.

Indeed, Remnick (2005) describes Garnett's translation style and mentions how she translated at breakneck speed, often skipping over sections which she did not understand. He also mentions Vladimir Nabokov's disdain for Garnett's translations, who was known to scribble vitriolic notes in the margins of Garnett translations during his tenure as a professor at Cornell and Wellesley in the United States. Remnick notes that children's book author Kornei Churnosky praised her translations of Turgenev and Chekhov but was less than pleased with her rendering of Dosteyevsky, complaining that she had smoothed over the erratic and challenging original text of that particular author. Thus, this work focuses on these claims of distinguishability in particular, for it is exactly these characteristics that can, in principle, be investigated using *supervised learning* techniques: Is it the case that one can automatically distinguish Garnett's renderings of Dosteyevsky from her translations of Turgenev, and if so, based on which textual characteristics, word distributions or individual word frequencies?

4 Corpus and methodology

The corpus was limited in these experiments to works by Dosteyevsky, Turgenev and Chekhov as these were the three authors translated by Garnett for which the most public domain text was available. Texts were downloaded from Project Gutenberg.[9]The final corpus consisted of eight works by Turgenev, seven works by Dosteyevsky and eleven collections of short stories by Chekhov. A selection of random text was made from each work matching the size of the smallest possible size of a work by each author and this selection was then divided into chunks of ten kilobytes each. The resulting corpus contains 942 segments from the three authors, 330 from Chekhov, 192 from Turgenev and 420 from Dosteyevsky. TagHelperTools was used to create the n-gram tokens, (Rosé et al., 2008) and calculate nineteen document statistics using TreeTagger, (Schmid, 1994) to tag texts for parts-of-speech. Weka, (Frank et al., 2005) was used for the *supervised learning* experiments, the SMO implemenation of a Support Vector Machine classifier along with the Naive Bayes and Simple Logistic Regression algorithms were used in the experiments.

The eighteen document level metrics used in the experiments are listed in Table 2. These were influenced by features used by Ilisei et al. (2010) in work which examined the problem of *translationese* detection in Spanish text. The two readability metrics employed are the Coleman-Liau Index, (Coleman and Liau, 1975) and the Automated Readability Index, (Smith and Senter, 1967). The n-gram features are calculated using TagHelper tools and the frequency of these features were reduced to a binary variable detailing the occurrence or non-occurrence of each feature in each segment.

5 Experiments

5.1 Document-level metrics

Experiments were carried out using different feature sets on the corpus described in Section 4. The experiments seek to classify the original author of a translated textual segment. The SVM classifier managed to achieve 87% accuracy when averaged using ten-fold cross validation on the whole corpus

[8](*nee* Black)

[9]www.gutenberg.org

Work	Author	Work	Author
The Bishop & O. Stories	Chekhov	The Cook's Wedding	Chekhov
The Chorus Girl	Chekhov	The Darling	Chekhov
The Duel	Chekhov	The Horse-Stealers	Chekhov
The School Master	Chekhov	The Party	Chekhov
The Wife	Chekhov	The Witch	Chekhov
Love & O. Stories	Chekhov	A Raw Youth	Dosteyevsky
Brothers Karamasov	Dosteyevsky	Crime & Punishment	Dosteyevsky
The Insulted and The Injured	Dosteyevsky	The Possessed	Dosteyevsky
White Nights	Dosteyevsky	Five Stories	Dosteyevsky
A House of Gentlefolk	Turgenev	Fathers & Children	Turgenev
On The Eve	Turgenev	Knock,Knock,Knock	Turgenev
Rudin	Turgenev	Smoke	Turgenev
The Torrents of Spring	Turgenev	The Jew	Turgenev

Table 1: Literary works in study

Feature	Desc.	Feature	Desc.
nounratio	nouns vs. total words	avgwordlength	average word length
pnounratio	pronouns vs. total words	prepratio	prepositions vs total words
lexrich	lemmas vs. total words	grammlex	closed vs. open class
complextotal	>1 verb: total sent.	simple complex	> 1 verb : <= 1 verb
simpletotal	<= 1 verb : total sent.	avgsent	average sentence length
infoload	open-class : total words	dmarkratio	discourse markers : total words
CLI	readability metric	fverbratio	finite verbs : total words
conjratio	conjunctions : total words	ARI	readability metric
numratio	numerals : total words	typetoken	word types : total words

Table 2: Document-level metrics used

using document-level features only. This result suggests that the authorial style of the three authors in question has indeed been preserved in translation.

Examining the features ranked by information gain in Table 3, it is clear that the ratio of nouns to total words and the ratio of pronouns to total words are highly distinguishing between the original authors. Ratio of prepositions to total words and the type-token ratio also feature in more elevated positions on the list than readability scores and sentence length measures.

5.2 N-gram features

The next set of experiments concerned the use of n-gram features, namely word unigram and POS bi-grams. For the word features, all noun features were removed as these, while providing clues to the identity of the author of a translation, are arguably not universal features of authorial style[10]. Verb features were not removed in such a fashion, however it may be argued that these also contain topical information and should be treated with caution. The remaining features were ranked by efficacy using the information gain metric and ten-fold cross validation and a subset of one hundred features were used for the classification experiments.

The SVM classifier in Weka with a linear kernel obtained 89.5% accuracy using a dataset of 100 words. The Simple Logistic regression classifier obtained 91.5% accuracy using the same feature set. This feature set was obtained by ranking the total list of word unigrams using information gain over ten-fold cross validation and removing the noun features as mentioned above. These high accuracy scores

[10]There is interest in lexical variation in translation, (Kenny, 2001) but this work focuses on stylistic features such as verbs and closed-class words as they are less prone to bias from the themes or topics in a text

obtained further reinforce the results obtained by using the document-level metrics, that a distinct textual style is learnable from the translations by Garnett of Dosteyevsky, Tolstoy and Turgenev. A number of these features and their relative frequencies are displayed in Table 6.

Feature	Rank.	Feature	Rank.
nounratio	1	avgwordlength	2
pnounratio	3	prepratio	4
typetoken	5	lexrich	6
simpletotal	7	simplecomplex	8
complextotal	9	grammlex	10
avgsent	11	infoload	12
cli	13	fverbratio	14
numratio	15	ari	16
conjratio	17	dmarkratio	18

Table 3: Metrics ranked using information gain and ten-fold cross validation

Feature set	Algorithm.	Accuracy
18 doc metrics	SVM	87%
18 doc metrics	Naive Bayes	74.2%
18 doc metrics	Naive Bayes	87.89%
100 words	SVM	89.5%
100 words	SimpLog	91.5%
1021 POS bigrams	SVM	83 %
1021 POS bigrams	SimpLog	78.98%
1021 POS bigrams	Naive Bayes	80%
1153 mix	SVM	95%
1153 mix	SVM	94.6 %
1153 mix	SimpLog	95%

Table 4: Accuracy overview

Using the 1021 unique POS bigrams which are present in the corpus as features, 83% classification accuracy was obtained using the SVM classifier, with Naive Bayes and Simple Logistic Regression managing 80% and 78.98% respectively.

5.3 Combined feature sets

Combining the feature sets from each of the experiments above, accuracy is improved. SVM obtains 95% accuracy, Naive Bayes and Simple Logistic Regression manage 94.6% and 95% respectively. This combined set contains 1153 features, 1021 POS bigrams, one hundred words and eighteen document level features. Ranking these features using ten-fold cross validation and Information Gain, the ranking displayed in Table 5 is obtained. Word unigrams and document-level features dominate the top fifty ranked features, with a number of POS-bigrams also occurring in the list.

6 Discussion

Tables 6 and 7 reflect the individual characteristics of each of the three authorial subcorpora examined here. The translations of Turgenev are distinguished by the higher average frequencies of the verbs *observed*, *repeated* and *replied*. Taking the value of the document-level metrics into account, Turgenev is to some extent unremarkable by these measures, although his works report higher average values for the two readability metrics, CLI and ARI, than the other two authors. The translations of Dosteyevsky distinguish themselves by the higher frequencies of adverbial forms such as *almost* and *perhaps*, which

Feature	Rank.	Feature	Rank.	Feature	Rank	Feature	Rank
prepratio	1	**pnounratio**	2	**nounratio**	3	almost	4
avgwordlength	5	observed	6	**simplecomplex**	7	**complextotal**	8
simpletotal	9	replied	10	repeated	11	near	12
smell	13	perhaps	14	**avgsent**	15	big	16
cried	17	added	18	sigh	19	rather	20
however	21	dark	22	purpose	23	sighed	24
certain	25	**typetoken**	26	**lexrich**	27	fact	28
few	29	eat	30	certainly	31	slowly	32
moment	33	**cli**	34	black	35	remarked	36
BOL_VBG	37	simply	38	ll	39	contrary	40
idea	41	quite	42	drank	43	CC_NNS	44
FW_NNP	45	NNP_RB	46	ah	47	high	48
ate	49	believe	50	slightly	51	**infoload**	52

Table 5: Mixed feature set ranked using information gain and ten-fold cross validation

Author	**almost**	**near**	**observed**	**perhaps**	**repeated**	**replied**	**smell**
Chekhov	0.000247	**0.000574**	0.000041	0.000289	0.000168	0.000013	**0.000226**
Dosteyevsky	**0.000958**	0.000244	0.000223	**0.001107**	0.000168	0.000042	0.000026
Turgenev	0.000741	0.000497	**0.000508**	0.000437	**0.000592**	**0.000373**	0.000070
Author	**added**	**big**	**cry**	**dark**	**however**	**rather**	**sigh**
Chekhov	0.000091	**0.000569**	**0.000361**	**0.000875**	0.000109	0.000146	**0.000757**
Dosteyevsky	0.000335	0.000131	0.000339	0.000285	0.000374	0.000446	0.000230
Turgenev	**0.000530**	0.000171	0.000198	0.000565	**0.000462**	**0.000538**	0.000483
Author	**certain**	**certainly**	**feat**	**fact**	**few**	**sighed**	**slowly**
Chekhov	0.000225	0.000114	0.003798	0.000482	0.000108	**0.000240**	0.000199
Dosteyevsky	**0.000763**	**0.000323**	0.003169	**0.000909**	0.000203	0.000022	0.000077
Turgenev	0.000576	0.000322	**0.004093**	0.000507	**0.000446**	0.000105	**0.000309**
Author	**I'll**	**black**	**contrary**	**idea**	**moment**	**remarked**	**simply**
Chekhov	0.014557	**0.000450**	0.000035	0.000312	0.000378	0.000001	0.000263
Dosteyevsky	0.013233	0.000170	**0.000205**	**0.000806**	**0.000999**	0.000014	**0.000701**
Turgenev	**0.016007**	0.000351	0.000093	0.000421	0.000355	**0.000140**	0.000342

Table 6: Relative frequencies for distinguishing words by author: Max values in bold

reflect uncertainty, but also adverbial forms such as *certain*, *certainly* and *simply*. They report a high average word length, and both a lower ratio of nouns to total words and lexical richness measure than the other two texts. They are not particular distinguished by their frequencies of verbal usage. The Chekhov translations are distinguishable by higher frequencies of *near* and *smell*, coupled with a lower average sentence length[11] and lower ratios of pronouns and prepositions to total words respectively. The three sentence type metrics are also distinctive. Perhaps the genre of the corpus has an effect here, as all of the included works by Chekhov are short stories while contributions from the other authors are primarily novels and novellas. Temporal variation or development of translatorial style may also play a role in any distinction, Garnett first began translating Turgenev in the late 19th century, followed by Dosteyevsky and Chekhov in the early 20th century, and it is probable that her knowledge of Russian and own writing style in English may have evolved over these years.

Reporting verbs[12] have been examined by Winters (2007), Mikhailov and Villikka (2001) and Baker (2000) in their work on finding distinguishing features of parallel translations of the same text. Here they

[11] Just over fifteen words, compared with over eighteen words for the other two authors.

[12] *Observed*, *repeated*, *replied* can be considered part of this category.

Author	**Chekhov**		**Turgenev**		**Dosteyevsky**	
Attribute	Mean	StdDev	Mean	StdDev	Mean	StdDev
grammlex	0.6553	0.0419	0.6388	0.0518	0.6849	0.0664
infoload	0.4482	0.0176	0.455	0.0237	0.4509	0.027
avgsent	**15.8881**	5.6812	18.6987	5.2877	18.1451	6.6252
nounratio	0.1759	0.0176	0.1723	0.0239	**0.1522**	0.0287
fverbratio	0.0903	0.0083	0.0942	0.0093	0.0943	0.01
pnounratio	**0.1047**	0.017	0.1184	0.0176	0.1278	0.0224
prepratio	**0.0423**	0.0071	0.0336	0.0057	0.0354	0.0068
conjratio	0.0913	0.0116	0.0867	0.011	0.0917	0.0148
numratio	0.0065	0.0025	0.0048	0.0021	0.0065	0.0036
typetoken	0.2954	0.0219	0.3007	0.0297	0.2758	0.031
avgwordlength	12.4996	0.6329	12.4417	0.7341	**13.4928**	1.0065
cli	3.8567	3.3722	5.4318	3.3074	5.0254	4.0703
ari	5.8797	0.9478	6.1201	1.1031	5.9542	1.2986
lexrich	0.2567	0.021	0.2586	0.0271	**0.2372**	0.0303
simplecomplex	**2.0079**	1.3585	1.278	0.526	1.3952	0.6163
dmarkratio	0.0011	0.0008	0.0015	0.0008	0.0012	0.0009
complextotal	**3.0075**	1.3584	2.278	0.526	2.3951	0.6162
simpletotal	**1.7428**	0.5121	1.9679	0.6037	1.9226	0.6496

Table 7: Mean and standard deviation per author: document metrics

occur as distinguishing features of authorial idiolects within works by the same translator. Of course, the efficacy of these features may be increased in these experiments as a result of eliminating noun features, although this was done in an attempt to mitigate the effect of topic based classification of the works of a particular author, and focus on features which represent deeper stylistic patterns. Further analyses of these phenomena must consult the nature of the source text, investigating to what degree of accuracy can the original works of each author be distinguished from one another.

7 Conclusions and Future Directions

This study has demonstrated the efficacy of supervised learning techniques as applied to the task of distinguishing authorial style in a literary corpus translated from Russian to English by a single translator. Both document metrics and n-gram features perform very well for this task, obtaining accuracies of over 80% using feature sets from each category. Combined feature sets improved performance, resulting in 95% classification accuracy between the three authors in question. Highly ranked features included the ratio of nouns to total words, the ratio of pronouns to total words and the ratios of prepositions to total words, also adverbs and reporting verbs such as *almost*, *observed*, *replied* and *repeated* and *near*. These results imply that in this case there is indeed a clear preservation of the individual authorial style by the translator in question, which to some extent refutes the claims of stylistic similarity or sameness across this particular translator's canon.[13], and supports the theory of a *translator's invisibility* as claimed by Venuti (1995). One aspect of the problem not focused on in this study is the relationship between the source and target text, and it is of interest in future work to investigate to what degree the stylistic shifts in translator's style reflect the original source text, or does the translator in fact create their own defined idiolect for a particular author? Further work may investigate how Garnett's style is distinct from another translator, there is evidence of stylistic differences existing between authors, and also between translators, with different features proving discriminating in both cases, as found in studies by Forsyth and Lam (2013) and Lynch (2013).

Future work on this topic will encompass a wider range of translators and languages in order to inves-

[13]Comments by Vladimir Nabokov and others as refered to by Remnick (2005).

tigate more general patterns in translated literature. Results using relatively shallow linguistic features such as POS n-grams and word class distributions have proven themselves useful in distinguishing authorial variation in a translator's style, however it is also of interest to apply deeper linguistic processing to these texts in order to investigate more fine-grained elements of authorial and translatorial style within text. Examples of technologies which could be applied include semantic role labeling, (Swier and Stevenson, 2004) deep syntactic parsing, (Lucic and Blake, 2011), and LDA for detecting levels of metaphor (Heintz et al., 2013), in order to obtain a clearer picture of the stylistic structure of such documents.

Acknowledgements

The Centre for Applied Data Analytics Research is an Enterprise Ireland and IDA initiative. Many thanks to Dr. Daniel Isemann at Universität Leipzig for comments on an early draft of this work and to Prof. Carl Vogel at Trinity College Dublin who provided guidance, inspiration and extensive comments on previous studies in this space.

References

M. Baker. 2000. Towards a methodology for investigating the style of a literary translator. *Target*, 12(2):241–266.

M. Baroni and S. Bernardini. 2006. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259.

L. Borin and K. Pruetz. 2001. Through a glass darkly: Part-of-speech distribution in original and translated text. *Language and Computers*, 37(1):30–44.

J. Burrows. 2002. The Englishing of Juvenal: computational stylistics and translated texts. *Style*, 36(4):677–699.

Meri Coleman and TL Liau. 1975. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60(2):283.

Mariano Felice and Lucia Specia. 2012. Linguistic features for quality estimation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 96–103. Association for Computational Linguistics.

Richard S. Forsyth and Phoenix W. Y. Lam. 2013. Found in translation: To what extent is authorial discriminability preserved by translators? *Literary and Linguistic Computing*.

E. Frank, M.A. Hall, G. Holmes, R. Kirkby, B. Pfahringer, and I.H. Witten. 2005. Weka: A machine learning workbench for data mining. *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers*, pages 1305–1314.

Michael Gamon. 2004. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the 20th international conference on Computational Linguistics*, page 841. Association for Computational Linguistics.

Sunita Goel and Jagdish Gangolly. 2012. Beyond the numbers: Mining the annual reports for hidden cues indicative of financial statement fraud. *Intelligent Systems in Accounting, Finance and Management*, 19(2):75–89.

Ilana Heintz, Ryan Gabbard, Mahesh Srinivasan, David Barner, Donald S Black, Marjorie Freedman, and Ralph Weischedel. 2013. Automatic extraction of linguistic metaphor with lda topic modeling. *Meta4NLP 2013*, page 58.

I. Ilisei and D. Inkpen. 2011. Translationese traits in romanian newspapers: A machine learning approach. *International Journal of Computational Linguistics and Applications*.

I. Ilisei, D. Inkpen, G. Corpas Pastor, and R. Mitkov. 2010. Identification of Translationese: A Machine Learning Approach. *Computational Linguistics and Intelligent Text Processing*, pages 503–511.

Dorothy Kenny. 2001. *Lexis and creativity in translation: a corpus-based study*. St Jerome Pub.

M. Koppel and N. Ordan. 2011. Translationese and its dialects. *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA*.

D. Kurokawa, C. Goutte, and P. Isabelle. 2009. Automatic Detection of Translated Text and its Impact on Machine Translation. In *Proceedings of the XII MT Summit,Ottawa, Ontario, Canada*. AMTA.

G. Lembersky, N. Ordan, and S. Wintner. 2011. Language Models for Machine Translation: Original vs. Translated Texts. *Empirical Methods in Natural Language Processing, Edinburgh, Scotland, 2011*.

D. Li, C. Zhang, and K. Liu. 2011. Translation style and ideology: a corpus-assisted analysis of two english translations of hongloumeng. *Literary and Linguistic Computing*, 26(2):153.

Ana Lucic and Catherine Blake. 2011. Comparing the similarities and differences between two translations. In *Digital Humanities 2011*, page 174. ALLC.

Gerard Lynch and Carl Vogel. 2009. Chasing the ghosts of ibsen: A computational stylistic analysis of drama in translation. In *Digital Humanities 2009: University of Maryland, College Park, MD, USA*, page 192. ALLC/ACH.

Gerard Lynch and Carl Vogel. 2012. Towards the automatic detection of the source language of a literary translation. In Martin Kay and Christian Boitet, editors, *COLING (Posters)*, pages 775–784. Indian Institute of Technology Bombay.

Gerard Lynch. 2013. *Identifying Translation Effects in English Natural Language Text*. Ph.D. thesis, Trinity College Dublin.

F. Mairesse and M. Walker. 2008. Trainable generation of big-five personality styles through data-driven parameter estimation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 165–173.

M. Mikhailov and M. Villikka. 2001. Is there such a thing as a translators style? In *Proceedings of Corpus Linguistics 2001, Lancaster, UK*, pages 378–385.

Charles A Moser. 1988. Translation: The achievement of constance garnett. *The American Scholar*, pages 431–438.

M. Popescu. 2011. Studying translationese at the character level. In *Proceedings of the 8th International Conference on Recent Advances in Natural Language Processing (RANLP'2011). Hissar, Bulgaria*.

David Remnick. 2005. The translation wars. *The New Yorker*, 7:98–109.

Carolyn Rosé, Yi-Chia Wang, Yue Cui, Jaime Arguello, Karsten Stegmann, Armin Weinberger, and Frank Fischer. 2008. Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported collaborative learning. *International journal of computer-supported collaborative learning*, 3(3):237–271.

Jan Rybicki and Magda Heydel. 2013. The stylistics and stylometry of collaborative translation: Woolfs night and day in polish. *Literary and Linguistic Computing*, 28(4):708–717.

J. Rybicki. 2006. Burrowing into Translation: Character Idiolects in Henryk Sienkiewicz's Trilogy and its Two English Translations. *Literary and Linguistic Computing*, 21(1):91–103.

J. Rybicki. 2012. The great mystery of the (almost) invisible translator. *Quantitative Methods in Corpus-Based Translation Studies: A Practical Guide to Descriptive Translation Research*, page 231.

H. Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of international conference on new methods in language processing*, volume 12, pages 44–49. Manchester, UK.

EA Smith and RJ Senter. 1967. Automated readability index. *AMRL-TR. Aerospace Medical Research Laboratories (6570th)*, page 1.

Robert S Swier and Suzanne Stevenson. 2004. Unsupervised semantic role labelling. In *Proceedings of EMNLP*, volume 95, page 102.

Dominik Vajn. 2009. *Two-dimensional theory of style in translations: an investigation into the style of literary translations*. Ph.D. thesis, University of Birmingham.

H. van Halteren. 2008. Source Language Markers in EUROPARL Translations. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 937–944. Coling 2008 Organizing Committee.

L. Venuti. 1995. *The translator's invisibility: A history of translation*. Routledge.

Q. Wang and D. Li. 2012. Looking for translator's fingerprints: a corpus-based study on Chinese translations of Ulysses. *Literary and Linguistic Computing*.

Marion Winters. 2007. F. scott fitzgerald's die schönen und verdammten: A corpus-based study of speech-act report verbs as a feature of translators' style. *Meta: Journal des traducteurs*, 52(3).

Author Verification Using Common N-Gram Profiles of Text Documents

Magdalena Jankowska and **Evangelos Milios** and **Vlado Kešelj**
Faculty of Computer Science, Dalhousie University
6050 University Avenue
Halifax, NS B3H 4R2, Canada
{jankowsk, eem, vlado}@cs.dal.ca

Abstract

Authorship verification is the problem of answering the question whether or not a sample text document was written by a specific person, given a few other documents known to be authored by them. We propose a proximity based method for one-class classification that applies the Common N-Gram (CNG) dissimilarity measure. The CNG dissimilarity (Kešelj et al., 2003) is based on the differences in the frequencies of n-grams of tokens (characters, words) that are most common in the considered documents. Our method utilizes the pairs of most dissimilar documents among documents of known authorship. We evaluate various variants of the method in the setting of a single classifier or an ensemble of classifiers, on a multilingual authorship verification corpus of the PAN 2013 Author Identification evaluation framework. Our method yields competitive results when compared to the results achieved by the participants of the PAN 2013 competition on the entire set, as well as separately on two subsets — English and Spanish ones — out of the three language subsets of the corpus.

1 Introduction

The task of computational detection of who wrote a given text is a widely studied linguistic and machine learning problem with applications in domains such as forensics, security, criminal and civil law, or literary research. The authorship verification problem is a type of such a computational authorship analysis task, in which, given a set of documents written by one author, and a sample document, we are asked whether or not this sample document was written by this given author. This is different from the more traditional problem of deciding who among a finite number of candidate authors for which we are given sample writings, wrote a document in question, and, albeit more difficult, is often considered to better reflect the real-life problems related to authorship detection (Koppel et al., 2012).

We describe our one-class proximity based classification method and evaluate it on the multilingual dataset of the Authorship Identification competition task of PAN 2013 (evaluation lab on uncovering plagiarism, authorship, and social software misuse) (Juola and Stamatatos, 2013).

During the competition, to which a variant of our method has been submitted (Jankowska et al., 2013), it yielded ranking 5th (joint) out of 18 with respect to the accuracy, and 1st rank out of 10 in the secondary ranking based on the area under the ROC curve (AUC), which evaluates the ordering of instances by the confidence score. In this paper we show some further experiments on how a different way of tuning the classifier parameters, using solely the training dataset of the competition, as well as an ensemble of classifiers based on our method, without any parameter tuning, leads to competitive accuracy results while still achieving high AUC values.

2 Related Work

The author analysis has been studied extensively in the context of the authorship attribution problem, in which there is a small set of candidate authors out of which the author of a questioned document is to

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 387–397, Dublin, Ireland, August 23-29 2014.

be selected. There are several papers (Stamatatos, 2009; Juola, 2008; Koppel et al., 2009) presenting excellent surveys of this area.

The two main categories (Stamatatos, 2009) of solutions for the problem are similarity based approaches, in which a classification is performed in a Neigherst Neighbour scheme, attributing a sample text to the author whose writing is most similar according to some measure, and machine-learning based approaches, in which each document by an author is treated as a data sample within a class, and a supervised classifier is trained on these data.

A more limited research has been performed on an open-set variant on this problem, in which it is possible that none of the candidate authors wrote a document in question, with authorship verification being the extreme case of an open-set problem with only one candidate. The "unmasking method" for authorship verification (Koppel and Schler, 2004) is successful for novel-length texts. This approach, similarly as our method, falls into a category of *intrinsic* methods (Juola and Stamatatos, 2013); it uses only the documents in question, without constructing classes of other authors. The ensemble of one-class classifiers (Halvani et al., 2013), which achieved high accuracy at the PAN 2013 Author Identification competition, is also an example of such an intrinsic method. It varies from our approach by using a different scheme of creating the dissimilarity between an unknown document and the known authorship set of texts, based on the Nearest Neighbour technique (Tax, 2001), as well as by a different distance measure and features used.

Another way of approaching the author verification problem is to cast it into a binary or multi-class classification, by creating a class or classes of other authors. The "imposters" method (Koppel and Winter, 2014) generates a very large set of texts by authors that did not write the questioned document, to transform the problem into a open-set author attribution problem with many candidates, handled by an ensemble-based similarity method (Koppel et al., 2011). A modified version of the imposters method (Seidman, 2013) achieved first ranking in the PAN 2013 Authorship Identification competition. The method (Veenman and Li, 2013), which achieved the highest accuracy on the English set in this competition, is also of such an *extrinsic* type; its first step is a careful selection of online documents similar to the ones in the problems. The method (Ghaeini, 2013), which produces competitive ordering of verification instances, uses weighted k-NN approach using classes of other authors created from other verification instances.

3 Methodology

The formulation of the authorship verification task for the Author Identification Task at PAN 2013 is the following: "Given a set of documents (no more than 10, possibly only one) by the same author, is an additional (out-of-set) document also by that author?" (Juola and Stamatatos, 2013).

We approach this task with an algorithm based on the idea of proximity based methods for one-class classification. In one-class classification framework, an object is classified as belonging or not belonging to a target class, while only sample examples of objects from the target class are available during the training phase. Our method resembles the idea of the k-centers algorithm for one-class classification (Ypma et al., 1998; Tax, 2001), with k being equal to the number of all training documents in the target set (i.e., written by the given author). The k-centers algorithm is suitable for cases when there are many data points from the target class; it uses equal radius sphere boundaries around the target data points and compares the sample document to the closest such centre. We propose a different classification condition, described below, utilizing the pairs of most dissimilar documents within the set of known documents.

Let $A = \{d_1, ..., d_k\}$, $k \geq 2$, be the input set of documents written by a given author, which we will call *known documents*. If only one known document is provided, we split it in half and treat these two chunks as two known documents. Let u be the input sample document, of which the authorship we are to verify, that is return the answer "Yes" or "No" to the posed question whether it was written by the given author.

Our algorithm calculates for each known document $d_i, i = 1, 2, ..., k$, the maximum dissimilarity between this document and all other known documents: $D^{max}(d_i, A)$, as well as the dissimilar-

ity between this document and the sample document u: $D(d_i, u)$, and finally the dissimilarity ratio $r(d_i, u, A) = \frac{D(d_i,u)}{D^{max}(d_i,A)}$ (and thus $r(d_i, u, A) < 1$ means that there exists a known document more dissimilar to d_i than u, while $r(d_i, u, A) > 1$ means that all the known documents are more similar to d_i than u). The average $M(u, A)$ of the dissimilarity ratio over all known documents $d_1, d_2, ..., d_k$ from A, is the subject of the thresholding: the sample u is classified as written by the same person as the known documents if and only if $M(u, A)$ is at most equal to a selected threshold θ. Notice that in this framework the dissimilarity between the documents does not need to be a metric distance, i.e., it does not need to fulfil the triangle inequality (as is the case for the dissimilarity measure we choose).

For the dissimilarity measure between documents we use the Common N-Gram (CNG) dissimilarity; proposed by Kešelj et al. (2003); this dissimilarity (or its variants) used in the Nearest Neighbour classification scheme (Common N-gram classifier) was successfully applied to authorship classification tasks (Kešelj et al., 2003; Juola, 2008; Stamatatos, 2007). The CNG dissimilarity is based on the differences in the usage frequencies of the most common n-grams of tokens (usually characters, but possibly other tokens) of the documents. Each document is represented by a *profile*: a sequence of the most common character n-grams (strings of characters of the given length n from the document) coupled with their frequencies (normalized by the length of the document). The dissimilarity between two documents of the profiles P_1 and P_2 is defined as follows:

$$D(P_1, P_2) = \sum_{x \in (P_1 \cup P_2)} \left(\frac{f_{P_1}(x) - f_{P_2}(x)}{\frac{f_{P_1}(x) + f_{P_2}(x)}{2}} \right)^2 \quad (1)$$

where x is a character n-gram from the union of two profiles, and $f_{P_i}(x)$ is the normalized frequency of the n-gram x in the the profile P_i, $i = 1, 2$ ($f_{P_i}(x) = 0$ whenever x does not appear in the profile P_i). The parameters of the dissimilarity are the length of the n-grams n and the length of the profile L. As our method is based on the ratios of dissimilarities between documents, we take care that the documents in a given problem are always represented by profiles of the same length. We experiment with two ways of selecting the length of the profiles. In the dynamic-length variant, the length of profiles is selected separately for each problem, based on the number of n-grams in the documents in the given instance (parametrized as a fraction f of all n-grams of the document that contains the least number of them). In the fixed-length variant, we use a selected fixed length L of profiles. For a one-class classifier we need to select two parameters defining the features used for dissimilarity (length of the n-grams n, and either the fixed length L of a profile, or the fraction f defining the profile length), and the parameter θ (for classifying by thresholding the average dissimilarity ratio M).

We linearly scale the measure M to represent it as a confidence score in the range from 0 (the highest confidence in the answer "No") to 1 (the highest confidence in the answer "Yes"), with the answer "Yes" given if and only if the confidence score is at least 0.5. The value of M equal to θ is transformed to the score 0.5, values greater than θ to the scores between 0 and 0.5, and values less than θ to the scores between 0.5 and 1 (a cutoff of 0.1 is applied, , i.e. all values of $M(u, A) < \theta - \textit{cutoff}$ are mapped to the score 1, and all values of $M(u, A) > \theta + \textit{cutoff}$ are mapped to the score 0).

4 Training and test datasets

We leverage the evaluation framework of the PAN 2013 competition task of Author Identification (Juola and Stamatatos, 2013), the datasets of which were carefully created for authorship verification, with effort made to match within each problem instance the texts by the same genre, register, theme and time of writing. The dataset consists of English, Greek and Spanish subsets. In each instance, the number of documents of known authorship is not greater than 10 (possibly only one). The dataset is divided into the training set `pan13-ai-train` and the test set `pan13-ai-test`. The training set was made available for the participants before the competition; the test set was used to evaluate the submissions and subsequently published (PAN, 2013).

To enrich the training dataset for our competition submission, we also compiled ourselves two additional datasets using existing sets for other authorship identification tasks. `mod-pan12-aa-EN` is

an English author verification set compiled from the fiction corpus for the Traditional Authorship Attribution sub task of the PAN 2012 competition (PAN, 2012; Juola, 2012). `mod-Bpc-GR` is a Greek author verification set compiled from the Greek dataset of journal articles (Stamatatos et al., 2000). It is important to note that these sets are different from the competition dataset in that we did not attempt to match the theme or time of writing of the texts.

Table 1 presents characteristics of the datasets.

	`pan13-ai-train`			
	total	English	Spanish	Greek
number of problems	35	10	5	20
mean of the known document number per problem	4.4	3.2	2.4	5.5
mean length of documents in words	1226	1038	653	1362
genre		textbooks	editorials, fiction	articles
	`pan13-ai-test`			
	total	English	Spanish	Greek
number of problems	85	30	25	30
mean of the known document number per problem	4.1	4.2	3.0	4.9
mean length of documents in words	1163	1043	890	1423
genre		textbooks	editorials, fiction	articles
	`mod-pan12-aa-EN` total: English			
number of problems	22			
mean of the known document number per problem	2.0			
mean length of documents in words	4799			
genre	fiction			
	`mod-Bpc-GR` total: Greek			
number of problems	76			
mean of the known document number per problem	2.5			
mean length of documents in words	1120			
genre	articles			

Table 1: Characteristics of datasets used in our authorship verification experiments.

5 Evaluation measures

In our experiments we use two measures of evaluation, based on the measures proposed for the PAN 2013 competition. The accuracy is the fraction of all problems that have been answered correctly. The AUC measure is the area under the ROC curve based on the confidence scores. It is the nature of applications of authorship verification, such as forensics, that makes the confidence score and not only the binary answer, an important aspect of a solution (Gollub et al., 2013).

For our method accuracy is equivalent to the measure that was used in the competition for the main evaluation. This measure is F_1, defined based on the fact that in the competition it was allowed to withdraw an answer (i.e., use an "I do not know" option). Precision and recall were defined as follows: $recall = \frac{\#correct\_answers}{\#problems}$, $precision = \frac{\#correct\_answers}{\#answers}$, and F_1 is the harmonic mean of precision and recall. For any method that, as our method, provides the answer "Yes" or "No" for all problem instances, the accuracy and F_1 are equivalent.

6 Types of classifiers

A single classifier of our method requires two parameters defining the features to be used to represent a document (the length of an n-gram and the length of a profile), as well as a selection of the threshold for the dissimilarity for the classification decision. We tune and evaluate four version of such single classifiers. Combining many such one-class classifiers, each using different combination of features defining parameters, into one ensemble, allows to remove or mitigate the parameter tuning. We describe the creation and the evaluation of four types of ensembles.

Table 2 reports the considered space for feature defining parameters. On a training set, for a given combination of feature defining parameters (n,L) or (n,f), we use the accuracy at the optimal threshold (a threshold θ that maximizes the accuracy), as a measure of performance for these parameters.

Parameters	
n	length of n-grams
L	# of n-grams: profile length (fixed-length)
f	fraction of n-grams for profile length (dynamic-length)
θ	threshold for classification
θ_{2+}	threshold for classification if at least 2 known documents are given
θ_1	threshold for classification if only one known document is given

Space of considered parameters	
n for character n-grams	$\{3, 4, ..., 9, 10\}$
n for word n-grams	$\{1, 2, 3\}$
L	$\{200, 500, 1000, 1500, 2000, 2500, 3000\}$
f	$\{0.2, 0.3, ..., 0.9, 1\}$

single classifiers		English	Spanish	Greek
vD1	n	6	7	10
	f	0.75		
	θ	1.02	1.005	1.002
vF1	n	6		7
	L	2000		2000
	θ_{2+}	1.02		1.008
	θ_1	1.06		1.04
vF2	n	7	3	9
	L	3000	2000	3000
	θ_{2+}	1.014	1.014	0.997
	θ_1	1.056	1.126	1.060
vD2	n	7	3	9
	f	0.8	0.6	0.8
	θ_{2+}	1.013	1.00530207	0.9966
	θ_1	1.053	1.089	1.059

ensembles		English	Spanish	Greek
eC	type	character		
	(n, L)	all in the considered space		
	θ	1		
eW	type	word		
	(n, L)	all in the considered space		
	θ	1		
eCW	type	character, word		
	(n, L)	all in the considered space		
	θ	1		
eCW	type	character, word		
	(n, L)	selected based on training data		
		(61)	(75)	(43)
	θ	1		

Table 2: Parameters for four variants of single one-class classifiers and four ensembles of one-class classifiers based on our method.

6.1 Single classifiers

For single character n-gram classifiers, we tuned the parameters for each language separately on training data, by selecting feature defining parameters based on their performance, and selecting the thresholds

to correspond to the optimal thresholds. Table 2 reports the parameters of four variants of single classifiers. We include our two submissions to the PAN 2013 Authorship Identification competition: the final submission `vF1` and the preliminary submission `vD1`. The other two classifiers were tuned and tested after the competition.

Our preliminary submission `vD1` (Table 2) is tuned on `pan13-ai-train`, with f chosen ad-hoc. This is the only classifier among the reported variants that does not use a preprocessing of truncation of all documents in a given problem instance to the length of the shortest document, which tend to increase the accuracy for cases of a significant difference in the length of documents.

For tuning of parameters of the final submission `vF1` (Table 2) we use not only `pan13-ai-train`, but also additional training sets `mod-pan12-aa-EN` and `mod-Bpc-GR`. We also introduce two threshold values: one for cases when there are at least two known documents, and another one for the cases when there is only one known document (which has to be divided in two). The intuition behind this double threshold approach is that when there is only one known document, the two halves of it can be more similar to each other than in other cases. After the parameters are selected based on subsets of training sets with only these problems that contain at least two known documents, the additional threshold is selected based on the optimal threshold on a modified "1-only" training set, from the problem of which all known documents except of a random single one is removed. For Spanish, with only three training instances with more than one known document, we use the same parameters as for English.

For tuning of `vF2` and `vD2` (Table 2) we use only competition training data, without the additional corpora used for `vF1`. Feature parameters are selected based on the performance on the subsets containing at least two known documents, and on the "1-only" modified sets (which allows us to use the Spanish training set for tuning the Spanish classifiers).

6.2 Ensembles of classifiers

We test ensembles of single one-class classifiers based on our method, with the ensemble combining answers of the classifiers, and each classifier using different set of features. An important advantage of an ensemble is the alleviation of the problem of tuning the parameters. Each classifier uses a different combination of parameters n and L defining the features. And as many classifiers are used, instead of tuning the threshold of a single classifier based on some training data, the threshold of each classifier is set to some fixed value, with 1 being a natural choice, as it corresponds to checking whether or not the unknown document is (on average) less similar to each given known document than the author's document that is most dissimilar to this given known document.

We test majority voting and voting weighted by the confidence scores of single classifiers. For each ensemble we combine answers of the classifiers in order to obtain the confidence score of the ensemble. For majority voting the confidence score of the ensemble is the ratio of the number of classifiers that output "Yes" to the total number of classifiers, the confidence score of the weighted voting is the average of the confidence scores of the single classifiers.

We experiment with n-grams being characters (utf8-encoded) and words (converted to uppercase). Table 2 summarize the ensembles. The ensemble `eC` is of all character n-gram classifiers in our space of considered parameters n and L; `eW` is of all word n-gram classifiers; `eCW` is of all classifiers of `eC` and `eW`. These ensembles do not use any training data. We also create a classifier `eCW_sel` (Table 2), which is a subset of the classifiers of `eCW`, selected based on the performance of the single classifiers on the training data of the competition. For each language separately, we remove classifiers that on the training data achieved lowest accuracies at their respective optimal thresholds, while keeping at least half of the character based classifiers and at least half of the word based classifiers. (For Spanish, `eCW_sel` and `eCW` differ just by one classifier: the only one that on the small Spanish training set has the optimal accuracy less than 1.)

7 Results

The accuracy and the area under the ROC curve (AUC) values achieved by the variants of our method on the PAN 2013 Author Identification test dataset are presented in Table 3. The table states also the

best PAN 2013 competition results of other participants[1] (that is the results of these participants that achieved the highest accuracy or AUC on any (sub)set). There were 17 other participants for which there are accuracy (or F_1) results, 9 of which submitted also confidence scores evaluated by AUC.

		PAN 2013 Author Identification test dataset							
		F_1 = accuracy except for Ghaeini,2013				AUC			
		all	English	Spanish	Greek	all	English	Spanish	Greek
single classifiers									
vD1		0.718	0.733	0.760	0.667	0.790	0.837	0.846	0.718
vF1		0.682	0.733	0.720	0.600	0.793	0.839	0.859	0.711
vD2		0.729	0.767	0.760	0.667	0.805	0.850	**0.936**	0.704
vF2		0.753	0.767	**0.880**	0.633	**0.810**	0.844	0.885	0.664
ensembles of classifiers									
eC	majority	0.729	0.800	0.840	0.567	0.754	0.777	0.833	0.620
	weight	0.729	**0.833**	0.800	0.567	0.764	0.830	0.859	0.582
eW	majority	0.718	0.733	0.720	0.700	0.763	0.830	0.805	0.700
	weight	0.741	0.767	0.760	0.700	0.822	**0.886**	0.853	0.782
eCW	majority	**0.800**	**0.833**	0.840	0.733	0.755	0.817	0.821	0.633
	weight	0.741	0.800	0.840	0.600	0.780	0.842	0.853	0.622
eCW_sel	majority	**0.800**	**0.833**	0.840	0.733	0.778	0.826	0.814	0.682
	weight	0.788	0.800	0.840	0.733	0.805	0.857	0.853	0.687
boxed values: best competition results of other PAN 2013 Author Identification participants									
Seidman,2013		0.753	0.800	0.600	**0.833**	0.735	0.792	0.583	**0.824**
Veenman and Li,2013		–	0.800	–	–	–	–	–	–
Halvani et al.,2013		0.718	0.700	0.840	0.633	–	–	–	–
Ghaeini,2013		0.606	0.691	0.667	0.461	0.729	0.837	0.926	0.527

Table 3: Area under the ROC curve (AUC) and F_1 (which is equal to accuracy for all algorithms except for (Ghaeini, 2013)) on the test dataset of PAN 2013 Author Identification competition task. Results of variants of our method compared with competition results of those among other competition participants that achieved the highest value of any evaluation measure on any (sub)set. The highest result in any category is bold; the highest result by other competition participants in any category is boxed.

All variants of our method perform better on the English and Spanish subset than on the Greek one, both in terms of the accuracy and in terms of AUC. On the Greek subset they are all outperformed by other competition participant(s). This is most likely due to the fact that the Greek subset was created in a way that makes it especially difficult for algorithms that are based on CNG character-based dissimilarity (Juola and Stamatatos, 2013), by using a variant of CNG dissimilarity for the character 3-grams in order to select difficult cases. This particularity of the set may also be the reason why the ensemble eC of character n-gram classifiers performed worse than other methods on this set.

The variants of our method are competitive in terms of the ordering of the verification instances according to the confidence score as measured by AUC. During the competition, our final submission vF1 achieved the first ranking according to the AUC on the entire set, the highest AUC on the English subset, and the second-highest AUC values on the Spanish and Greek subset, out of 10 participants that submit-

[1]The results of our methods are on the published competition dataset. The results by other participants are the published competition results. The actual competition evaluation set for Spanish may have some text in a different encoding then the published set; our final submission method vF1 yielded on it a different result than on the published dataset.

ted confidence scores. All variants of our method perform better than any other competition participant on the entire set. On the English subset the single classifiers and the ensembles with weighted voting have AUC above 0.8, and out of those only eC has AUC lower than the best result by other participants. On the Spanish subset all variants of our method achieved AUC above 0.8, with vD2 achieving AUC higher than the best competition result on this subset.

In terms of overall accuracy on the entire set, the ensembles combining character and word based classifiers: eCW with majority voting and eCW_sel with both types of voting, achieve accuracy higher then the best overall accuracy in the competition. They also match or surpass the best competition accuracy on the English subset, and match the best competition accuracy on the Spanish subset. The highest accuracy on the English subset was achieved by eC with weighted voting, eCW with majority voting, and eCW_sel with majority voting (higher than the best competition result). vF2 yields on the Spanish subset accuracy higher than the best competition result.

For the ensembles of classifiers, on the English and Spanish subsets, the AUC for voting weighted by the confidence scores are higher than the AUC for the majority voting, but not so on the Greek subset. This is consistent with the fact that on the Greek subset the confidence scores for single classifier variants yield worse ordering (AUC) than on other sets. Creation of eCW_sel by removing from the ensemble eCW the classifiers that perform worst on the training data improves the Greek results, and slightly the English results.

We tested the statistical significance of accuracy differences between all pairs of accuracies reported in Table 3 by the exact binomial McNemar's test (Dietterich, 1998). Only few of these differences are statistically significant. On the entire set these are: the difference between the accuracy of eCW with majority voting and of eC with majority voting, vD1 and vF1, as well as the difference between the accuracies of eCW_sel with weighted voting and of vF1. On the Greek subset, this is the difference between the accuracies of the submission (Seidman, 2013) and the lower accuracy of eC with weighted voting.

		English mod-pan12-aa-EN		Greek mod-Bpc-GR	
		accuracy	AUC	accuracy	AUC
vD1		0.545	0.649	0.605	0.661
vD2		0.727	0.826	0.566	0.698
vF2		**0.773**	**0.843**	0.618	0.709
eC	majority	0.636	**0.843**	0.658	0.694
	weighted	0.682	0.806	0.671	0.703
eW	majority	0.636	0.674	**0.750**	**0.757**
	weighted	0.727	0.736	0.737	0.749
eCW	majority	0.636	0.785	0.737	0.725
	weighted	0.682	0.818	0.711	0.719
eCW_sel	majority	0.636	0.789	**0.750**	0.742
	weighted	0.682	0.826	0.737	0.737

Table 4: Accuracy and area under ROC curve (AUC) of our method on other English and Greek datasets. The sets were compiled by ourselves for the purpose of enriching training domain for other variant of our classifier. The highest result in any category is bold.

The datasets mod-pan12-aa-EN and mod-Bpc-GR were compiled by ourselves from other authorship attribution sets for the purpose of enriching the training corpora for our final submission vF1. The comparison between results on the English and Greek subsets of vF1 with the results of vF2 (for which these additional sets were not used), shows that vF2 achieved better results on English data. while vF1 has higher AUC on Greek data.

Though these additional sets were not created specifically for authorship verification evaluation, we

examine the results of our methods on these sets (with the exception of vF1, which is tuned on them). We present the results in Table 4. vD1 performs poorly on mod-pan12-aa-EN. This is in part due to the fact that in this set the documents in a given problem instance can differ significantly with respect to the length, and the variant vD1 does not use the preprocessing of truncation all files withing a problem to the same length. The variants vD2 and vF2 (which apply this truncation) yielded accuracy and AUC similar in value to the ones achieved on the PAN 2013 English subset. The ensembles containing character n-gram classifiers yielded similar AUC on mod-pan12-aa-EN as on the PAN2013 English subset, close in value to 0.8. But their accuracies are distinctly lower than the results on the English competition subset, with values below 0.7 (for each such an ensemble, vast majority of the misclassified instances are false negatives: cases classified as not written by the same person when in fact they are). For mod-Bpc-GR the single classifiers (with parameters tuned on the competition Greek subset) perform rather poorly, with results similar but lower in values than the results yielded on the competition Greek test set. The ensembles containing word n-gram based classifiers perform better than the ensembles containing only the character n-gram classifiers, yielding both AUC and accuracy in the range of 0.71 – 0.75.

8 Future Work

It will be of interest to investigate the relation between the performance of our method and the number and the length of the considered texts. An interesting direction indicated by results of our experiments is also the analysis of the role of word n-grams and character n-grams for authorship verification depending on the genre of the texts, and on the topical similarity between the documents.

9 Conclusions

We present our proximity based one-class classification method for authorship verification. The method uses for each document of known authorship the most dissimilar document of the same author, and examines how much more or less similar is the questioned document. We use Common N-Gram dissimilarity based on differences in frequencies of character and word n-grams.

We evaluate our method on the set of PAN 2013 Authorship Identification competition. One variant of our method was submitted to the competition. The ordering by scores indicating the confidence that the documents were written by the same person, yielded by our method, and evaluated by area under ROC curve (AUC), is competitive with respect to other participants of the competition, overall, and on the English and Spanish subsets. On the entire set, AUC by each variant of our method is higher than the best result by other participants. In terms of accuracy, the method also performs better on the English and Spanish subsets of the dataset, and worse on the Greek one. An ensemble combining character based classifiers and word based classifiers yields the best accuracy, surpassing the best competition result on the entire set and on the English subset, while matching the best competition result on the Spanish subset.

As all proximity based one-class classification algorithms, our method relies on a selected threshold on the proximity between the questioned text and the set of documents of known authorship. Additionally, a single classifier requires two parameters defining the features representing documents. Ensembles of classifiers allow to alleviate the parameter tuning, by using many classifiers for many combinations of feature defining parameters, with a threshold fixed to 1 (a natural, albeit arbitrary, value).

Acknowledgements

This research was funded by a contract from the Boeing Company, Killam Predoctoral Scholarship, and a Collaborative Research and Development grant from the Natural Sciences and Engineering Research Council of Canada.

References

Thomas G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10:1895–1923.

M.R. Ghaeini. 2013. Intrinsic Author Identification Using Modified Weighted KNN - Notebook for PAN at CLEF 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, September.

Tim Gollub, Martin Potthast, Anna Beyer, Matthias Busse, Francisco M. Rangel Pardo, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2013. Recent trends in digital text forensics and its evaluation - plagiarism detection, author identification, and author profiling. In Pamela Forner, Henning Müller, Roberto Paredes, Paolo Rosso, and Benno Stein, editors, *CLEF*, volume 8138 of *Lecture Notes in Computer Science*, pages 282–302. Springer.

Oren Halvani, Martin Steinebach, and Ralf Zimmermann. 2013. Authorship Verification via k-Nearest Neighbor Estimation - Notebook for PAN at CLEF 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, September.

Magdalena Jankowska, Vlado Kešelj, and Evangelos Milios. 2013. Proximity Based One-class Classification with Common N-Gram Dissimilarity for Authorship Verification Task - Notebook for PAN at CLEF 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, September.

Patrick Juola and Efstathios Stamatatos. 2013. Overview of the Author Identification Task at PAN 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, September.

Patrick Juola. 2008. Authorship attribution. *Foundations and Trends® in Information Retrieval*, 1(3):233–334.

Patrick Juola. 2012. An overview of the traditional authorship attribution subtask. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF (Online Working Notes/Labs/Workshop)*.

Vlado Kešelj, Fuchun Peng, Nick Cercone, and Calvin Thomas. 2003. N-gram-based author profiles for authorship attribution. In *Proceedings of the Conference Pacific Association for Computational Linguistics, PACLING'03*, pages 255–264, Dalhousie University, Halifax, Nova Scotia, Canada, August.

Moshe Koppel and Jonathan Schler. 2004. Authorship verification as a one-class classification problem. In *Proceedings of the 21st International Conference on Machine Learning*, ICML '04, page 489–495, Banf, Alberta, Canada, July. ACM.

Moshe Koppel and Yaron Winter. 2014. Determining if two documents are written by the same author. *Journal of the Association for Information Science and Technology*, 65(1):178–187.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2009. Computational methods in authorship attribution. *Journal of the American Society for Information Science and Technology*, 60(1):9–26.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2011. Authorship attribution in the wild. *Language Resources and Evaluation*, 45(1):83–94, March.

Moshe Koppel, Jonathan Schler, Shlomo Argamon, and Yaron Winter. 2012. The "Fundamental Problem" of Authorship Attribution. *English Studies*, 93(3):284–291.

PAN. 2012. Dataset of PAN 2012, Author Identification task. `http://www.uni-weimar.de/medien/webis/research/events/pan-12/pan12-web/authorship.html`. Accessed on Apr 2, 2013.

PAN. 2013. Dataset of PAN 2013, Author Identification task. `http://www.uni-weimar.de/medien/webis/research/events/pan-13/pan13-web/author-identification.html`. Accessed on Oct 8, 2013.

Shachar Seidman. 2013. Authorship Verification Using the Impostors Method - Notebook for PAN at CLEF 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, September.

Efstathios Stamatatos, George Kokkinakis, and Nikos Fakotakis. 2000. Automatic text categorization in terms of genre and author. *Computational Linguistics*, 26(4):471–495, December.

Efstathios Stamatatos. 2007. Author identification using imbalanced and limited training texts. In *Proceeding of the 18th International Workshop on Database and Expert Systems Applications, DEXA'07*, pages 237–241, Regensburg, Germany, September.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.

David Tax. 2001. *One Class Classification. Concept-learning in the absence of counter-examples.* Ph.D. thesis, Delft University of Technology, June.

Cor J. Veenman and Zhenshi Li. 2013. Authorship Verification with Compression Features. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, September.

Alexander Ypma, Er Ypma, and Robert P.W. Duin. 1998. Support objects for domain approximation. In *Proceedings of International Conference on Artificial Neural Networks*, pages 2–4, Skovde, Sweden, September. Springer.

Dynamically Integrating Cross-Domain Translation Memory into Phrase-Based Machine Translation during Decoding

Kun Wang[†] **Chengqing Zong**[†] **Keh-Yih Su**[‡]
[†]National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
[‡]Institute of Information Science, Academia Sinica, Taiwan
[†]{kunwang, cqzong}@nlpr.ia.ac.cn
[‡]kysu@iis.sinica.edu.tw

Abstract

Our previous work focuses on combining translation memory (TM) and statistical machine translation (SMT) when the TM database and the SMT training set are the same. However, the TM database will deviate from the SMT training set in the real task when time goes by. In this work, we concentrate on the task when the TM database and the SMT training set are different and even from different domains. Firstly, we dynamically merge the matched TM phrase-pairs into the SMT phrase table to meet the real application. Secondly, we propose an improved integrated model to distinguish the original and the newly-added phrase-pairs. Thirdly, a simple but effective TM adaptation method is adopted to favor the consistent translations in cross-domain test. Our experiments have shown that merging the TM phrase-pairs achieves significant improvements. Furthermore, the proposed approaches are significantly better than the TM, the SMT and previous integration works for both in-domain and cross-domain tests.

1 Introduction

Since the translation memory (TM) system and the statistical machine translation (SMT) system complement each other in those matched sub-segments and unmatched sub-segments (Wang et al., 2013), combining them can improve the output quality significantly, especially when high-similarity fuzzy matches are available. Therefore, combining TM and SMT is drawing more and more attention in recent years (He et al., 2010a; 2010b; 2011; Koehn and Senellart, 2010; Zhechev and van Genabith, 2010; Ma et al., 2011; Dara et al., 2013; Wang et al., 2013).

Those previous works on combining TM and SMT can be classified into four categories: (1) selecting the better translation sentence from TM and SMT (He et al., 2010a; 2010b; Dara et al., 2013); (2) incorporating TM matched sub-segments into SMT in a pipelined manner (Koehn and Senellart, 2010; He et al., 2011; Ma et al., 2011); (3) only enhancing the SMT phrase table with new TM phrase-pairs (Bi çici and Dymetman, 2008; Simard and Isabelle, 2009); and (4) incorporating the associated TM information with each source phrase to guide the SMT decoding (Wang et al., 2013).

However, all previous works mentioned above only focus on the case in which the TM database and the SMT training set share the same data-set. Nonetheless, in real applications, the TM database will deviate from the SMT training set when time goes by, because the TM database will be dynamically enlarged when more translations are generated by the human translator. Therefore, this paper will concentrate on a more realistic case, in which the TM database and the SMT training set are different and even from different domains.

When the TM database and the SMT training set share the same data-set, the integrated model (Wang et al., 2013) can avoid the drawbacks of the pipeline approaches and outperforms the other approaches significantly. However, this integrated model only refers to the TM information but not adopts the matched TM phrase-pairs as candidates during decoding. Therefore, many TM phrase-pairs cannot be covered by the SMT phrase table when the TM database and the SMT training set are dif-

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 398–408, Dublin, Ireland, August 23-29 2014.

ferent. It is thus impossible to generate those unseen TM target phrases. This problem would even get worse when the TM database and the SMT training set are from different domains.

To make the integrated model meet the real application, we dynamically merge the matched TM phrase-pairs into the SMT phrase table. In addition, an improved integrated model is proposed to distinguish the original SMT phrase-pairs and the newly-added ones extracted from TM. Furthermore, a simple but effective TM adaptation method is adopted to favor the consistent translation in cross-domain test. To our best knowledge, this is the first unified framework for integrating TM into SMT during decoding when the TM database and the SMT training set are different (even from different domains).

On the TM database which consists of Chinese–English computer technical documents, our experiments have shown that merging the matched TM phrase-pairs achieves significant improvement when the fuzzy match score is above 0.5. Besides, the proposed approaches are significantly better than either the SMT or the TM systems for both the in-domain and the cross-domain tests when the fuzzy match score is above 0.4. Furthermore, the proposed approaches also outperform previous integration works significantly in all test conditions.

2 Integrated Model

Wang et al. (2013) incorporated the TM information into the phrase-based SMT, and re-defined the translation problem as:

$$\hat{t} = \arg\max_t P(t|s, tm\_s, tm\_t, tm\_f, s\_a, tm\_a)$$

Where s denotes the given source sentence, t is a corresponding target translation, and $\hat{t}$ is the final result; $[tm\_s, tm\_t, tm\_f, s\_a, tm\_a]$ is the associated information of the best TM sentence-pairs; tm_s and tm_t are the corresponding TM source and target sentences, respectively; tm_f denotes its corresponding fuzzy match score (from 0 to 1); s_a is the monolingual alignment information between s and tm_s; and tm_a denotes the bilingual word alignment information between tm_s and tm_t.

With the TM information, this problem can be simplified to:

$$\hat{t} \triangleq \text{argmax}\left\{P\left(\bar{t}_1^{\text{K}}\middle|\bar{s}_{a(1)}^{a(\text{K})}\right) \times \prod_{k=1}^{K} \max_{tm\_\bar{t}_{a(k)}} P(M_k|L_k, z)\right\} \quad (1)$$

Where $\bar{s}_{a(k)}$ and $\bar{t}_k$ denote the k-th associated source and target phrases, respectively; $tm\_\bar{s}_{a(k)}$ and $tm\_\bar{t}_{a(k)}$ are the corresponding TM source and target phrases associated with the given source phrase $\bar{s}_{a(k)}$ (total K phrases without insertion). M_k is the corresponding TM target phrase matching status for the current target candidate $\bar{t}_k$, which reflects the quality of the given candidate; L_k is the linking status vector of $\bar{s}_{a(k)}$ (the aligned source phrase, within $\bar{s}_{a(1)}^{a(\text{K})}$, of $\bar{t}_k$), which indicates the matching and linking status in the source side (and is closely related to the matching status of the target side). tm_f is uniformly divided into ten fuzzy match intervals and the index z specifies the corresponding interval.

In Equation (1), the first factor is just the typical phrase-based SMT model, and the second factor $P(M_k|L_k, z)$ is the information derived from the TM sentence pair. Afterwards, the factor $P(M_k|L_k, z)$ was further derived with TM matching status as follows:

$$P(M_k|L_k, z) \approx \begin{Bmatrix} \text{P}(TCM_k|SCM_k, NLN_k, LTC_k, SPL_k, SEP_k, z) \\ \times \text{P}(LTC_k|CSS_k, SCM_k, NLN_k, SEP_k, z) \\ \times \text{P}(CPM_k|TCM_k, SCM_k, NLN_k, z) \end{Bmatrix} \quad (2)$$

Where the first factor reflects the TM content matching status, the second factor is the relationship between various TM target phrases, and the third factor is the reordering information implied by TM. Equation (2) is adopted to guide the SMT decoding, and is denoted as the integrated Model-III in (Wang et al., 2013) (also called **Model-III** in this paper thereafter).

For space limitation, only those features which are also adopted in our additional introduced probability factor (to be specified later) will be briefly introduced here:

Target Phrase Content Matching Status (TCM): It indicates the content matching status between $\bar{t}_k$ and $tm\_\bar{t}_{a(k)}$, and reflects the quality of $\bar{t}_k$. It is a member of {*Same*, *High*, *Low*, *NA* (*Not-Applicable*)}.

Source Phrase Content Matching Status (SCM): It indicates the content matching status between $\bar{s}_{a(k)}$ and $tm\_\bar{s}_{a(k)}$, and affects the matching status of $\bar{t}_k$ and $tm\_\bar{t}_{a(k)}$ greatly. It is a member of {*Same*, *High*, *Low*, *NA*}.

Number of Linking Neighbors (NLN): Usually, the context of a source phrase would affect its target translation. The more similar the context is, the more likely that the translation is the same. NLN is adopted to measure the context similarity.

3 Proposed Approaches

3.1 Merging the TM Phrase-Pairs

Since all TM phrase-pairs are only referred while re-scoring the SMT candidates in Model-III, they are not regarded as candidates during decoding. When the TM database and the SMT training set are the same, this restriction is reasonable because the SMT phrase table can cover all the continuous TM phrase pairs within the phrase length limit. However, this would not be true when the TM database and the SMT training set are different. Therefore, the SMT phrase table should be further enhanced with those matched new TM phrase pairs in this case.

According to their relations with the SMT phrase table, TM phrase pairs can be classified into three different categories: (1) the whole TM phrase-pair can be found in the original SMT phrase table; (2) only TM source phrase exists in the original SMT phrase table, but its corresponding target phrase does not; (3) even TM source phrase cannot be found in the original SMT phrase table. Since the first category has been covered by the original SMT phrase table, only the phrase-pairs from the second and the third categories should be added into the SMT phrase table dynamically for each input sentence. To distinguish those newly added phrase-pairs from the original SMT phrase-pairs, we use eight additional feature weights λ_m for the translation probability (lexical and phrase transfer in both directions) and two more feature weights for the phrase penalty (details will be specified later in Section 4).

The above approach is inspired by the work of (Bi ɕci and Dymetman, 2008). However, there are three differences between our approach and theirs. Firstly, we add all those matched TM phrase-pairs (include all associated sub-phrase pairs), while Bi ɕci and Dymetman (2008) only added the longest matched one; Secondly, we add all the possible TM target phrase-pairs for a given TM source phrase while they extracted only one TM target phrase regardless of the existence of multiple TM target candidates; Lastly, we use different feature weights to distinguish those newly added TM phrase-pairs from the original SMT phrase-pairs, while they treated them equally.

3.2 Distinguishing the TM Phrase-Pairs

As mentioned in Section 3.1, we need to merge those TM matched phrase pairs into the SMT phrase table when the TM database and the SMT training set are different. However, the original integrated Model-III does not distinguish the newly added TM phrase-pairs from those original SMT phrase-pairs in $P(M_k|L_k,z)$. Therefore, we introduce two new features ***Source Phrase Origin*** (**SPO**) and ***Target Phrase Origin*** (**TPO**), which are a member of {*Original*, *Newly-Added*}, to the original Model-III in (Wang et al., 2013) to favor the newly added TM phrase-pairs, and re-derive $P(M_k|L_k,z)$ as follows (assume that TPO is only dependent on SPO, NLN and z):

$$\begin{aligned} & P(M_k|L_k,z) \\ & \triangleq P([TCM,LTC,CPM,TPO]_k|[SCM,NLN,CSS,SPL,SEP,SPO]_k,z) \\ & \approx \left\{ \begin{array}{l} \mathrm{P}(TCM_k|SCM_k,NLN_k,LTC_k,SPL_k,SEP_k,z) \\ \quad \times \mathrm{P}(LTC_k|CSS_k,SCM_k,NLN_k,SEP_k,z) \\ \quad \times \mathrm{P}(CPM_k|TCM_k,SCM_k,NLN_k,z) \\ \quad \times \mathrm{P}(TPO_k|SPO_k,NLN_k,z) \end{array} \right\} \end{aligned} \quad (2)$$

The additional factor $\mathrm{P}(TPO_k|SPO_k,NLN_k,z)$ in the above equation is added to handle those newly added TM phrase-pairs. This would be the proposed ***Distinguishing Model***. For the phrases from the original SMT phrase table, both the SPO and TPO features would be "*Original*"; for the phrases from the second category mentioned in Section 3.1, the SPO would be "*Original*" but the TPO would be "*Newly-Added*"; for the phrases from the third category, both the SPO and TPO features would be "*Newly-Added*".

3.3 TM Adaptation

In real applications, the TM database is usually not big enough to train an SMT system when it is applied to a special technical domain other than the news domain. Besides, many professional translators do not want to expose the whole TM database to the SMT system providers (Cancedda, 2012). In this situation, we will be forced to first train an SMT model on an **out** domain (usually the news domain) which possesses a lot of training data, and then fix the obtained phrase-based SMT model. Afterwards, we incorporate it on line with an additional TM database which is from another **in** domain.

To simulate the above scenario, we will thus train our integrated model on the out domain. However, we have a domain-mismatch problem for this cross-domain test. Generally, in the technical domain, which is suitable for TM application, the translations (especially for technical terms) are much more consistent than that in the news domain. That is, the same source phrase in various places tends to have exactly the same translation in technical domains. Therefore, when we use Distinguishing Model to perform forced decoding, the obtained results would possess different statistics among the in-domain development set and the out-domain training set. For example, at interval [0.9, 1.0), when SCM is "Same", 94.6% of TCM are "Same" in the development set (**in**), while this ratio is only 65.1% in the training set (**out**). Therefore, the factor $P(TCM_k|SCM_k, NLN_k, LTC_k, SPL_k, SEP_k, z)$ from the test set will possess a different probability distribution in comparison with that from the training set. However, the development set is not big enough (only a few hundreds sentence-pairs at each interval) to re-train all TM factors of the proposed model. Therefore, we simply add the following h_1 feature to reflect the tendency of having high translation consistency in the development set:

$$h_1(\bar{t}, \bar{s}, z) = \begin{cases} 1.0, if\ SCM_k = Same \text{ and } TCM_k = Same \\ 0.0, otherwise \end{cases}$$

Where $\bar{s}$ and $\bar{t}$ denote the source phrase, the target candidate, respectively.

Furthermore, various source synonyms might generate the same translation (Zhu et al., 2013). Therefore, even SCM≠Same, we still favor the SMT phrase-pair candidate which exactly matches TM target phrase. For example, if source words are synonyms such as "需要" (want) and "要" (want), "如果" (if) and "若" (if), "立即" (at once) and "马上" (at once), the target translations would be the same. Therefore, the issue of having high translation consistency in the technical domain is also applied. We thus further add the following h_2 feature to reflect the tendency of having high translation consistency in this case ("High" and "Low" are grouped into "Other" for the SCM):

$$h_2(\bar{t}, \bar{s}, z) = \begin{cases} 1.0, if\ SCM_k = Other \text{ and } TCM_k = Same \\ 0.0, otherwise \end{cases}$$

Afterwards, the associated feature weights are tuned on the development set.

4 Experiments

4.1 Experimental Setup

We use the same TM data-set adopted by Wang et al. (2013), which is a Chinese–English TM database consisting of computer technical documents. It includes about 267k sentence pairs. All the experiments are conducted around this TM data-set. To compare the performances under different conditions, the same development set and the test set will be shared by both in-domain and cross-domain tests. Since the associated SMT training-set and TM database will vary under different experimental configurations, they will be specified later in each sub-section.

In this work, the translation memory system (denoted as **TM**) and the phrase-based machine translation system (denoted as **SMT**) are adopted as our two baseline systems. Following (Wang et al., 2013), for TM, the word-based fuzzy match score is adopted as the similarity measure; also, for the phrase-based SMT system, the same Moses toolkit (Koehn et al., 2007) and the same set of following features are adopted: the phrase translation model, the language model, the distance-based reordering model, the lexicalized reordering model and the word penalty. The system configurations are as follows: GIZA++ (Och and Ney, 2003) is used to obtain the bidirectional word alignments. Afterwards, "intersection" refinement (Koehn et al., 2003) is adopted to extract phrase-pairs. We use SRI Language Model

	#Sentences	#Chn. Words	#Chn. VOC.	#Eng. Words	#Eng. VOC.
New TM Database	130,953	1,808,992	30,164	1,811,413	30,807
SMT Training Set	130,953	1,814,524	29,792	1,815,615	30,516

Table 1: Corpus Statistics for In-Domain Tests

Intervals	[0.9, 1.0)	[0.8, 0.9)	[0.7, 0.8)	[0.6, 0.7)	[0.5, 0.6)	[0.4, 0.5)	[0.3, 0.4)	(0.0, 0.3)	**(0.0, 1.0)**
#Sentences	147	255	244	355	488	514	419	154	**2,576**
#Words	2,431	3,438	3,299	4,674	6,125	7,525	7,082	4,074	**38,648**
W/S	16.5	13.5	13.5	13.2	12.6	14.6	16.9	26.5	**15.0**

Table 2: Corpus Statistics for In-Domain Test-Set (W/S: the average #words per sentence)

toolkit (Stolcke, 2002) to train a 5-gram model with modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998) on the target-side (English) training corpus. All the feature weights and the weight for each probability factor are tuned on the development set with minimum-error-rate training (MERT) (Och, 2003). The maximum phrase length is set to 7 in our experiments.

To compare our proposed models with those state-of-the-art methods, we re-implement two XML-Markup approaches (Koehn and Senellart, 2010; and the upper bound version of (Ma et al, 2011)) and the Model-III (Wang et al., 2013) as three baseline systems, and denote them as **Koehn-10**, **Ma-11-U** and **Model-III**, respectively. Similar to (Wang et al., 2013), we only re-implement the XML-Markup method used in (Ma et al, 2011), but not their discriminative learning method.

Following (Wang et al., 2013), we also train the TCM, LTC and CPM factors in the SMT training set with cross-fold translation. Since the TPO factor (conditioning on NLN and Distinguishing Model) is based on Model-III, we first use Model-III to generate the desired results on the development set via forced decoding, and then generate the training samples of TPO factor for Distinguishing Model.

In this work, the translation performance is measured with case-insensitive BLEU-4 score (Papineni et al., 2002) and TER score (Snover et al., 2006). Statistical significance tests are conducted with re-sampling (1,000 times) approach (Koehn, 2004) in 95% confidence level.

4.2 In-Domain Translation Results

In the in-domain test, the original TM dataset is first randomly divided into two parts. The first part is then adopted as the new TM database, while the second part is adopted as the SMT training set. The detailed corpus statistics is shown in Table 1. Since the TM database is different from that adopted in (Wang et al., 2013), the statistics shown in Table 2 at each interval is also different from theirs.

All matched TM phrase-pairs are extracted according to the word alignment generated from the phrase-based SMT system. Since there are not enough samples to estimate the translation probabilities for those newly added TM phrase-pairs, we use the following method to assign the translation probabilities. For those TM phrase-pairs that only their source phrases exist in the original SMT phrase table (the second category mentioned in Section 3.1), as their source phrases have already existed in the SMT phrase table, there is at least one associated target phrase in the original SMT phrase table. For each new TM phrase-pair, we thus directly assign the maximum probability among its associated original target phrases to it. For those TM phrase-pairs that even their source phrase cannot be found in the original SMT phrase table (the third category), as there is no corresponding phrase-pair in the original SMT phrase table, we will simply assign probability "1.0" (this value is not important as its associated weight will be tuned later) as their four translation probabilities. To distinguish those newly added phrase-pairs from the original SMT phrase-pairs, we use eight additional feature weights for the translation probability and two more feature weights for the phrase penalty.

To evaluate the effectiveness of adding TM phrase-pairs, we compare the cases of whether merging TM phrase-pairs or not for both SMT and Model-III. Table 3 and Table 4 give the translation results in BLEU and TER, respectively. "SMT" and "Model-III" denote that we do not merge the TM phrase-pairs into the SMT phrase table during decoding. That is, they only use the original SMT phrase table.

Intervals	TM	SMT	SMT$^+$	Model-III	Model-III$^+$	Distinguishing	Koehn-10	Ma-11-U
[0.9, 1.0)	79.89	63.65	73.55 +	80.69	86.40 +*#	**86.69** +*#	82.21	67.58
[0.8, 0.9)	72.65	60.75	74.04 +	78.95 *	83.35 +*#	**83.44** +*#	79.50 *	67.03
[0.7, 0.8)	59.59	60.57	65.52 +	68.55 *	71.37 +*#	**72.06** +*#	67.52	62.60
[0.6, 0.7)	41.57	53.38	56.14 +	55.61 #	57.75 +*#	**58.73** +*#$	51.83	56.74
[0.5, 0.6)	25.17	45.60	46.95 +	47.40 #	**48.39** +*#	48.27 *#	39.08	47.94
[0.4, 0.5)	14.62	41.81	42.03	42.60 #	42.30 #	**43.04** *#$	31.60	42.93
[0.3, 0.4)	7.50	**35.95**	35.49	36.10 #	35.31 #	35.34 #	25.25	36.58
(0.0, 0.3)	4.94	32.64	33.22	33.45 #	**33.23** #	**33.23** #	23.70	33.10
(0.0, 1.0)	31.11	46.68	49.41 +	51.00 *#	52.26 +*#	**52.56** +*#$	44.28	48.91

Table 3: In-Domain Translation Results (BLEU). Scores marked with "+" indicates that those newly added TM phrase-pairs significantly ($p < 0.05$) improve the translation results ("SMT" vs. "SMT$^+$", "Model-III" vs. "Model-III$^+$", and "Model-III" vs. "Distinguishing"). Scores marked with "*" are significantly better ($p < 0.05$) than both TM and SMT$^+$ systems, and those marked with "#" are significantly better ($p < 0.05$) than Koehn-10. Scores marked with "$" are significantly better ($p < 0.05$) than Model-III$^+$ ("Model-III$^+$" vs. "Distinguishing")

Intervals	TM	SMT	SMT$^+$	Model-III	Model-III$^+$	Distinguishing	Koehn-10	Ma-11-U
[0.9, 1.0)	10.42	27.14	17.64 +	13.32	8.76 +*#	**8.22** +*#	12.95	23.94
[0.8, 0.9)	16.07	28.73	17.66 +	14.69 *	**10.46** +*#	10.49 +*#	14.72 *	23.83
[0.7, 0.8)	28.68	29.47	24.99 +	22.01 *	20.15 +*#	**19.33** +*#	23.96	27.43
[0.6, 0.7)	48.59	33.76	31.53 +	31.57 #	29.77 +*#	**28.95** +*#$	36.89	30.98
[0.5, 0.6)	63.13	40.57	39.00 +	38.79 #	**38.00** *#	38.51 #	47.08	38.44
[0.4, 0.5)	74.02	44.09	43.66	**42.84 *#**	43.43 #	42.88 *#$	55.35	42.31
[0.3, 0.4)	81.09	**50.00**	50.63	50.04 #	50.70 #	50.90 #	63.28	48.83
(0.0, 0.3)	84.34	**55.58**	56.66	54.68 #	55.96 *#	55.96 *#	68.00	54.51
(0.0, 1.0)	58.58	40.88	38.55 +	37.26 *#	36.47 +*#	**36.28** +*#	45.63	38.73

Table 4: In-Domain Translation Results (TER). The marks are the same as that in Table 3.

"SMT$^+$" and "Model-III$^+$" mean that we merge the TM phrase-pairs into the SMT phrase table dynamically. In these tables, "+" indicates that those newly added TM phrase-pairs significantly improve the translation results ("SMT" vs. "SMT$^+$", "Model-III" vs. "Model-III$^+$", and "Model-III" vs. "Distinguishing").

It can be seen that adding TM phrase-pairs significantly improve the translation results when the fuzzy match score is above 0.5 (comparing SMT with SMT$^+$, and Model-III with Model-III$^+$). For example, at interval [0.9, 1.0), those added TM phrase-pairs significantly improve the SMT system from 63.65 to 73.55, and Model-III from 80.69 to 86.40. However, if Model-III$^+$ is compared with Model-III, the improvements from merging the TM phrase-pairs get less when the fuzzy match score decreases, because the matched TM parts are fewer at low fuzzy match intervals.

Also, with the same original SMT phrase table, Model-III exceeds the SMT system at each interval. For example, at interval [0.9, 1.0), the TM information significantly improve the translation result from 63.65 to 80.69. It thus shows that the TM information is very useful. However, it is still worse than the TM in TER (13.32 vs. 10.42). On the other hand, although Model-III has greatly exceeded the SMT at each interval, Model-III$^+$ still significantly outperforms Model-III at most intervals. Therefore, the benefit of utilizing TM information and the benefit of adding TM phrase-pairs are not covered by each other and can be jointly enjoyed. Take the interval [0.9, 1.0) as an example, the TM information first improve the translation results from 63.65 (SMT) to 80.69 (Model-III), and then the added TM phrase-pairs further boosts it to 86.40 (Model-III$^+$).

Besides, Table 3 and Table 4 also present the translation results of our other two baselines (Koehn-10 and Ma-11-U), and the proposed Distinguishing Model. Scores marked with "*" indicate that they are significantly better ($p < 0.05$) than both the TM and the SMT+ baselines, and those marked with "#" are significantly better ($p < 0.05$) than Koehn-10. Scores marked with "$" are significantly better than Model-III$^+$. The bold entries are the best result at each interval.

In comparison with the TM and the SMT$^+$ systems, Model-III$^+$ is significantly better than both of them in either BLEU or TER scores when the fuzzy match score is above 0.5; also, Distinguishing Model outperforms both the TM and the SMT$^+$ systems in either BLEU or TER scores when the fuzzy match score is above 0.4. Furthermore, the improvements from both Model-III$^+$ and Distinguishing Model get less when the fuzzy match score decreases, as the TM information is less reliable at low fuzzy match intervals.

Across all intervals (the last row in the table), Distinguishing Model not only achieves the best BLEU score (52.56), but also gets the best TER score (36.28). At those intervals when the fuzzy match score is above 0.4, Model-III$^+$ and Distinguishing Model are the best two in either BLEU or TER scores. Besides, Distinguishing Model slightly exceeds Model-III$^+$ at most intervals. However, both Model-III$^+$ and Distinguishing Model achieve significant improvements over the TM and the SMT$^+$.

Compared with previous works, it can be seen that both Model-III$^+$ and Distinguishing Model significantly outperform Koehn-10 in either BLEU or TER scores at all intervals, and are significantly better than Model-III when the fuzzy match score is above 0.6. Furthermore, the proposed approaches (both Model-III$^+$ and Distinguishing Model) achieve a much better TER score than the TM system does at the interval [0.9, 1.0); while Model-III and Koehn-10 are worse than the TM system at this interval. Also, both Model-III$^+$ and Distinguishing Model exceed Ma-11-U at most intervals. Therefore, it can be concluded that the proposed models outperform previous approaches significantly in this scenario.

To further verify the proposed approaches in this case, we swap the TM database and the SMT training set and re-run the experiments. Similar and significant improvements are still observed: both Model-III$^+$ and the Distinguishing Model achieve significant improvements over the TM and the SMT$^+$. All those results have shown that the proposed approaches are robust.

In real environments, the SMT training set and the TM database could be the same before translation projects starts. However, the TM database will gradually deviate from the SMT training set while the translation task progresses. Nonetheless, our experiments have shown that the proposed Distinguishing Model is effective even when the TM database and the SMT training set are totally different (which would be the extreme case for real applications). Therefore, it can be concluded that this proposed approach is robust.

4.3 Cross-Domain Translation Results

To evaluate the cross domain performance, we adopt the news corpora about computer and science from CWMT09 (Liu and Zhao, 2009) as the SMT training set, and adopt the whole TM dataset as the TM database. The SMT training set includes about 404k bilingual sentence-pairs (which includes about 9M Chinese words and 8.7M English words). Corpus statistics is shown in Table 5. Since the TM database and the test set (also the development set) are the same as that in (Wang et al., 2013), the statistics at each interval is the same as theirs but different from Table 2.

The training procedure is the same as that mentioned in the last sub-section. Table 6 and Table 7 present the translation results of TM, SMT, SMT$^+$, two baselines (Koehn-10 and Model-III), and three proposed approaches (Model-III$^+$, Distinguishing and Adaptation). The Adaptation approach means that we add two consistent related features based on Distinguishing Model (Section 3.3). All the formats are the same as that adopted in Table 3 and Table 4. Besides, scores marked by "&" are significantly better than Distinguishing Model.

Comparing the TM with the SMT, the performance of in-domain TM significantly exceeds that of out-domain SMT. Since the fuzzy match intervals are divided according to the TM database, the translation result of the SMT system at interval [0.8, 0.9) even slightly outperforms that at interval [0.9, 1.0). Besides, adding TM phrase-pairs significantly improves the translation results when the fuzzy match score is above 0.5 (SMT vs. SMT$^+$, and Model-III vs. Model-III$^+$). Furthermore, the benefit of utilizing TM information and the benefit of adding TM phrase-pairs are not covered by each other, and can be jointly enjoyed. Furthermore, compared with TM, SMT, SMT$^+$ and Model-III, both Model-III$^+$ and Distinguishing Model achieve better translation results when the fuzzy match score is above 0.4. All observed trends are similar to that in the last sub-section.

	#Sentences	#Chn. Words	#Chn. VOC.	#Eng. Words	#Eng. VOC.
TM Database	261,906	3,623,516	43,112	3,627,028	44,221
SMT Training Set	404,172	9,007,614	102,073	8,737,801	107,883

Table 5: Corpus Statistics for Cross-Domain Tests

Intervals	TM	SMT	SMT$^+$	Model-III	Model-III$^+$	Distinguishing	Adaptation	Koehn-10
[0.9, 1.0)	81.31	30.87	64.74 +	64.79	82.28 +	83.19 +*$	**84.89** *#$&	81.52
[0.8, 0.9)	73.25	31.94	60.13 +	61.91	74.21 +	74.72 +*	**79.78** *#$&	76.47 *
[0.7, 0.8)	63.62	30.63	51.64 +	51.44	62.94 +	63.32 +	**67.74** *$&	67.12 *$&
[0.6, 0.7)	43.64	28.95	39.94 +	38.28	46.28 +*	46.46 +*	**49.49** *$&	48.47 *
[0.5, 0.6)	27.37	27.61	32.49 +	28.85	34.50 +*	34.87 +*	**37.12** *#$&	35.25 *
[0.4, 0.5)	15.43	27.16	27.35	27.30 #	27.47 #	27.82 #	**28.80** *#$&	25.10
[0.3, 0.4)	8.24	**23.85**	22.66	23.81 #	22.41 #	22.41 #	22.95 #	20.72
(0.0, 0.3)	4.13	**24.64**	24.25	24.24 #	23.65 #	24.12 #	24.31 #	18.79
(0.0, 1.0)	40.17	28.30	40.59 +	40.47	47.37 +*	47.70 +*#$	**49.79** *#$&	47.09 *

Table 6: Cross-Domain Translation Results (BLEU). The marks are the same as that in Table 3. Besides, scores marked by "$" are significantly better ($p < 0.05$) than Model-III$^+$, and those marked by "&" are significantly better than "Distinguishing" ("Adaptation" vs. "Distinguishing").

Intervals	TM	SMT	SMT$^+$	Model-III	Model-III$^+$	Distinguishing	Adaptation	Koehn-10
[0.9, 1.0)	9.79	54.54	27.07 +	27.09	11.81 +	11.01 +	**9.58** #$&	13.51
[0.8, 0.9)	16.21	52.86	29.33 +	28.04	17.13 +	17.47 +	**13.80** *#$&	17.29
[0.7, 0.8)	27.79	52.42	36.48 +	35.56	27.07 +	26.40 +$	**23.04** *$&	24.31 *$&
[0.6, 0.7)	46.40	54.74	47.39 +	48.06	41.13 +*	40.36 +*$	**37.45** *#$&	40.16 *
[0.5, 0.6)	62.59	57.18	53.08 +	56.78	51.77 +*	51.60 +*	**48.08** *#$&	51.57
[0.4, 0.5)	73.93	57.19	56.57	57.19 #	56.82 #	56.53 #	**54.42** *#$&	61.32
[0.3, 0.4)	79.86	60.62	61.16	61.35 #	61.31 #	61.31 #	**60.33** #$&	68.82
(0.0, 0.3)	85.31	63.62	62.81	62.22 #	63.04 #	62.07 #	**61.87** #	74.85
(0.0, 1.0)	50.51	56.42	46.89 +	47.38 #	41.63 +*#	41.27 +*#$	**38.87** *#$&	43.95 *

Table 7: Cross-Domain Translation Results (TER). The marks are the same as that in Table 6.

However, both Model-III$^+$ and Distinguishing Model are worse than Koehn-10 at some high fuzzy match intervals. The reason is that the TM factors are trained on the news domain but the test set is from computer technical domain. Therefore, it is not strange that the Adaptation approach achieves the best translation results at all intervals in either BLEU or TER when the fuzzy match score is above 0.4. At most intervals, the Adaptation approach significantly outperforms Koehn-10 in either BLEU or TER, especially for the high fuzzy match intervals such as [0.9, 1.0) and [0.8, 0.9). Furthermore, the Adaptation approach achieves better TER than the TM system and Koehn-10 at intervals [0.9, 1.0) and [0.8, 0.9). All obtained results have shown that the Adaptation approach is effective and robust for cross-domain test. Moreover, it can be seen that the h1 feature (mentioned in Section 3.3) is more effective than the h2 feature.

5 Related Work

According to the way of combination, those previous works can be classified into four categories (as specified in Section 1). The first category uses a classifier (or a re-ranker) to judge whether TM or SMT gives a better translation sentence, and then delivers the better one to the post-editor (He et al., 2010a; He et al., 2010b; Dara et al., 2013). Since the outputs of SMT and TM are not merged but only re-ranked, the possible improvement resulted from those approaches is quite limited.

The second category incorporates TM matched parts into the SMT input sentence in a pipelined manner (Koehn and Senellart, 2010; Zhechev and van Genabith, 2010; He et al., 2011; Ma et al., 2011). These approaches usually translate the sentence in two stages: (1) first determine whether the

extracted TM sentence pair should be adopted or not, and then merge the relevant translations of matched parts into the input sentence; (2) then force the SMT system to only translate those unmatched parts at decoding. There are three drawbacks for this kind of pipeline approaches (Wang et al., 2013). Firstly, whether those matched parts should be adopted or not is determined at the sentence level. Secondly, they select only one TM target phrase before decoding. Thirdly, they do not utilize the SMT probabilistic information for the matched parts.

The third category mainly adds the longest matched TM phrase pairs into the SMT phrase table (Bi ̧ici and Dymetman, 2008; Simard and Isabelle, 2009), and associates them with a fixed large probability value to favor the TM target phrase. However, they only add one aligned target phrase for each matched source phrase and did not distinguish the original and the newly-added phrase-pairs.

The last category incorporates the associated TM information of each source phrase into the SMT during decoding (Wang et al., 2013). This category can avoid the drawbacks of the pipeline approaches, and thus achieves superior results when the TM database and the SMT training set are the same. However, they only refer to the TM information and do not regard the TM phrase-pairs as candidates during decoding. Therefore, the superiority of this approach disappears when the TM database and the SMT training set are different, because many TM phrase-pairs cannot be found in the original SMT phrase table in this case.

Our approach combines the strength of both the third and the last categories. During decoding, the associated TM information is referred to re-score the SMT candidates. At the same time, all matched TM phrase-pairs are dynamically merged into the phrase table. Moreover, this is the first unified framework for integrating TM into SMT at decoding when the TM database and the SMT training set are different. Although some previous works of the second and third categories can be also applied when the TM database and the SMT training set are different, they did not explicitly focus on and test this case.

Last, since the example-based machine translation (EBMT, [Nagao, 1984]) is similar to that of using TM, some approaches (Watanabe and Sumita, 2003; Smith and Clark, 2009; Dandapat et al., 2011; 2012; Phillips, 2011) also combined EBMT with SMT. It would be interesting to compare our approaches with theirs in the future.

6 Conclusion

Combining TM and SMT can greatly improve the translation performance and reduce human post-editing effort. In comparison with those previous approaches, our work makes the following contributions:

(1) Dynamically merge the matched TM phrase-pairs into the SMT phrase table to meet the real application;
(2) Propose an improved integrated model to distinguish the original SMT phrase-pairs from the newly-added ones extracted from TM;
(3) Adopt a simple but effective TM adaptation method to favor the consistent translation in cross-domain test.

This is the first work adopting a unified framework to integrate the TM information into the SMT model during decoding when the TM database and the SMT training set are different. On the TM database which consists of Chinese–English computer technical documents, our experiments have shown that merging the TM phrase-pairs achieves significant improvements when the fuzzy match score is above 0.5. Furthermore, the proposed approaches are significantly better than either the SMT or the TM systems for both the in-domain and the cross-domain tests. Last, the proposed approaches outperform previous works significantly in all test conditions.

Acknowledgements

This research work was partially funded by the Natural Science Foundation of China under Grant No. 61333018, the Hi-Tech Research and Development Program ("863" Program) of China under Grant No. 2012AA011101, the Key Project of Knowledge Innovation Program of Chinese Academy of Sciences under Grant No. KGZD-EW-501, and Toshiba (China) R&D Center.

Reference

Ergun Bi ̧ici and Marc Dymetman. 2008. Dynamic translation memory: using statistical machine translation to improve translation memory fuzzy matches. In *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2008)*, pages 454–465.

Nicola Cancedda. 2012. Private Access to Phrase Tables for Statistical Machine Translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 23–27.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. *Technical Report TR-10-98*, Harvard University Center for Research in Computing Technology.

Chiang, David. 2005. A hierarchical phrase-based model for statistical machine translation, In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL)*, pages 263–270.

Sandipan Dandapat, Sara Morrissey, Andy Way, and Mikel L Forcada. 2011. Using example-based MT to support statistical MT when translating homogeneous data in resource-poor settings, In *Proceedings of the 15th Annual Meeting of the European Association for Machine Translation (EAMT 2011)*, pages 201–208.

Sandipan Dandapat, Sara Morrissey, Andy Way, and Joseph Van Genabith. 2012. Combining EBMT, SMT, TM and IR technologies for quality and scale, In *Proceedings of the Joint Workshop on Exploiting Synergies between Information Retrieval and Machine Translation (ESIRMT) and Hybrid Approaches to Machine Translation (HyTra)*, pages 48–58.

Aswarth Dara, Sandipan Dandapat, Declan Groves, and Josef van Genabith. TMTprime: a recommender system for MT and TM integration. In *Proceedings of the NAACL HLT 2013 Demonstration Session*, pages 10–13.

Yifan He, Yanjun Ma, Josef van Genabith and Andy Way, 2010a. Bridging SMT and TM with translation recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 622–630.

Yifan He, Yanjun Ma, Andy Way, and Josef Van Genabith. 2010b. Integrating N-best SMT outputs into a TM system, In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 374–382.

Yifan He, Yanjun Ma, Andy Way and Josef van Genabith. 2011. Rich linguistic features for translation memory-inspired consistent translation. In *Proceedings of the Thirteenth Machine Translation Summit*, pages 456–463.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 388–395, Barcelona, Spain.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer and Ondřej Bojar. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 177–180.

Philipp Koehn, Franz Josef Och and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54.

Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *AMTA Workshop on MT Research and the Translation Industry*, pages 21–31.

Liu, Qun and Hongmei Zhao. 2009. Report on CWMT2009 MT Translation Evaluation. In *Proceedings of the 5th China Workshop on Machine Translation (CWMT2009)*, pages 1–31, Nanjing, China.

Yanjun Ma, Yifan He, Andy Way and Josef van Genabith. 2011. Consistent translation using dis-criminative learning: a translation memory-inspired approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1239–1248, Portland, Oregon.

Makoto Nagao, 1984. A framework of a mechanical translation between Japanese and English by anal-ogy principle. In: Banerji, Alick Elithorn and Ran-an (ed). *Artifiical and Human Intelligence: Edited Review Papers* Presented at the International NATO Symposium on Artificial and Human Intelligence. North-Holland, Amsterdam, 173–180.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29 (1). pages 19–51.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Aaron B. Phillips, 2011. Cunei: open-source machine translation with relevance-based models of each translation instance. *Machine Translation*, 25 (2). pages 166-177.

Michel Simard and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, pages 120–127.

James Smith and Stephen Clark. 2009. EBMT for SMT: a new EBMT-SMT hybrid. In *Proceedings of the 3rd International Workshop on Example-Based Machine Translation (EBMT'09)*, pages 3–10, Dublin, Ireland.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Ma-chine Translation in the Americas (AMTA-2006)*, pages 223–231.

Andreas Stolcke. 2002. SRILM-an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 311–318.

Taro Watanabe, Eiichiro Sumita. 2003. Example-based decoding for statistical machine translation, In *Proceeding of Machine Translation Summit IX*, pages 410–417.

Kun Wang, Chengqing Zong and Keh-Yih Su, 2013. Integrating translation memory into phrase-based machine translation during decoding. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 11–21.

Ventsislav Zhechev and Josef van Genabith. 2010. Seeding statistical machine translation with translation memory output through tree-based structural alignment. In *Proceedings of the 4th Workshop on Syntax and Structure in Statistical Translation*, pages 43–51.

Xiaoning Zhu, Zhongjun He, Hua Wu, Haifeng Wang, Conghui Zhu, and Tiejun Zhao. 2013. Improving pivot-based statistical machine translation using random walk. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 524–534.

Machine Translation Quality Estimation Across Domains

José G. C. de Souza
University of Trento
Fondazione Bruno Kessler
Trento, Italy
desouza@fbk.eu

Marco Turchi
Fondazione Bruno Kessler
Trento, Italy
turchi@fbk.eu

Matteo Negri
Fondazione Bruno Kessler
Trento, Italy
negri@fbk.eu

Abstract

Machine Translation (MT) Quality Estimation (QE) aims to automatically measure the quality of MT system output without reference translations. In spite of the progress achieved in recent years, current MT QE systems are not capable of dealing with data coming from different train/test distributions or domains, and scenarios in which training data is scarce. We investigate different multitask learning methods that can cope with such limitations and show that they overcome current state-of-the-art methods in real-world conditions where training and test data come from different domains.

1 Introduction

Machine Translation (MT) Quality Estimation (QE) aims to automatically predict the quality of MT output without using reference translations (Blatz et al., 2003; Specia et al., 2009). QE systems usually employ supervised machine learning models that use different information extracted from (source, target) sentence pairs as features along with quality scores as labels. The notion of quality that these models measure can be indicated by different scores. Some examples are the average number of edits required to post-edit the MT output, i.e., human translation edit rate[1] (HTER (Snover et al., 2006)), and the time (in seconds) required to post-edit a translation produced by an MT system (Specia, 2011).

Research on QE has received a strong boost in recent years due to the increase in the usage of MT systems in real-world applications. Automatic and reference-free MT quality prediction demonstrated to be useful for different applications, such as: deciding whether the translation output can be published without post-editing (Soricut and Echihabi, 2010), filtering out low-quality translation suggestions that should be rewritten from scratch (Specia et al., 2009), selecting the best translation output from a pool of MT systems (Specia et al., 2010), and informing readers of the translation whether it is reliable or not (Turchi et al., 2012). Another example is the computer-assisted translation (CAT) scenario, in which it might be necessary to predict the quality of translation suggestions generated by different MT systems to support the activity of post editors working with different genres of text.

The dominant QE framework presents some characteristics that can limit models' applicability in such real-world scenarios. First, the scores used as training labels (HTER, time) are costly to obtain because they are derived from manual post-editions of MT output. Such requirement makes it difficult to develop models for domains in which there is a limited amount of labeled data. Second, the learning methods currently used (for instance in the framework of QE shared evaluation campaigns)[2] assume that training and test data are sampled from the same distribution. Though reasonable as a first evaluation setting to promote research in the field, this controlled scenario is not realistic as different data in real-world applications might be post-edited by different translators, the translations might be generated by different MT systems and the documents being translated might belong to different domains or genres. To

[1] Edit distance is calculated as the number of edits (word insertions, deletions, substitutions, and shifts) divided by the number of words in the reference. Lower HTER values indicate better translations.

[2] In the last two editions of the yearly Workshop on Machine Translation, several QE shared tasks have been proposed (Callison-Burch et al., 2012; Bojar et al., 2013).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 409–420, Dublin, Ireland, August 23-29 2014.

overcome these limitations a plausible research objective is to exploit techniques that: (i) allow domains and distributions of features to be different between training and test data, and (ii) that cope with the scarce amount of training labels by sharing information across domains, a common scenario for transfer learning.

In this paper we investigate the use of techniques that can exploit the training instances from different domains to learn a QE model for a specific target domain for which there is a small amount of labeled data. In particular, we are interested in approaches that allow not only learning from one single source domain but also from multiple source domains simultaneously, by leveraging the labels from all available data to improve results in a target domain.

Given these requirements, we experiment with different *multitask learning* techniques that perform transfer learning via a common task structure (domain relatedness). Furthermore, we employ an approach based on *feature augmentation* that has been successfully used in other natural language processing tasks. We present a series of experiments over three domains with increasing amounts of training data, showing that our adaptive approaches outperform competitive baselines.

The contributions of our work are: (i) a first exploration of techniques that overcome the limitation of current QE learning methods when dealing with data with different training and test distributions and domains, and (ii) an empirical verification of the amount of training data required by such techniques to outperform competitive baselines on different target domains. To the best of our knowledge, this is the first work addressing the challenges posed by domain adaptation in MT QE.

2 Related Work

Quality estimation has recently gained increasing attention, also boosted by two evaluation campaigns organized within the Workshop on Machine Translation (WMT) (Callison-Burch et al., 2012; Bojar et al., 2013). The bulk of work done so far has focused on the controlled WMT evaluation framework and, in particular, on two major aspects of the problem: feature engineering and machine learning methods.

Feature engineering accounts for linguistically-based predictors that aim to model different perspectives of the quality estimation problem. The research ranges from identifying indicators that approximate the complexity of translating the source sentence and designing features that model the fluency of the automatically generated translation, to linguistically motivated measures that estimate how adequate the translation is in comparison to the source sentence in terms of meaning (Blatz et al., 2003; Mehdad et al., 2012; Hardmeier et al., 2012; Rubino et al., 2012; Specia et al., 2012; de Souza et al., 2013a).

State-of-the-art QE explores different supervised linear or non-linear learning methods for regression or classification such as Support Vector Machines (SVM), different types of Decision Trees, Neural Networks, Elastic-Net, Gaussian Processes, Naive Bayes, among others (Specia et al., 2009; Buck, 2012; Beck et al., 2013; Souza et al., 2014). Another aspect related to the learning methods that has received attention is the optimal selection of features in order to overcome issues related with the high-dimensionality of the feature space (Soricut et al., 2012; de Souza et al., 2013a; Beck et al., 2013; de Souza et al., 2013b).

Despite constant improvements, such learning methods have limitations. The main one is that they assume that both training and test data are independently and identically distributed. As a consequence, when they are applied to data from a different distribution or domain they show poor performance. This limitation harms the performance of QE systems for several real-world applications, such as CAT environments. Advanced CAT systems currently integrate suggestions obtained from MT engines with those derived from translation memories (TMs). In such framework, the compelling need to speed up the translation process and reduce its costs by presenting human translators with good-quality suggestions raises interesting research challenges for the QE community. In such environments, translation jobs come from different domains that might be translated by different MT systems and are routed to professional translators with different idiolect, background and quality standards (Turchi et al., 2013). Such variability calls for flexible and adaptive QE solutions by investigating two directions: (i) modeling translator behaviour (Turchi et al., 2014) and (ii) maximize the learning capabilities from all the available data. The second research objective motivates our investigation on methods that allow the training and test domains and

the distributions to be different.

Recent work in QE focused on aspects that are problematic even in the controlled WMT scenario, and are closely related to the flexibility/adaptability issue. Focusing on the first of the two aforementioned directions (i.e. modeling translators' behaviour), Cohn and Specia (2013) propose a Multitask Gaussian Process method that jointly learns a series of annotator-specific models and that outperforms models trained for each annotator. Our work differs from theirs in that we are interested in the latter research direction (i.e. coping with domain and distribution diversity) and we use in and out-of-domain data to learn robust in-domain models. Our scenario represents a more challenging setting than the one tackled in (Cohn and Specia, 2013), which does not consider different domains.

In *transfer learning* there are many techniques suitable to fulfill our requirements. The aim of transfer learning is to extract the knowledge from one or more source tasks and apply it to a target task (Pan and Yang, 2010). One type of transfer learning is *multitask learning* (MTL), which uses domain-specific training signals of related tasks to improve model generalization (Caruana, 1997). Although it was not originally thought for transferring knowledge to a new task, MTL can be used to achieve this objective due to its capability to capture task relatedness, which is important knowledge that can be applied to a new task (Jiang, 2009).

Domain adaptation is a kind of transfer learning in which source and target domains (i.e. training and test) are different but the tasks are the same (Pan and Yang, 2010). The domain adaptation techniques that inspire our work have been successfully applied to a variety of NLP tasks (Blitzer et al., 2006; Jiang and Zhai, 2007). For instance, an effective solution for supervised domain adaptation, EasyAdapt (SVR FEDA henceforth), was proposed in (Daumé III, 2007) and applied to named entity recognition, part-of-speech tagging and shallow parsing. The approach transforms the domain adaptation problem into a standard learning problem by augmenting the source and target feature set. The feature space is transformed to be a cross-product of the features of the source and target domains augmented with the original target domain features. In *supervised* domain adaptation one has access to out-of-domain labels and wants to leverage a small amount of available in-domain labeled data to train a model (Daumé III, 2007), the case of this study. This is different from the *semi-supervised* case in which in-domain labels are not available.

3 Adaptation for QE

An important assumption in MTL is that different tasks (domains in our case) are correlated via a certain structure. Examples of such structures are the hidden layers in a neural network (Caruana, 1997) and shared feature representation (Argyriou et al., 2007) among others. This common structure allows for knowledge transfer among tasks and has been demonstrated to improve model generalization over single task learning (STL) for different problems in different areas. Under this scenario, several assumptions can be made about the relatedness among the tasks, leading to different transfer structures. We explore three approaches to MTL that deal with task relatedness in different ways. These are the "Dirty" approach to MTL (Jalali et al., 2010), Sparse Trace MTL (Chen et al., 2012) and Robust MTL (Chen et al., 2011). The three approaches use different regularization techniques that capture task relatedness using norms over the weights of the features.

Before describing the three approaches, we introduce some basic notation similar to (Chen et al., 2011). In MTL there are T tasks and each task $t \in T$ has m training samples $\{(x_1^{(t)}, y_1^{(t)}), \ldots, (x_m^{(t)}, y_m^{(t)})\}$, with $x_i^{(t)} \in \mathbb{R}^d$ where d is the number of features and $y_i^{(t)} \in \mathbb{R}$ is the output (the response variable or label). The input features and labels are stacked together to form two different matrices $X^{(t)} = [x_1^{(t)}, \ldots, x_m^{(t)}]$ and $Y^{(t)} = [x_1^{(t)}, \ldots, x_m^{(t)}]$, respectively. The weights of the features for each task are represented by W, where each column corresponds to a task and each row corresponds to a feature.

The **"Dirty"** approach to MTL follows the idea that different tasks may share the same discriminative features (Argyriou et al., 2007). However, it also considers that different tasks might have different discriminative features that are inherent to each task. Therefore, the method encourages shared-sparsity among tasks and among features in each task. It decomposes W into two components, one is a row-

sparsed matrix that corresponds to the features shared among the tasks and the other is an element-wise sparse matrix that corresponds to the non-shared features that are important for each task independently. More formally, the "Dirty" approach is explained by Equation 1.

$$\min_{W} \sum_{t=1}^{T} ||(W^{(t)}X^{(t)} - Y^{(t)})||_2^2 + \lambda_s||S||_1 + \lambda_b||B||_{1,\infty} \text{ subject to: } W = S + B \quad (1)$$

where $||(W^{(t)}X^{(t)} - Y^{(t)})||_2^2$ is the least squares loss function, S is the regularization term that encourages element-wise sparsity and B is the block-structured row-sparsity regularizer. The $||.||_2$ is the l_2-norm (Euclidean distance), $||.||_1$ is the l_1-norm (given by $\sum_{i=1} |x_i|$) and $||.||_{1,\infty}$ is the row grouped l_1-norm. The λ_s and λ_b are non-negative trade-off parameters that control the amount of regularization applied to S and B, respectively.

Sparse Trace MTL considers the problem of learning incoherent sparse and low-rank patterns from multiple related tasks. This approach captures task relationship via a shared low-rank structure of the weight matrix W. As computing the low-rank structure of a matrix leads to a NP-hard optimization problem, Chen et al. (2012) proposed to compute the trace norm as a surrogate, making the optimization problem tractable. In addition to learning the low-rank patterns, this method also considers the fact that different tasks may have different inherent discriminative features. It decomposes W into two components: S, which models element-wise sparsity, and Q, which captures task relationship via the trace norm. The convex problem minimized by Sparse Trace is given in Equation 2.

$$\min_{W} \sum_{t=1}^{T} ||(W^{(t)}X^{(t)} - Y^{(t)})||_2^2 + \lambda_s||S||_1 \text{ subject to: } W = S + Q, ||Q||_* < \lambda_p \quad (2)$$

where $||.||_*$ is the trace norm, given by the sum of the singular values σ_i of W, i.e., $||W||_* = \sum_{i=1} \sigma_i(W)$. Here, λ_p controls the rank of Q and λ_s controls the sparsity of S.

The key assumption in MTL is that tasks are related in some way. However, this assumption might not hold for a series of real-world problems. In situations in which tasks are not related a negative transfer of information among tasks might occur, harming the generalization of the model. One way to deal with this problem is to: (i) group related tasks in one structure and share knowledge among them, and (ii) identify irrelevant tasks maintaining them in a different group that does not share information with the first group. This is the idea of **Robust** MTL (RMTL henceforth). The algorithm approximates task relatedness via a low-rank structure like Sparse Trace and identifies outlier tasks using a group-sparse structure (column-sparse, at task level). Robust MTL is described by Equation 3. It employs a non-negative linear combination of the trace norm (the task relatedness component L) and a column-sparse structure induced by the $l_{1,2}$-norm (the outlier task detection component S). If a task is an outlier it will have non-zero entries in S.

$$\min_{W} \sum_{t=1}^{T} ||(W^{(t)}X^{(t)} - Y^{(t)})||_2^2 + \lambda_l||L||_* + \lambda_s||S||_{1,2} \text{ subject to: } W = L + S \quad (3)$$

where $||S||_{1,2}$ is the group regularizer that induces sparsity on the tasks.

4 Experimental Setting

In this section we describe the data used for our experiments, the features extracted, the set up of the learning methods, the baselines used for comparison and the evaluation of the models. The goal of our experiments is to show that the methods presented in Section 3 outperform competitive baselines and standard QE learning methods that are not capable of adapting to different domains. We experiment with three different domains of comparable size and evaluate the performance of the adaptive methods and the standard techniques with different amounts of training data. The MTL models described in section 3 are trained with the Malsar toolkit implementation (Zhou et al., 2012). The hyper-parameters are optimized

using 5-fold cross-validation in a grid search procedure. The parameter values are searched in an interval ranging from 10^{-3} to 10^{3}.

4.1 Data

Our experiments focus on the English-French language pair and encompass three very different domains: newswire text (henceforth News), transcriptions of Technology Entertainment Design talks (TED) and Information Technology manuals (IT). Such domains are a challenging combination for adaptive systems since they come from very different sources spanning speech and written discourse (TED and News/IT, respectively) as well as a very well defined and controlled vocabulary in the case of IT.

Each domain is composed of 363 tuples formed by the source sentence in English, the French translation produced by an MT system and a human post-edition of the translated sentence. For each pair (translation, post-edition) we use as labels the HTER score computed with TERCpp[3]. For the three domains we use half of the data for training (181 instances) and half of the data for testing (182 instances). The limited amount of instances for training contrasts with the 800 or more instances of the WMT evaluation campaigns and is closer to real-world applications where the availability of large and representative training sets is far from being guaranteed (e.g. the CAT scenario).

The sentence tuples for the first two domains are randomly sampled from the Trace corpus[4]. The translations were generated by two different MT systems, a state-of-the-art phrase-based statistical MT system and a commercial rule-based system. Furthermore, the translations were post-edited by up to four different translators, as described in (Wisniewski et al., 2013).

Domain	No. of tokens	Vocab. size	Avg. sent. length
TED source	6858	1659	19
TED target	7016	1828	19
IT source	3310	1004	9
IT target	3134	1049	8
News source	7605	2273	21
News target	8230	2346	23

Table 1: Datasets statistics for each domain.

The TED talks domain is formed by subtitles of several talks in a range of topics presented in the TED conferences. The complete dataset has been used for MT and automatic speech recognition systems evaluation within the International Workshop on Spoken Language Translation (IWSLT). The News domain is formed by newswire text used in WMT translation campaigns and covers different topics. The IT texts come from a software user manual translated by a statistical MT system based on the state-of-the-art phrase-based Moses toolkit (Koehn et al., 2007) trained on about 2M parallel sentences. The post-editions were collected from one professional translator operating on the Matecat[5] CAT tool in real working conditions. Table 1 provides macro-indicators (number of tokens, vocabulary size, average sentence length) that evidence the large difference between the domains addressed by our experiments and give an idea of the difficulty of the task.

A peculiarity of the TED domain is that it is formed by manual transcriptions of speech translated by different MT systems, configuring a different type of discourse than News and IT. In TED, the vocabulary size in the source and target sentences is lower than that of the News domain but higher than IT. News presents the most varied vocabulary, which is an evidence of the more varied lexical choice represented by the several topics that compose the domain. Moreover, News has the highest average sentence length, a characteristic of non-technical written discourse, which tends to have longer sentences than spoken discourse and domains dominated by technical jargon. Such a characteristic is exactly what differentiates IT from the other two domains. IT sentences are technical and present a reduced average number of

[3] http://sourceforge.net/projects/tercpp/

[4] http://anrtrace.limsi.fr/trace_postedit.tar.bz2

[5] www.matecat.com

words, as evidenced by the vocabulary size (the smallest among the three domains). These numbers suggest a divergence between IT and the other two domains, possibly making adaptation more difficult.

4.2 Features

For all the experiments we use the same feature set composed of seventeen features proposed in (Specia et al., 2009). The set is formed by features that model the complexity of translating the source sentence (e.g. the average source token length or the number of tokens in the source sentence), and the fluency of the translated sentence produced by the MT system (e.g. the language model probability of the translation). The decision to use this feature set is motivated by the fact that it demonstrated to be robust across language pairs, MT systems and text domains (Specia et al., 2009). The 17 features are:

- number of tokens in the source sentence and in the generated translation;
- average source token length;
- average number of occurences of the target word within the generated translation;
- language model probability of the source sentence and generated translation;
- average number of translations per source word in the sentence: as given by IBM 1 model thresholded so that $P(t|s) > 0.2$ weighted by the inverse frequency of each word in the source side of the SMT training corpus ◯;
- average number of translations per source word in the sentence: as given by IBM 1 model thresholded so that $P(t|s) > 0.01$ weighted by the inverse frequency of each word in the source side of the SMT training corpus;
- percentage of unigrams◯, bigrams and trigrams◯ in the first quartile of frequency (lower frequency words) in a corpus of the source language;
- percentage of unigrams◯, bigrams and trigrams in the fourth quartile of frequency (higher frequency words) in a corpus of the source language;
- percentage of unigrams in the source sentence seen in the source side of the SMT training corpus;
- number of punctuation marks in the source sentence and in the hypothesis translation;

4.3 Baselines

As a term of comparison, we consider these baselines in our experiments. A simple to implement but difficult to beat baseline when dealing with regression on tasks with different distributions is to compute the mean of the training labels and use it as the prediction for each testing point (Rubino et al., 2013). Hereafter we refer to this baseline as μ. Since supervised domain adaptation techniques should outperform models that are trained only on the available in-domain data, we also use as baseline the regressor built only on the available in-domain data (SVR in-domain). Furthermore, as a third baseline, we train a regressor by pooling together training data of all domains, combining source and target data without any kind of task relationship mechanism (SVR Pooling).

The baselines are trained on the feature set described earlier in Section 4.2 with an SVM regression (SVR) method using the implementation of Scikit-learn (Pedregosa et al., 2011). The radial basis function (RBF) kernel is used for all baselines. The hyper-parameters of the model are optimized using randomized search optimization process with 50 iterations as described in (Bergstra and Bengio, 2012) and used previously for QE in (de Souza et al., 2013a). The best parameters are found using 5-fold cross-validation on the training data and ϵ, γ and C are sampled from exponential distributions scaled at 0.1 for the first two parameters and scaled at 100 for the last one. It is important to notice that the SVR with RBF kernel methods learn non-linear models that have been shown to perform better than linear models on the set of features used for predicting HTER. On the contrary, the MTL methods presented in Section 3 are methods that do not explore kernels or any other kind of non-linear learning method.

Source / Target	IT_{tgt}	$News_{tgt}$	TED_{tgt}
IT_{src}	0.2081	0.2341	0.2232
$News_{src}$	0.2368	0.1690	0.2130
TED_{src}	0.2183	0.2263	0.1928

Table 2: Results of the SVR in-domain baseline trained and evaluated in each domain (average of 50 different shuffles). Rows represent the domain data used to train the model and columns represent the domain data used to evaluate the model. Scores are MAE.

4.4 Evaluation

The accuracy of the models is evaluated with the mean absolute error (MAE), which was also used in previous work and in the WMT QE shared tasks (Bojar et al., 2013). MAE is the average of the absolute difference between the prediction $\hat{y}_i$ of a model and the gold standard response y_i (Equation 4). As it is an error measure, lower values mean better performance.

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|\hat{y}_i - y_i| \tag{4}$$

To test the statistical significance of our results we need to perform comparisons of multiple models. In addition, we would like to test the significance over different training amounts. Given these requirements we need to perform multiple hypothesis tests instead of paired tests. It has been shown that for comparisons of multiple machine learning models, the recommended approach is to use a non-parametric multiple hypothesis test followed by a post-hoc analysis that compares each pair of hypothesis (Demšar, 2006). In our experiments we use the Friedman test (Friedman, 1937; Friedman, 1940) followed by a post-hoc analysis of the pairs of regressors using Holm's procedure (Holm, 1979) to perform the pairwise comparisons when the null hypothesis is rejected. All tests for both Friedman and post-hoc analysis are run with $\alpha = 0.05$. For more details about these methods, we refer the reader to (Demšar, 2006; Garcia and Herrera, 2008) which provide a complete review about the application of multiple hypothesis testing to machine learning methods.

5 Results and Discussion

Our experiments are organized as follows. First, we evaluate the performance of single task learning methods on different cross-domain experiments. Then, we report the evaluation for the multitask learning methods and discuss the results.

5.1 Single Task Learning

With the objective of having an insight about the difference between the domains, we train the SVR in-domain baseline with all available training data for each domain and evaluate its performance on the same domain and in the two remaining domains.

Results are reported in Table 2, where the diagonal shows the figures for the in-domain evaluation. These numbers suggest that the IT domain configures a more difficult challenge for the learning algorithm. The IT in-domain model (IT_{src}-IT_{tgt}) presents a performance 21% inferior to News and 8% inferior to TED. For all models trained on a source domain different than the target domain there is a drop in performance, as it is expected from a system that assumes that training and test data are sampled from the same distribution. In addition, when predicting IT using the model trained on News, we have a perfomance drop of 13% whereas using the model trained on TED the performance drops up to 4%.

5.2 Multitask learning

We run the baselines described in Section 4.3 and the methods described in Section 3 on different amounts of training data, ranging from 18 to 181 instances (10% and 100%, respectively). The motivation is to verify how much training data is required by the MTL methods to outperform the baselines for a target domain. Table 3 presents the results for the three domains with models trained on 30, 50 and

100% of the training data (54, 90 and 181 instances, respectively). Each method was run on 50 different train/test splits of the data in order to account for the variability of points in each split.

Method	TED	News	IT
30 % of training data (54 instances)			
mean	0.1951	0.1711	0.2174
SVR In-Domain	0.2013	0.1753	0.2235
SVR Pooling	0.1962	0.1899	0.2201
SVR FEDA	0.1952	0.1839	0.2193
MTL Dirty	0.1954	0.1708	0.2193
MTL SparseTrace	0.1976	0.1743	0.2222
MTL RMTL	**0.1946**	**0.1685**	**0.2162**
50% of training data (90 instances)			
mean	0.1943	0.1707	0.2170
SVR In-Domain	0.1976	0.1711	0.2183
SVR Pooling	0.1951	0.1865	0.2191
SVR FEDA	0.1937	0.1806	0.2161
MTL Dirty	0.1927	0.1678	0.2148
MTL SparseTrace	0.1922	0.1672	0.2157
MTL RMTL	**0.1878**	**0.1653**	**0.2119**
100% of training data (181 instances)			
mean	0.1936	0.1690	0.2162
SVR In-Domain	0.1928	0.1690	0.2081
SVR Pooling	0.1927	0.1849	0.2203
SVR FEDA	0.1908	0.1757	0.2107
MTL Dirty	0.1878	0.1666	0.2083
MTL SparseTrace	0.1881	0.1661	0.2094
MTL RMTL	**0.1846**	**0.1653**	**0.2075**

Table 3: Average performance of fifty runs of the models on different train and test splits with 30, 50 and 100 percent of training data. The average scores reported are the MAE.

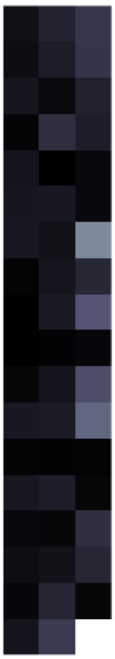


Figure 1: Visualization of the RMTL task outlier model when trained on all the 181 instances of training data. Cells with darker shades are closer to zero. Cells with lighter shades are closer to one. Columns with only black entries are considered inlier tasks (domains). From left to right, columns correspond to News, TED and IT domains. The first 17 rows correspond to the features used to train the model and the last row in corresponds to the bias term.

For all three domains, a general trend is that MTL RMTL is the method that reaches the lowest MAE when compared to all the other models. Given the difference among the domains, it is very likely that MTL Dirty and MTL SparseTrace suffer from the negative transfer problem (the assumption that all tasks are similar does not hold). MTL RMTL is the only method among the methods presented here that copes with negative transfer among tasks. The significance tests indicate that MTL RMTL improvements are statistically significant with respect to all baselines depending on the range of training data used to compute the test.

- For **TED**, the Friedman test rejects the null hypothesis with $p = 4.62^{-5}$. Post-hoc analysis indicates that there are differences statistically significant between MTL RMTL and all the three baselines with $p \leq 0.002$.

- For **News**, the Friedman test measures significant differences with $p = 1.14^{-9}$ and the post-hoc analysis indicates that MTL RMTL is statistically significant with respect to SVR in-domain and SVR Pooling with $p = 0.002$ for varying amounts of training data from 10 to 100%. As can be seen in Figure 2, MTL RMTL starts with a very high MAE using 10% of the data (approximately 0.21 MAE) but improves dramatically with 20% of the data. Calculating the significance test with 20 to 100% of training data, MTL RMTL is significantly better than all baselines with $p \leq 2.89^{-10}$.

- For **IT**, in a similar situation to the News domain, RMTL is significantly better than all baselines

trained on 30% to 100% of the training data (Friedman test's $p = 2.86^{-4}$ and post-hoc analysis' $p \leq 3.73^{-7}$).

Another observed trend is that the MTL models benefit from increasing amounts of training data. MTL RMTL has an improvement in performance of 5.13% for TED, 4% for News and 1.85% for IT when trained on 100% of the training data in comparison with the model trained on 30% of training data.

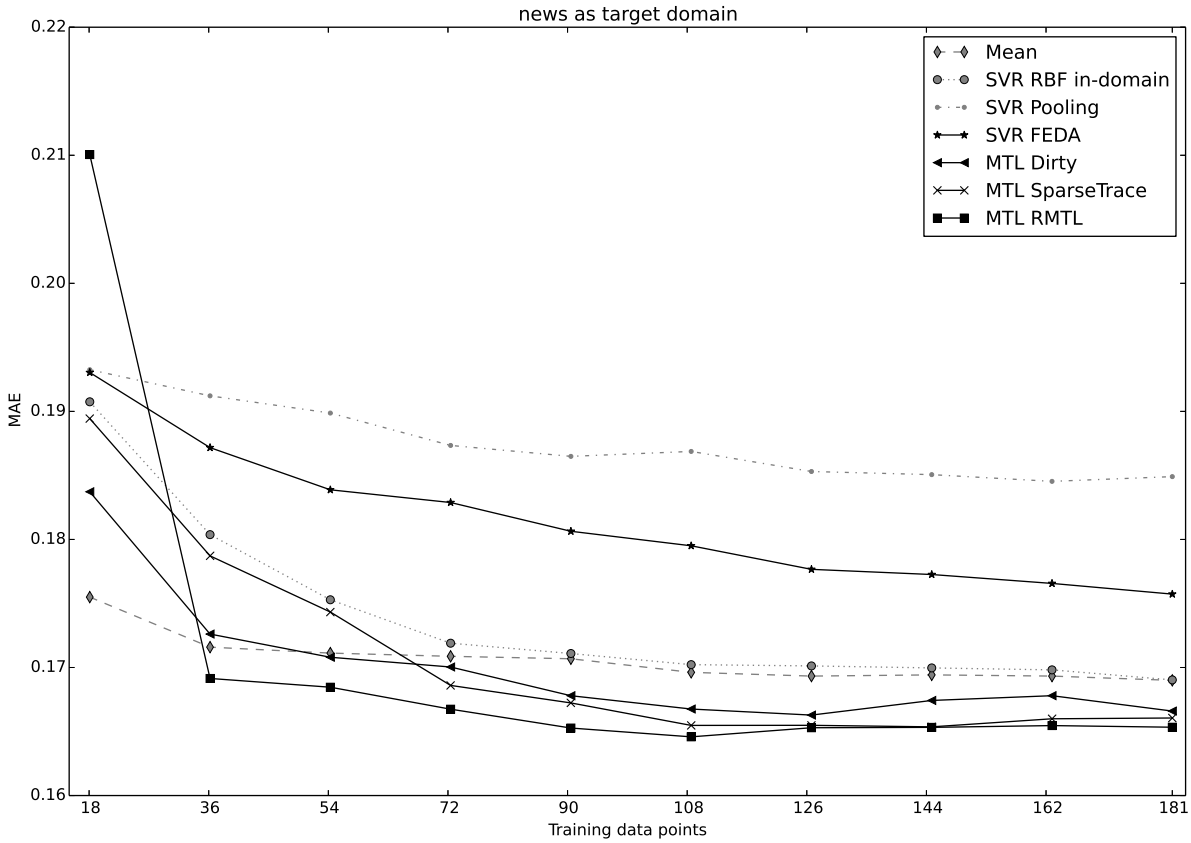


Figure 2: Learning curves for the News domain.

The results for the IT domain are in line with the in-domain experiments in which we observed that IT is a more challenging domain in comparison to TED and News. The MAE of IT is always higher than for the other domains on in-domain and MTL experiments. Another evidence of this is the model learned by the RMTL method when using all training data and run on one of the 50 training/test splits. A graphic representation of the RMTL outlier task detection component (described in Section 3) is shown in Figure 1.

From left to right, each column represents News, TED, and IT domains, respectively, while each row is the instantiation of a feature in the corresponding task. Columns with non-black entries represent outlier tasks. The highest number of entries with lighter shades is in the third column, IT. Several features in this task are considered outliers with respect to the same features in the other tasks. Consequently, the learning method takes the weights into consideration to a greater extent when learned with the outlier model for the IT domain. Entries with the lightest shades in the IT domain correspond to the features marked with ◯ in Section 4.2. These outlier features are directly affected by the length of the sentences on which they are computed (source or target) given that the number of tokens influences the final value of the feature. This outcome goes in the same direction of our analysis of the three domains (Section 4.1) that indicates a very different vocabulary size and average sentence length for IT when compared to the other two domains.

To a lesser extent than IT, News and TED domains also present a few lighter-shaded entries in the outlier component (1st and 2nd column). This suggests that MTL RMTL was capable of transfering information among the domains in a more efficient way than the other MTL methods analyzed.

Overall the experiments presented show encouraging results in the direction of coping with QE data coming from different domains/genres, translated by different MT systems and post-edited by different translators. Results show that even in such difficult conditions, the methods investigated are capable of outperforming competitive baselines based on non-linear models on different domains. As a rationale, models that consider not only similarity between the domains but also deal with some sort of dissimilarity should be considered. This is the case of the best performing method, MTL RMTL, which identifies outlier tasks in order to avoid negative transfer among tasks.

6 Conclusion

In this work we presented an investigation of methods that overcome limitations presented by current MT QE state-of-the-art systems when applied to real world conditions. In such scenarios (e.g. CAT environment) the requirements are two-fold: (i) learning in the presence of different train/test feature and label distributions and across different domains/genres, and (ii) the capability of learning with scarce training data. In our experiments, we explored transfer learning methods, in particular multitask learning, and we showed that such methods can cope with the needs of real-world scenarios.

We showed that multitask learning methods are capable to learn robust models for three different domains that perform better than three strong baselines trained on the same amount of data. The methods explored here benefit from increasing amounts of training data but also perform well when operating with very limited amounts of data. We believe that the results obtained in this first exploration of model adaptation for the problem can encourage the MT QE community to shift the focus from controlled scenarios to more applicable, real-world contexts that require more robust methods.

Acknowledgements

This work has been partially supported by the EC-funded project MateCat (ICT-2011.4.2-287688).

References

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2007. Multi-task feature learning. In *Advances in neural information processing systems*, volume 19.

Daniel Beck, Kashif Shah, Trevor Cohn, and Lucia Specia. 2013. SHEF-Lite: When less is more for translation quality estimation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 337–342.

James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, 13:281–305.

Joseph John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2003. Confidence estimation for machine translation. In *20th COLING*, pages 315–321.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 120–128, Morristown, NJ, USA. Association for Computational Linguistics.

Ondej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Eighth Workshop on Statistical Machine Translation*, pages 1–44.

Christian Buck. 2012. Black Box Features for the WMT 2012 Quality Estimation Shared Task. In *Proceedings of the 7th Workshop on Statistical Machine Translation*, pages 91–95.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the 7th Workshop on Statistical Machine Translation*, pages 10–51, Montr{é}al, Canada, June. Association for Computational Linguistics.

Rich Caruana. 1997. Multitask Learning. *Machine learning*, 28(1):41–75.

Jianhui Chen, Jiayu Zhou, and Jieping Ye. 2011. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11*, page 42, New York, New York, USA. ACM Press.

Jianhui Chen, Ji Liu, and Jieping Ye. 2012. Learning incoherent sparse and low-rank patterns from multiple tasks. *ACM Transactions on Knowledge Discovery from Data*, 5(4):22, February.

Trevor Cohn and Lucia Specia. 2013. Modelling Annotator Bias with Multi-task Gaussian Processes: An application to Machine Translation Quality Estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 32–42.

Hal Daumé III. 2007. Frustratingly Easy Domain Ddaptation. In *Conference of the Association for Computational Linguistics (ACL)*.

José G. C. de Souza, Christian Buck, Marco Turchi, and Matteo Negri. 2013a. FBK-UEdin participation to the WMT13 Quality Estimation shared-task. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 352–358.

José G.C. de Souza, Miquel Esplà-Gomis, Marco Turchi, and Matteo Negri. 2013b. Exploiting qualitative information from automatic word alignment for cross-lingual nlp tasks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 771–776, Sofia, Bulgaria, August. Association for Computational Linguistics.

Janez Demšar. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine Learning Research*, 7:1–30, December.

Milton Friedman. 1937. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, 32(200):675–701.

Milton Friedman. 1940. A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *The Annals of Mathematical Statistics*, 11(1):86–92.

Salvador Garcia and Francisco Herrera. 2008. An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons. *Journal of Machine Learning Research*, 9:2677–2694.

Christian Hardmeier, Joakim Nivre, and Jorg Tiedemann. 2012. Tree Kernels for Machine Translation Quality Estimation. In *Proceedings of the 7th Workshop on Statistical Machine Translation*, number 2011, pages 109–113.

Sture Holm. 1979. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics*, 6(2):pp. 65–70.

Ali Jalali, PD Ravikumar, S Sanghavi, and C Ruan. 2010. A Dirty Model for Multi-task Learning. In *Advances in Neural Information Processing Systems (NIPS) 23*.

Jing Jiang and Chengxiang Zhai. 2007. Instance Weighting for Domain Adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, number June, pages 264–271.

Jing Jiang. 2009. Multi-Task Transfer Learning for Weakly-Supervised Relation Extraction. In *ACL '09 Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL*, number August, pages 1012–1020.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zenz, Chris Dyer, Ondej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL 2007 Demo and Poster Sessions*, number June, pages 177–180.

Yashar Mehdad, Matteo Negri, and Marcello Federico. 2012. Match without a Referee : Evaluating MT Adequacy without Reference Translations. In *Proceedings of the 7th Workshop on Statistical Machine Translation*, pages 171–180.

Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Mathieu Brucher, Mathieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Raphael Rubino, Jennifer Foster, Joachim Wagner, Johann Roturier, Rasul Samad Zadeh Kaljahi, and Fred Hollowood. 2012. DCU-Symantec Submission for the WMT 2012 Quality Estimation Task. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 138–144, Montr{é}al, Canada, June. Association for Computational Linguistics.

Raphael Rubino, José G. C. de Souza, Jennifer Foster, and Lucia Specia. 2013. Topic Models for Translation Quality Estimation for Gisting Purposes. In *Machine Translation Summit (MT Summit) XIV*, pages 295–302.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Association for Machine Translation in the Americas*.

Radu Soricut and A Echihabi. 2010. Trustrank: Inducing trust in automatic translations via ranking. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, number July, pages 612–621.

Radu Soricut, Nguyen Bach, and Ziyuan Wang. 2012. The SDL Language Weaver Systems in the WMT12 Quality Estimation Shared Task. In *Proceedings of the 7th Workshop on Statistical Machine Translation*, pages 145–151.

José G. C. de Souza, Jesús González-Rubio, Christian Buck, Marco Turchi, and Matteo Negri. 2014. FBK-UPV-UEdin participation in the WMT14 Quality Estimation shared-task. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, Baltimore, MD, USA, June.

Lucia Specia, Marco Turchi, Nello Cristianini, Nicola Cancedda, and Marc Dymetman. 2009. Estimating the Sentence-Level Quality of Machine Translation Systems. In *Proceedings of the 13th Annual Conference of the EAMT*, number May, pages 28–35.

Lucia Specia, Dhwaj Raj, and Marco Turchi. 2010. Machine translation evaluation versus quality estimation. *Machine Translation*, 24(1):39–50, May.

Lucia Specia, Stafford Street, Regent Court, and Mariano Felice. 2012. Linguistic Features for Quality Estimation. In *Proceedings of the 7th Workshop on Statistical Machine Translation*, pages 96–103.

Lucia Specia. 2011. Exploiting objective annotations for measuring translation post-editing effort. In *Proceedings of the European Association for Machine Translation*, number May, pages 73–80.

Marco Turchi, Josef Steinberger, and Lucia Specia. 2012. Relevance ranking for translated texts. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, number May, pages 153–160.

Marco Turchi, Matteo Negri, and Marcello Federico. 2013. Coping with the Subjectivity of Human Judgements in MT Quality Estimation. In *Proceedings of the 8th Workshop on Statistical Machine Translation (WMT'13)*, Sofia, Bulgaria, August.

Marco Turchi, Antonios Anastasopoulos, José G. C. de Souza, and Matteo Negri. 2014. Adaptive Quality Estimation for Machine Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.

Guillaume Wisniewski, Anil Kumar Singh, Natalia Segal, and François Yvon. 2013. Design and Analysis of a Large Corpus of Post-Edited Translations: Quality Estimation, Failure Analysis and the Variability of Post-Edition. In *Machine Translation Summit XIV*, pages 117–124.

Jiayu Zhou, Jianhui Chen, and Jieping Ye. 2012. MALSAR: Multi-tAsk Learning via StructurAl Regularization.

Investigating the Usefulness of Generalized Word Representations in SMT

Nadir Durrani
University of Edinburgh
dnadir@inf.ed.ac.uk

Philipp Koehn
University of Edinburgh
pkoehn@inf.ed.ac.uk

Helmut Schmid **Alexander Fraser**
Ludwig Maximilian University Munich
fraser,schmid@cis.uni-muenchen.de

Abstract

We investigate the use of generalized representations (POS, morphological analysis and word clusters) in phrase-based models and the N-gram-based *Operation Sequence Model (OSM)*. Our integration enables these models to learn richer lexical and reordering patterns, consider wider contextual information and generalize better in sparse data conditions. When interpolating generalized OSM models on the standard IWSLT and WMT tasks we observed improvements of up to +1.35 on the English-to-German task and +0.63 for the German-to-English task. Using automatically generated word classes in standard phrase-based models and the OSM models yields an average improvement of +0.80 across 8 language pairs on the IWSLT shared task.

1 Introduction

The increasing availability of digital text has galvanized the use of empirical methods in many fields including Machine Translation. Given bilingual text, it is now possible to automatically learn translation rules that required years of effort previously. Bilingual data, however, is abundantly available for only a handful of language pairs. The problem of reliably estimating statistical models for translation becomes more of a challenge under sparse data conditions especially when translating into morphologically rich or syntactically divergent languages. The former becomes challenging due to lexical sparsity and the latter suffers from sparsity in learning underlying reordering patterns. The last decade of research in Statistical Machine Translation has witnessed many attempts to integrate linguistic analysis into SMT models, to address the challenges of (i) translating into morphologically rich language languages, (ii) modeling syntactic divergence across languages for better generalization in sparse data conditions.

The integration of the *Operation Sequence Model* into phrase-based paradigm (Durrani et al., 2013a; Durrani et al., 2013b) improved the reordering capability and addressed the problem of the phrasal independence assumption in the phrase-based models. The OSM model integrates translation and reordering into a single generative story. By jointly considering translation and reordering context across phrasal boundaries, the OSM model considers much richer conditioning than phrasal translation and lexicalized reordering models. However, due to data sparsity the model often falls back to very small context sizes. We address this problem by learning operation sequences over generalized representations such as POS and Morph tags. This enables us to learn richer translation and reordering patterns that can generalize better in sparse data conditions. The model benefits from wider contextual information as we show empirically in our results.

We investigate two methods to combine generalized OSM models with the lexically driven OSM model and experimented on German-English translation tasks. Our best system that uses a linear combination of different OSM models gives significant improvements over a competitive baseline system. An improvement of up to +1.35 was observed on the English-to-German and up to +0.63 BLEU points on the German-to-English task over a factored augmented baseline system (Koehn and Hoang, 2007).

POS taggers and morphological analyzers, however, are not available for many resource poor languages. In the second half of the paper we investigate whether annotating the data with automatic word

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 421–432, Dublin, Ireland, August 23-29 2014.

clusters helps improve the performance. Word clustering is similar to POS-tagging/Morphological annotation except that it also captures interesting syntactic and lexical semantics, for example countries and languages are grouped in separate clusters, animate objects are differentiated from inanimate objects, colors are grouped in a separate cluster etc. Word clusters, however, deterministically map each word type to a unique[1] cluster, unlike POS/Morph tagging, and therefore might be less useful for disambiguation. We use the `mkcls` utility in GIZA (Och and Ney, 2003) to cluster source and target vocabularies into classes and will therefore refer to automatic classes as Och clusters/classes in this paper.

We first use Och classes as an additional factor in phrase-based translation model, along with a target LM model over cluster-ids to improve the baseline system. We then additionally use the OSM model over cluster-ids. Our experiments include translation from English to Dutch, French, Italian, Polish, Portuguese, Russian, Spanish, Slovenian and Turkish on IWSLT shared task data. Our results show an average improvement of +0.80, ranging from +0.41 to +2.02. Compared to the improved baseline system obtained by using Och classes as a factor in phrase-based translation models, adding an OSM model over cluster-ids improved performance in four (French, Spanish, Dutch and Slovenian) out of eight cases. In other cases performance stayed constant or dropped slightly. We also used POS annotations for three tasks, namely translating from English into French, Spanish and Dutch to compare the performance of the two different kinds of generalizations. Surprisingly, using Och classes always performed better than using POS annotations. The rest of the paper is organized as follows. Section 2 gives an account on related work. Section 3 discusses the factor-based OSM model. Section 4 presents the experimental setup and the results. Section 5 concludes the paper.

2 Related Work

Previous work on integrating linguistic knowledge into SMT models can be broken into two groups. The first group focuses on using linguistic knowledge to improve reordering between syntactically different languages. A second group focuses on translating into morphologically rich languages.

Initial efforts to use linguistic annotation focused on rearranging source sentences to be in the target order. Xia and McCord (2004) proposed a method to automatically learn rewrite rules to preorder source sentences. Collins et al. (2005) and Popović and Ney (2006) proposed methods for reordering the source using a small set of handcrafted rules. Crego and Mariño (2007) use syntactic trees to derive rewrite rules. Hoang and Koehn (2009) used POS tags to create templates for surface word translation to create longer phrase translation. A whole new paradigm of using syntactic annotation to address long range reorderings has emerged following Galley et al. (2006), Zollmann and Venugopal (2006), Chiang (2007) etc. Crego and Yvon (2010) and Niehues et al. (2011) used a Tuple Sequence Model (TSM) over POS tags in an N-gram-based search to improve mid-range reorderings. Our work is similar to them except that OSM model is substantially different from the TSM model as it integrates both the translation and reordering mechanisms into a combined model. Therefore both translation and reordering decisions can benefit from richer generalized representations.

A second group of work addresses the problem of translating into morphologically richer languages. The idea of translating to stems and then inflecting the stems in a separate step has been studied by Toutanova et al. (2008), de Gispert and Mariño (2008), Fraser et al. (2012), Chahuneau et al. (2013) and others. Koehn and Hoang (2007) proposed to integrate different levels of linguistic information as factors into the phrase-based translation model. Yeniterzi and Oflazer (2010) used source syntactic structures as additional complex tag factors for English-to-Turkish phrase-based machine translation. Green and DeNero (2012) proposed a target-side, class-based agreement model to handle morpho-syntactic agreement errors when translating from English-to-Arabic. El Kholy and Habash (2012) tested three models to find out which features are best handled by modeling them as a part of translation, and which ones are better predicted through generation, also in the English-to-Arabic task. Several researchers attempted to use word lattices to handle generalized representation (Dyer et al., 2008; Hardmeier et al., 2010; Wuebker and Ney, 2012). Automatically clustering the training data into word classes in order to obtain smoother

[1]We are referring to hard clustering here. Soft clustering is intractable as it requires a marginalization over all possible classes when calculating the n-gram probabilities.

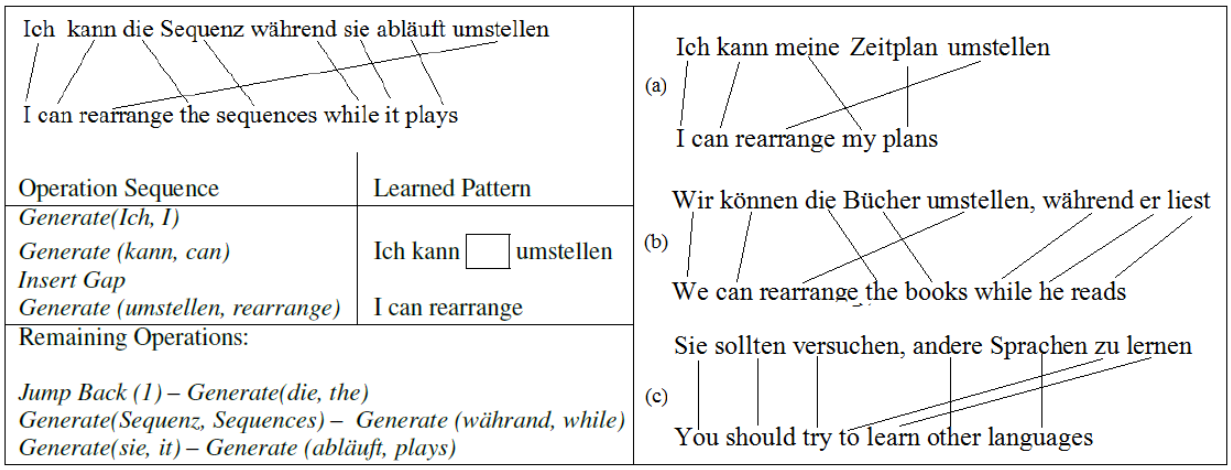


Figure 1: Operation Sequence Model – Training Sentence with Generation and Test Sentences

distributions and better generalizations has been a widely known and applied technique in natural language processing. Training based on word classes has been previously explored by various researchers. Cherry (2013) addressed data sparsity in lexicalized reordering models by using sparse features based on word classes. Other parallel attempts on using word-class models include Wuebker et al. (2013), Chahuneau et al. (2013) and Bisazza and Monz (2014).

More recent research has started to set apart from the conventional maximum likelihood estimates toward neural network-based models that use continuous space representation (Schwenk, 2012; Le et al., 2012; Hu et al., 2014; Gao et al., 2014). Although these methods have achieved impressive improvements, traditional models continue to dominate the field due to their simplicity and low computational complexity. How much of the improvement will be retained when scaling these models to all available data instead of a limited amount will be interesting.

3 Operation Sequence Model

The Operation Sequence Model (Durrani et al., 2011) is an instance of the N-gram based SMT framework (Casacuberta and Vidal, 2004; Mariño et al., 2006). It represents the translation process through a sequence of operations. An operation can be to simultaneously generate source or target words or to perform reordering. Reordering is carried out through jump and gap operations. The model is different from its ancestors in that it strongly integrates translation and reordering into a single generative story in which translation decisions can influence and get impacted by the reordering decisions and vice versa. Given a bilingual sentence pair $< F, E >$ and its alignment A, a sequence of operations $o_1, o_2 \ldots, o_J$ is generated deterministically through a conversion algorithm. The model is learned by learning Markov chains over these sequences and is formally defined as:

$$p_{osm}(F, E, A) = \prod_{j=1}^{J} p(o_j | o_{j-n+1}, ..., o_{j-1})$$

Figure 1 shows an example of an aligned bilingual sentence pair and the corresponding operation sequence used to generate it. There is a 1-1 correspondence between a sentence pair and its operation sequence. We thus get a unique sequence for every bilingual sentence pair given the alignment.

3.1 Motivation

Due to data sparsity it is impossible to observe all possible reordering patterns with all possible lexical choices in translation operations. The lexically driven OSM model therefore often backs off to very small context sizes. Coming back to the training example in Figure 1. The useful reordering pattern

learned through this example is:

Ich kann □ umstellen → I can rearrange

which is memorized through the operation sequence:

Generate(Ich, I) – Generate(kann, can) – Insert Gap – Generate(umstellen, rearrange)

It can generalize to the test sentence shown in Figure 1(a). However, it fails to generalize to the sentences in Figure 1(b) and (c) although the underlying reordering pattern is the same. The second part of the German verb complex usually appears at the end of a clause or a sentence and needs to be moved in order to produce the correct English word order. However, due to data sparsity such a combination of lexical decisions and reordering decisions may not be observed during training. The model would therefore fail to generalize in such circumstances. This problem can be addressed by learning a generalized form of the same reordering rule. By annotating the corpus with word classes such as POS tags, we obtain the reordering pattern:

PPER VMFIN □ VVINF → PP MD VB

memorized through the operation sequence:

Generate (PPER,PP) – Generate (VMFIN,MD) – Insert Gap – Generate (VVINF,VB)

This rule generalizes to all the test sentences in Figure 1. Since the OSM model strongly couples translation and reordering, the probability of each translation or reordering operation depends on the n previous translation/reordering decisions. The generalization of the model by replacing words with POS tags allows the model to consider a wider syntactic context, thus improving lexical decisions and the reordering capability of the model. Using different kinds of word classes, we can also control the type of abstraction. Using lemmas for example, we can map different forms of the verb "können – can" (kann, kannst, konnte) to a single class. Och clusters can provide different levels of granularity.

3.2 Models

Given that we can learn OSM models over different word representations, the question then is how to combine the lexically driven OSM model with an OSM model based on a generalized word representation. The simplest approach is to treat each OSM model as a separate feature in the log-linear framework, thus summing up the weighted log probabilities. The effect of this is similar to an *And* operation. A translation is considered good if both, the word-based OSM and the POS-based OSM models indicate that it is a good translation. However, an *Or* operation might be more desirable in some scenarios. The operation *Generate (trotz, in spite of)* should be ranked high although the POS-based operation *Generate(APPR, IN IN IN)* is improbable. Similarly, the generalized operation sequence:

Insert Gap – Generate (ADJ, JJ) – Jump Back – Generate (NOM, NN)

that captures the swapping of noun and adjective in French-English, should be ranked higher even though *noir* (black) never appeared after *cheval* (horse) during training and the sequence:

Insert Gap – Generate (noir, black) – Jump Back – Generate (cheval , horse)

is never observed. Instead of using both the models, a single model that could switch between different generalized OSMs during translation and choose the one which gives the best prediction in each situation, can be used. In order to achieve this effect, we formulated a second model that interpolates the lexically driven OSM model with its generalized variants. However, we can only

interpolate two models that predict the same representation. The lexically driven OSM predicts the surface forms whereas the POS-based OSM predicts POS translations. To make the two comparable, we multiply the POS-based OSM probability with the probability of the lexical operation given the POS operation. More specifically the probability of the generalized model gm can be defined as:

$$p_{gm}(o_j|o_{j-n+1}^{j-1}) = p_{osm_{pos}}(o'_j|o'^{j-1}_{j-n+1})\ p(o_j|o'_j) \tag{1}$$

where $p_{osm_{pos}}$ is the operation sequence model learned over POS tags and $p(o_j|o'_j)$ is the probability of the lexical operation given the POS-based operation. It is 1 for all reordering operations. We assume here that for each lexical operation o_j a corresponding POS-based operation o'_j is uniquely determined. With $p_{osm_{sur}} = p_{osm_{sur}}(o_j|o_{j-n+1}^{j-1})$ (lexically driven OSM model) and $p_{gm} = p_{gm}(o_j|o_{j-n+1}^{j-1})$ (generalized OSM model as described above), the overall probability of the new model p_{osm} is defined as:

$$p_{osm} = \alpha p_{osm_{sur}} + (1-\alpha)p_{gm} \tag{2}$$

Such an interpolation is expensive in the discriminative training. It would require a sub-tuning routine inside of tuning, a main loop to train all the features including the OSM model and an inner loop to distribute the weight assigned to OSM model among lexically driven and POS-based OSM models. We therefore just take the larger one of the two model values and add a POS-based translation penalty ϕ. The value of this penalty is the number of times that the POS-based operation was chosen when translating a sentence. This penalty acts similarly as the prior α above. Using this formulation, the model could therefore be redefined as:

$$p_{osm} = \begin{cases} p_{osm_{sur}} & \text{if } p_{osm_{sur}} \geq e^{\lambda} p_{gm} \\ e^{\lambda} p_{gm} & \text{otherwise} \end{cases} \tag{3}$$

where λ is the weight for the POS driven translation penalty ϕ. This allows the optimizer to control whether it prefers the lexically driven or the POS-driven OSM model. By setting a very low weight λ the optimizer can force the translator to always choose lexically driven OSM. This formulation can be extended to multiple generalized OSM models based on e.g. POS tags, morphological tags, or word clusters. Equation 2 can be rewritten as follows:

$$p_{osm} = \alpha_1 p_{osm_{sur}} + \sum_{i=2}^{n} \alpha_i p_{gm_i} \tag{4}$$

with $\sum_{i=1}^{n} \alpha_i = 1$ and p_{gm_i} defined analogous to Equation 1.

Setting $p_{gm_1} = p_{osm_{sur}}$ and $\lambda_1 = 0$, we can again simplify Equation 4 by taking the maximum to:

$$p_{osm} = \max_{i=1}^{n} e^{\lambda_i} p_{gm_i} \tag{5}$$

We use a translation penalty ϕ_i for each generalized model and tune its weight λ_i along with the weights of other features. We will refer to this model as $\mathbf{Model_{or}}$ in this paper and the commonly used log-linear interpolation of the features as $\mathbf{Model_{and}}$. The intuition behind $\mathbf{Model_{or}}$ is that we back-off to generalized representations only when the lexically driven model doesn't provide enough contextual evidence. The downside of this approach, however, is that unlike $\mathbf{Model_{and}}$, it cannot distribute weights over multiple features and solely relies on a single model.

4 Evaluation

Data: We ran experiments with data made available for the translation task of the IWSLT-13 (Cettolo et al., 2013): International Workshop on Spoken Language Translation[2] and WMT-13 (Bojar et al., 2013): Eighth Workshop on Statistical Machine Translation.[3] The sizes of bitext used for the estimation of translation and monolingual language models are reported in Table 1.

We used LoPar (Schmid, 2000) to obtain morphological analysis and POS annotation of German and MXPOST (Ratnaparkhi, 1998), a maximum entropy model for English POS tags. For other language pairs we used TreeTagger (Schmid, 1994).

[2] http://www.iwslt2013.org/

[3] http://www.statmt.org/wmt13/

Pair	Parallel	Monolingual	Pair	Parallel	Monolingual	Pair	Parallel	Monolingual
de–en	≈4.6 M	≈287.3 M	en–de	≈4.6 M	≈59.5 M	en–fr	≈5.5 M	≈69 M
en–es	≈4.1 M	≈59.6 M	en–nl	≈2.1 M	≈21.7 M	en–ru	≈1.15 M	≈21 M
en–pt	≈1.0 M	≈2.3 M	en–pl	≈0.77 M	≈0.8 M	en–sl	≈0.63 M	≈0.65 M
en–tr	≈0.13 M	≈0.14 M						

Table 1: Number of Sentences (in Millions) used for Training

Model	$\mathbf{iwslt_{10}}$	$\mathbf{wmt_{13}}$	$\mathbf{iwslt_{10}}$	$\mathbf{wmt_{13}}$
	English-to-German		German-to-English	
Baseline	23.56	20.38	31.46	27.27
$\mathbf{M_{and_{(pos,pos)}}}$	**23.93** Δ+0.37	20.61 Δ+0.23	**31.91** Δ+0.45	27.55 Δ+0.28
$\mathbf{M_{and_{(pos,morph)}}}$	**24.62** Δ+1.06	**20.88** Δ+0.50	**32.09** Δ+0.63	**27.62** Δ+0.35
$\mathbf{M_{and_{(all)}}}$	**24.91** Δ+1.35	**20.93** Δ+0.55	**32.00** Δ+0.54	**27.71** Δ+0.44
$\mathbf{M_{or_{(pos,pos)}}}$	23.61 Δ+0.05	20.24 Δ-0.14	31.55 Δ+0.09	27.32 Δ+0.05
$\mathbf{M_{or_{(pos,morph)}}}$	23.83 Δ+0.27	20.44 Δ+0.08	31.58 Δ+0.12	27.20 Δ-0.07
$\mathbf{M_{or_{(all)}}}$	23.88 Δ+0.32	20.55 Δ+0.17	31.40 Δ-0.06	27.15 Δ-0.12

Table 2: Evaluating Generalized OSM Models for German-English pairs – Bold: Statistically Significant (Koehn, 2004) w.r.t Baseline

Baseline System: We trained a Moses system (Koehn et al., 2007), replicating the settings described in (Birch et al., 2013) developed for the 2013 Workshop on Spoken Language Translation. The features included: a maximum sentence length of 80, grow-diag-final-and symmetrization of GIZA++ alignments, an interpolated Kneser-Ney smoothed 5-gram language model with KenLM (Heafield, 2011) used at runtime, a lexically-driven 5-gram operation sequence model (Durrani et al., 2013b) with 4 additional supportive features: 2 gap-based penalties, 1 distance-based feature and 1 deletion penalty, lexicalized reordering, sparse lexical and domain features (Hasler et al., 2012), a distortion limit of 6, 100-best translation options, Minimum Bayes Risk decoding (Kumar and Byrne, 2004), Cube Pruning (Huang and Chiang, 2007), with a stack-size of 1000 during tuning and 5000 during test and the no-reordering-over-punctuation heuristic. We used the compact phrase table representation by Junczys-Dowmunt (2012). For our German-to-English experiments, we used compound splitting (Koehn and Knight, 2003). German-to-English and English-to-German baseline systems also used POS and morphological target sequence models built on the in-domain subset of the parallel corpus using Kneser-Ney smoothed 7-gram models and as additional factors in phrase translation models (Koehn and Hoang, 2007). We used an unsupervised transliteration model (Durrani et al., 2014) to transliterate OOV words when translating into Russian.

Tuning and Test: The systems were tuned on the dev2010 dataset and evaluated on the test2010-2013 datasets made available for the IWSLT-13 workshop. We performed a secondary set of experiments for German-English pairs using tuning and test sets made available for the WMT-13 workshop. We concatenated the news-test sets 2008 and 2009 to obtain a large dev-set of 4576 sentences. Evaluation was performed on the news-test set 2013 which contains 3000 sentences. Tuning was performed using the k-best batch MIRA algorithm (Cherry and Foster, 2012) with at most 25 iterations. We use BLEU (Papineni et al., 2002) as a metric to evaluate our results.

Results I – Using Linguistic Annotation: We trained 5-gram OSM models over different representations and added these to the baseline system. First we evaluated $\mathbf{Model_{and}}$ ($\mathbf{M_{and}}$) which uses a MIRA tuned linear combination of different OSM models versus $\mathbf{Model_{or}}$ ($\mathbf{M_{or}}$) which computes only one OSM model but allows the generator to switch between different OSM models built on various generalized forms. Table 2 shows results from running experiments on German-English pairs. We found that the simpler model $\mathbf{Model_{and}}$ outperforms $\mathbf{Model_{or}}$ in all the experiments. $\mathbf{Model_{or}}$ does not give significant improvements over the baseline system and shows an occasional drop. This result is contrary to the expectation formulated in Section 3.2. We speculate that the optimizer faces problems to train this kind of model, because it cannot take into account that the selected OSM model can change when the weight parameter is modified. It assumes that the feature stays constant. In our formulation the same

derivation can occur with different feature scores in different decoding runs and the optimizer is unable to handle this. Our speculation is based on the observation of λ_ϕ, the weight of feature ϕ which allows the translator to switch between different OSM models. The value of λ_ϕ was not stable across different iterations and different experiments.

$\mathbf{Model_{and}}$ consistently improves the baseline. Adding an OSM model over [pos, morph] (source:pos, target:morph) combination gave the best results, giving a statistically significant gain of +1.06 on the $\mathbf{iwslt_{10}}$ test-set and +0.50 on the $\mathbf{wmt_{13}}$ test-set. Using an OSM model over a [pos,pos] combination also showed improvements, however, not as much as using morphological tags. Morphological tags provide richer information for disambiguation when translating into German. Note that the baseline system also used a target sequence model over morphological tags. Nevertheless using an OSM [pos,morph] model still gives significant improvements which shows that learning a joint model over source and target units is more fruitful than only considering target-side information. Using both the models together gave best results for English-to-German giving a further improvement of +0.29 on the $\mathbf{iwslt_{10}}$ task but no real gain on the $\mathbf{wmt_{13}}$ task. Using morphological tags also produced the best results for the German-to-English pair, giving a statistically significant gain of +0.63 on $\mathbf{iwslt_{10}}$ and +0.35 on $\mathbf{wmt_{13}}$. Using both the models together did not give any further significant improvements. The results changed by +0.10 and -0.09 on the $\mathbf{wmt_{13}}$ and $\mathbf{iwslt_{10}}$ test-sets respectively.

Results-II – Using Och Classes: In our secondary experiments we tested the effect of using Och clusters. The overall goal was to study whether using unsupervised word classes can serve the same purpose as POS tags and to compare the two methods of annotating the data. We obtained Och clusters using the `mkcls` utility (Och, 1999) in GIZA++ (Och and Ney, 2003). This is generally run during the alignment process where data is divided into 50 classes to estimate IBM Model-4. Chahuneau et al. (2013) found mapping data to 600 Och clusters useful, so we used this as well. We additionally experimented with using 200 and 1000 classes. We integrated Och clusters as additional factors[4] when training the phrase-translation models and used a monolingual n-gram model over cluster-ids built on the target-side of the in-domain corpus. Then we added a 5-gram OSM model over cluster-ids. We replace surface forms with their cluster-ids in source and target corpus and convert it to operation sequences, that jointly generate source and target cluster-ids. We only used $\mathbf{Model_{and}}$ for these experiments when adding an OSM model over cluster-ids.

	$\mathbf{B_0}$	**50**	**200**	**600**	**1000**	**POS**	**50**	**200**	**600**	**1000**	**POS**
		Target Sequence Model over Word Clusters					**Operation Sequence Model over Word Clusters**				
en – fr	33.17	33.30	33.40	33.05	33.05	33.14	**33.76**	33.74	33.58	33.75	33.03
en – es	34.14	34.33	34.58	34.46	33.96	33.91	**34.73**	34.62	34.60	34.55	34.35
en – nl	26.51	26.67	26.15	26.31	26.47	26.55	**26.91**	26.52	26.61	26.49	26.62
en – ru	13.12	13.34	13.51	13.53	**13.97**	–	13.61	13.66	13.80	13.63	–
en – sl	17.98	18.67	18.55	17.67	17.97	–	18.64	**18.91**	18.17	17.98	–
en – pt	30.80	31.62	32.21	32.40	32.44	–	31.77	32.44	32.34	31.90	–
en – pl	9.74	9.90	10.11	10.05	10.43	–	10.06	10.19	10.24	10.14	–
en – tr	7.18	7.43	7.45	7.50	7.50	–	7.26	7.28	7.51	7.54	–

Table 3: Evaluating Phrase-based and N-gram-based Translation Models over Och Clusters

Table 3 shows results from using models based on cluster-ids. The left side of the table evaluate the use of adding a target sequence model over cluster-ids using a factored-based translation model. Results improved consistently in all resource poor languages (pt, pl, tr) giving significant improvements in most of the cases. Mixed results were obtained for the pairs with a reasonable amount of parallel data (fr, es, nl), showing an occasional drop in performance. However, improvements can be found for all the language pairs.

[4]Note that adding cluster-ids in factored models alone has no impact in this scenario, as we are using hard clustering (each word deterministically maps onto a unique cluster-id). In a joint source-target factored model which is what we are using, it will result in an identical distribution as the baseline system.

In the right half of the table we tested whether additionally using an OSM model built over cluster-ids, on top of a phrase-based system that uses cluster-ids as factor and target language model, improves the performance any further. Consistent improvements were seen in Spanish and French. Better systems were produced in the case of French, Spanish, Dutch and Slovenian. No improvements were observed for Turkish and Portuguese whereas the performance got worse in Polish and Russian.

Using 50 classes consistently improved the baseline. Different numbers of clusters provide different levels of abstraction and granularity. We also tried using OSM models over different numbers of clusters simultaneously for English-to-Spanish, English-to-French and English-to-Dutch pairs in an effort to explore whether using different numbers of clusters to classify data provides different information. A slight gain was observed for EN-ES as the best system improved from 34.73 to 34.95. No further gains were observed for the other two pairs.

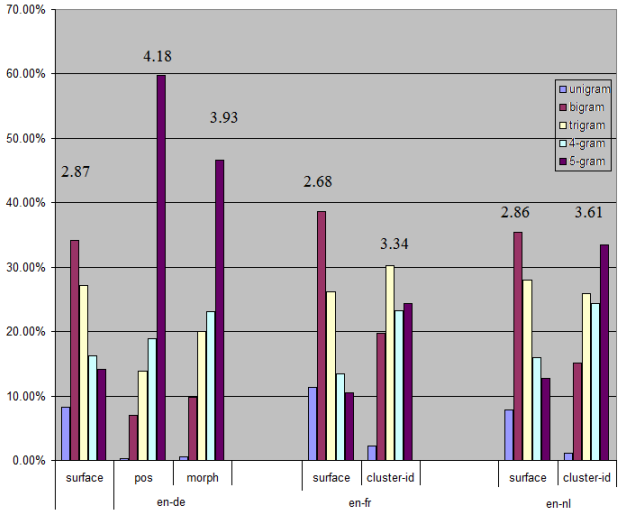
We also used POS annotation as a factor instead of Och clusters in French, Spanish and Dutch. See the POS columns of Table 3. Using POS as an additional factor, did not improve over the baseline performance. A significant drop was seen in the case of English-to-Spanish. Using a POS-based OSM on top of the POS-based phrase-model did not help either except for Spanish where results got improved by +0.44 over its phrase-based variant that used a POS factor. However, using Och clusters produced better results in all three cases. We speculate that the reason for this result is that Och clusters are more evenly distributed as compared to POS tags where the distribution is biased toward noun class and secondly Och clusters are optimized for language modeling. Also each word is deterministically mapped to a single class but can have multiple POS tags. The latter thus causes a sparser translation model. Finally Table 4 shows the comparison of results on $\mathbf{iwslt_{11-13}}$ by running baseline $\mathbf{B_0}$ and best systems $\mathbf{B_x}$ in Tables 3.

	$\mathbf{iwslt_{11}}$		$\mathbf{iwslt_{12}}$		$\mathbf{iwslt_{13}}$		**Avg**		
	$\mathbf{B_0}$	$\mathbf{B_x}$	$\mathbf{B_0}$	$\mathbf{B_x}$	$\mathbf{B_0}$	$\mathbf{B_x}$	$\mathbf{B_0}$	$\mathbf{B_x}$	$\mathbf{\Delta}$
en – fr	39.84	40.63	40.50	41.24	–	–	40.24	40.94	+0.70
en – es	32.89	33.24	26.45	26.81	34.01	34.73	31.12	31.60	+0.48
en – nl	30.01	30.31	26.40	26.72	24.96	25.57	27.12	27.53	+0.41
en – ru	14.93	15.91	13.01	13.53	15.65	16.4	14.53	15.28	+0.75
en – sl	–	–	11.34	12.40	12.85	13.73	12.09	13.10	+1.01
en – pt	31.61	33.62	33.24	34.91	30.83	33.24	31.89	33.92	+2.02
en – pl	12.73	13.13	9.52	10.50	11.30	11.54	11.18	11.72	+0.53
en – tr	7.01	7.42	6.99	7.43	6.21	6.84	6.74	7.23	+0.49
Avg	24.15	24.89	20.93	21.69	19.40	20.29	21.49	22.29	+0.80

Table 4: Evaluating on Test Sets $\mathbf{iwslt_{11-13}}$ – $\mathbf{B_0}$ = Baseline System, $\mathbf{B_x}$ = Best Systems in Tables 2

Analysis: In a post-evaluation analysis we confirmed whether using generalized OSM models actually consider a wider contextual window than its lexically driven variant. The graph shown in Figure 2 shows average context size considered (on top of each set of bars) and percentages of 1-5 gram matches by different OSM models. The results show that the probability of an operation is conditioned on less than a trigram in the OSM model over surface forms. In comparison OSM models over POS, morph or cluster-ids consider a window of roughly 4 previous operations thus considering more contextual information. The percentage of 5-gram matches increases from 15.5% to 59.2% using POS-based OSM model and up to 45.6% in morph-based OSM model, the number of unigram matches are decreased from 8.30% to less than 1% in both the models. Similar observation is made for the OSM models over clusters where 5-gram matches improve from 12% to 30% on average, showing the ability of the generalized models to use richer conditioning thus improving the translation quality.

We also analyzed what kind of words are clustered together using Och classes and found that clusters capture both syntax and lexical semantics. Figure 2 (b) shows several useful clusters to exhibit this. We also saw negative examples where words from different classes are clustered together. “Boy”, “Girl” and “Man” for example were clustered into a single class but “Woman” in another. Similarly “Grey” and “Orange” were grouped together with animated objects.



Colors: Red, Black, Green, Blue …
Animals: Spider, Elephant, Cat, Rabbit …
Cities: London, Paris, Chicago …
Languages: English, Arabic, Farsi …
Royal Family: Queen, King, Prince …
Food: Sugar, Salt, Pepper, Milk …
Stars: Sagittarius, Aquarius, Virgo …
Verbs: Improve, Accelerate, Improvise …
Adjectives: Stupid, Ugly, Funny, Rusty …

Colors: Rouge, Noir, Vert, Bleu…
Animals: Chat, Lapin, Serpent, Crocodile …
Cities: Londres, Paris, Lahore …
Languages: Urdu, Hindi, Tamil …
Royal Family: Roi, Prince, Reine …
Food: Sucre, Sel, Poivre, Lait…
Stars: Sagittarius, Aquarius, Capricorn …
Verbs: Améliorer, Accélérer, Improviser …
Adjectives: Stupide, Drôle …

Figure 2: (a) Average Size of N-grams Used in Different OSM Models and Percentages of 1-5 Gram Matches in Three Language Pairs (b) Different Word Clusters using 50 Classes

5 Conclusion

In this paper we investigated the usefulness of integrating word classes in phrase-based models and Operation Sequence N-gram models. We explored two models of interpolating generalized OSM models and tested variations on the standard IWSLT and WMT tasks. Our results showed that the simpler more commonly used method of integrating the models in the log-linear framework worked best. We showed that by learning OSM models over generalized POS and morphological representations, we were able to build richer models that outperformed state-of-the-art baseline systems. Statistically significant gains of up to +1.35 and +0.63 were observed in English-to-German and German-to-English tasks. We also made use of Och classes as additional factors in phrase translation and language models. These were tested translating from English to 8 different languages which includes a mixture of morphologically rich (French, Spanish and Russian, Dutch, and Turkish) and sparse data (Portuguese, Polish, Slovenian and Turkish) languages. Our results show that using clusters was helpful in all of the cases. Using the OSM model over word-clusters additionally improved the performance further. Our results show an average improvement of +0.80, ranging from +0.41 to +2.02. Our EN-FR systems were ranked third (on tst2013) and second (on tst2011-tst2012) in IWSLT-13 translation task following EU-Bridge (Freitag et al., 2013) which used our output for system combination. The code to train class-based models has been made available to the research community via the Moses toolkit. See Advanced Features[5] in the Moses Decoder for details.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful feedback and suggestions. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreements n° 287658 (EU-Bridge) and n° 287688 (MateCat). Alexander Fraser was funded by Deutsche Forschungsgemeinschaft grant Models of Morphosyntax for Statistical Machine Translation. Helmut Schmid was supported by Deutsche Forschungsgemeinschaft grant SFB 732. This publication only reflects the authors' views.

References

Alexandra Birch, Nadir Durrani, and Philipp Koehn. 2013. Edinburgh SLT and MT System Description for the IWSLT 2013 Evaluation. In *Proceedings of the 10th International Workshop on Spoken Language Translation*,

[5] http://www.statmt.org/moses/?n=Moses.AdvancedFeatures

pages 40–48, Heidelberg, Germany, December.

Arianna Bisazza and Christof Monz. 2014. Class-Based Language Modeling for Translating into Morphologically Rich Languages. In *Proceedings of the 25th Annual Conference on Computational Linguistics (COLING)*, Dublin, Ireland, August.

Ondrej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Eighth Workshop on Statistical Machine Translation*, WMT-2013, pages 1–44, Sofia, Bulgaria.

Francisco Casacuberta and Enrique Vidal. 2004. Machine Translation with Inferred Stochastic Finite-State Transducers. *Computational Linguistics*, 30:205–225.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2013. Report on the 10th IWSLT Evaluation Campaign. *Proceedings of the International Workshop on Spoken Language Translation, Heidelberg, Germany*.

Victor Chahuneau, Eva Schlinger, Noah A. Smith, and Chris Dyer. 2013. Translating into Morphologically Rich Languages with Synthetic Phrases. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

Colin Cherry and George Foster. 2012. Batch Tuning Strategies for Statistical Machine Translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436, Montréal, Canada, June. Association for Computational Linguistics.

Colin Cherry. 2013. Improved Reordering for Phrase-Based Translation using Sparse Features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 22–31, Atlanta, Georgia, June. Association for Computational Linguistics.

David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause Restructuring for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 531–540, Ann Arbor, MI.

Josep M. Crego and José B. Mariño. 2007. Syntax-Enhanced N-gram-Based SMT. In *Proceedings of the 11th Machine Translation Summit*, MT Summit XI, pages 111–118.

Josep M. Crego and François Yvon. 2010. Improving Reordering with Linguistically Informed Bilingual N-Grams. In *Coling 2010: Posters*, pages 197–205, Beijing, China, August. Coling 2010 Organizing Committee.

Adrià de Gispert and José B. Mariño. 2008. On the Impact of Morphology in English to Spanish statistical MT. *Speech Communication*, 50(11-12):1034–1046.

Nadir Durrani, Helmut Schmid, and Alexander Fraser. 2011. A Joint Sequence Translation Model with Integrated Reordering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1045–1054, Portland, Oregon, USA, June.

Nadir Durrani, Alexander Fraser, and Helmut Schmid. 2013a. Model With Minimal Translation Units, But Decode With Phrases. In *The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Nadir Durrani, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013b. Can Markov Models Over Minimal Translation Units Help Phrase-Based SMT? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, August. Association for Computational Linguistics.

Nadir Durrani, Hassan Sajjad, Hieu Hoang, and Philipp Koehn. 2014. Integrating an Unsupervised Transliteration Model into Statistical Machine Translation. In *Proceedings of the 15th Conference of the European Chapter of the ACL (EACL 2014)*, Gothenburg, Sweden, April. Association for Computational Linguistics.

Christopher Dyer, Smaranda Muresan, and Philip Resnik. 2008. Generalizing Word Lattice Translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1012–1020, Columbus, OH, USA. The Association for Computer Linguistics.

Ahmed El Kholy and Nizar Habash. 2012. Translate, Predict or Generate: Modeling Rich Morphology in Statistical Machine Translation. volume 12.

Alexander Fraser, Marion Weller, Aoife Cahill, and Fabienne Cap. 2012. Modeling Inflection and Word-Formation in SMT. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 664–674, Avignon, France, April. Association for Computational Linguistics.

Markus Freitag, Stephan Peitz, Joern Wuebker, Hermann Ney, Nadir Durrani, Matthias Huck, Philipp Koehn, Thanh-Le Ha, Jan Niehues, Mohammed Mediani, Teresa Herrmann, Alex Waibel, Nicola Bertoldi, Mauro Cettolo, and Marcello Federico. 2013. EU-BRIDGE MT: Text Translation of Talks in the EU-BRIDGE Project. In *International Workshop on Spoken Language Translation*, Heidelberg, Germany, December.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable Inference and Training of Context-Rich Syntactic Translation Models. In *Proceedings of COLING-ACL*, pages 961–968, Sydney, Australia. Association for Computational Linguistics.

Jianfeng Gao, Xiaodong He, Scott Wen-tau Yih, and Li Deng. 2014. Learning Continuous Phrase Representations for Translation Modeling. In *Proceedings of the Association for Computational Linguistics*, Baltimore, MD, USA, June.

Spence Green and John DeNero. 2012. A Class-Based Agreement Model for Generating Accurately Inflected Translations. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–155, Jeju Island, Korea, July. Association for Computational Linguistics.

Christian Hardmeier, Arianna Bisazza, and Marcello Federico. 2010. FBK at WMT 2010: Word Lattices for Morphological Reduction and Chunk-Based Reordering. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 88–92, Uppsala, Sweden, July. Association for Computational Linguistics.

Eva Hasler, Barry Haddow, and Philipp Koehn. 2012. Sparse Lexicalised features and Topic Adaptation for SMT. In *Proceedings of the seventh International Workshop on Spoken Language Translation (IWSLT)*, pages 268–275.

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom, July.

Hieu Hoang and Philipp Koehn. 2009. Improving Mid-Range Re-Ordering Using Templates of Factors. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 372–379, Athens, Greece, March. Association for Computational Linguistics.

Yuening Hu, Michael Auli, Qin Gao, and Jianfeng Gao. 2014. Minimum Translation Modeling with Recurrent Neural Networks. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 20–29, Gothenburg, Sweden, April. Association for Computational Linguistics.

Liang Huang and David Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic, June. Association for Computational Linguistics.

Marcin Junczys-Dowmunt. 2012. Phrasal Rank-Encoding: Exploiting Phrase Redundancy and Translational Relations for Phrase Table Compression. *The Prague Bulletin of Mathematical Linguistics*, 98:63–74.

Philipp Koehn and Hieu Hoang. 2007. Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, Prague, Czech Republic, June. Association for Computational Linguistics.

Philipp Koehn and Kevin Knight. 2003. Empirical methods for compound splitting. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 187–193, Morristown, NJ.

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

Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July.

Shankar Kumar and William J. Byrne. 2004. Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *HLT-NAACL*, pages 169–176.

Hai-Son Le, Alexandre Allauzen, and François Yvon. 2012. Continuous Space Translation Models with Neural Networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–48, Montréal, Canada, June. Association for Computational Linguistics.

José B. Mariño, Rafael E. Banchs, Josep M. Crego, Adrià de Gispert, Patrik Lambert, José A. R. Fonollosa, and Marta R. Costa-jussà. 2006. N-gram-Based Machine Translation. *Computational Linguistics*, 32(4):527–549.

Jan Niehues, Teresa Herrmann, Stephan Vogel, and Alex Waibel. 2011. Wider Context by Using Bilingual Language Models in Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 198–206, Edinburgh, Scotland, July. Association for Computational Linguistics.

Franz J. Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Franz J. Och. 1999. An Efficient Method for Determining Bilingual Word Classes. In *Processings of EACL*, pages 71–76, Bergen, Norway.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Morristown, NJ, USA.

Maja Popović and Hermann Ney. 2006. POS-based Word Reorderings for Statistical Machine Translation. In *International Conference on Language Resources and Evaluation*, pages 1278–1283, Genoa, Italy.

Adwait Ratnaparkhi. 1998. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

Helmut Schmid. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK.

Helmut Schmid. 2000. Lopar: Design and implementation. Bericht des sonderforschungsbereiches "sprachtheoretische grundlagen fr die computerlinguistik", Institute for Computational Linguistics, University of Stuttgart.

Holger Schwenk. 2012. Continuous Space Translation Models for Phrase-Based Statistical Machine Translation. In *Proceedings of COLING 2012: Posters*, pages 1071–1080, Mumbai, India, December. The COLING 2012 Organizing Committee.

Kristina Toutanova, Hisami Suzuki, and Achim Ruopp. 2008. Applying Morphology Generation Models to Machine Translation. In *Proceedings of ACL-08: HLT*, pages 514–522, Columbus, Ohio, June. Association for Computational Linguistics.

Joern Wuebker and Hermann Ney. 2012. Phrase Model Training for Statistical Machine Translation with Word Lattices of Preprocessing Alternatives. In *NAACL 2012 Seventh Workshop on Statistical Machine Translation*, pages 450–459, Montreal, Canada, June. Association for Computational Linguistics.

Joern Wuebker, Stephan Peitz, Felix Rietig, and Hermann Ney. 2013. Improving Statistical Machine Translation with Word Class Models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1377–1381, Seattle, Washington, USA, October. Association for Computational Linguistics.

Fei Xia and Michael McCord. 2004. Improving a Statistical MT System with Automatically Learned Rewrite Patterns. In *Proceedings of Coling 2004*, pages 508–514, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Reyyan Yeniterzi and Kemal Oflazer. 2010. Syntax-to-Morphology Mapping in Factored Phrase-Based Statistical Machine Translation from English to Turkish. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 454–464, Uppsala, Sweden, July. Association for Computational Linguistics.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 138–141, New York City, June. Association for Computational Linguistics.

Confusion Network for Arabic Name Disambiguation and Transliteration in Statistical Machine Translation

Young-Suk Lee
IBM T. J. Watson Research Center
1101 Kitchawan Road
Yorktown Heights, NY 10598, USA
ysuklee@us.ibm.com

Abstract

Arabic words are often ambiguous between name and non-name interpretations, frequently leading to incorrect name translations. We present a technique to disambiguate and transliterate names even if name interpretations do not exist or have relatively low probability distributions in the parallel training corpus. The key idea comprises named entity classing at the preprocessing step, decoding of a simple confusion network created from the name class label and the input word at the statistical machine translation step, and transliteration of names at the post-processing step. Human evaluations indicate that the proposed technique leads to a statistically significant translation quality improvement of highly ambiguous evaluation data sets without degrading the translation quality of a data set with very few names.

1 Introduction

Arabic person and location names are often ambiguous between name and non-name interpretations, as noted in (Hermjakob et al., 2008; Zayed et al., 2013). (1) and (2) illustrate such ambiguities for Iraqi Arabic, where the ambiguous names and their translations are in bold-face and the Buckwalter transliteration of Arabic is provided in parentheses:[1]

(1) a. انـي ساكـن بـشقـة يـم الـمـدرسة ب**خضراء**
(Any sAkn b$qp ym Almdrsp b**xDrA'**)
I live in an apartment near the school in **Khadraa**

b. مـصبـوغة **خضراء**
(mSbwgp **xDrA'**)
It is painted **green**

(2) a. شيـقـدر يـقـول إلـك **صبـاح**
($yqdr yqwl Alk **SbAH**)
What can **Sabah** tell you?

b. **صبـاح** الـخيـر إنـت أكـيـد نـقـيـب حسام
(**SbAH** Alxyr Ant Akyd nqyb HsAm)
Good **morning** you must be captain Hosam

In this paper, we propose a technique for disambiguating and transliterating Arabic names in an end-to-end statistical machine translation system. The key idea lies in name classing at the preprocessing step, decoding of a simple confusion network created from the class label *$name*, and the input word at the machine translation step, and transliteration of names by a character-based phrase transliteration model at the post-processing step.

While Bertoldi et al. (2007) propose confusion network decoding to handle multiple speech recognition outputs for phrase translation and Dyer et al. (2008) generalize lattice decoding algorithm to

[1]Arabic should be read from right to left, and the Buckwalter transliteration should be read from left to right.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 433–443, Dublin, Ireland, August 23-29 2014.

tackle word segmentation ambiguities for hierarchical phrase-based translation, the current proposal is the first to deploy a confusion network for name disambiguation and translation. The character-based phrase transliteration model captures the asymmetry between Arabic and English vowel systems by treating English vowels as spontaneous words attachable to the neighboring target phrases for phrase (a sequence of characters) acquisition.

Confusion network decoding enables the system to choose between name and other translations of the source word on the basis of the decoding cost computed from all of the decoder feature functions which incorporate name tag scores into translation model scores. Probabilistic choice between name versus non-name interpretations makes the technique robust to name classing errors, without stipulating the frequency threshold of the names to be transliterated in order to avoid translation quality degradation (Hermjakob et al., 2008; Li et al., 2013). A tight integration of named entity detection and classing into the machine translation system, coupled with a generative approach to name transliteration, enables the system to produce reliable name translations even when name interpretations do not exist or have relatively low distributions in the parallel corpus, distinguishing the current proposal from Hermjakob et al. (2008).

In Section 2, we give an overview of the translation system. In Section 3, we discuss the model training and confusion network decoding. In Section 4, we detail name transliteration model. We present the experimental results in Section 5. We discuss related work in Section 6 and conclude the paper in Section 7.

2 End-to-end Translation System Overview

Arabic name disambiguation and transliteration techniques are incorporated into an end-to-end phrase translation system (Och and Ney, 2002; Koehn et al., 2003; Koehn et al., 2007). Our phrase translation system builds on Tillmann (2003) for translation model training and an in-house implementation of Ney and Tillmann (2003) for beam search phrase decoding.

Iraqi Arabic to English end-to-end phrase translation systems are trained on DARPA TransTac data (Hewavitharana et al., 2013), comprising 766,410 sentence pairs (~6.8 million morpheme tokens in Arabic, ~7.3 million word tokens in English; ~55k unique vocabulary in Arabic and ~35k unique vocabulary in English). The data consist of sub-corpora of several domains including military combined operations, medical, humanitarian aid, disaster relief, etc., and have been created primarily for speech-to-speech translations. The process flow of Arabic to English translation incorporating the proposed technique is shown in Figure 1. The components relevant to name disambiguation and transliteration are in bold face.

Given the input sentence (3), the spelling normalizer normalizes آني to اني.

(3) آني ساكن بشقة يم المدرسة بخضراء
(|ny sAkn b$qp ym Almdrsp bxDrA')

The morpheme segmenter segments a word into morphemes (Lee et al., 2003; Lee, 2004; Habash and Sadat, 2006) as in (4), where # indicates that the morpheme is a prefix.

(4) اني ساكن ب# شقة يم ال# مدرسة ب# خضراء
(Any sAkn b# $qp ym Al# mdrsp b# xDrA')

Part-of-speech tagging is applied to the morphemes, identifying a name with the tag NOUN_PROP. The input word tagged as NOUN_PROP is classified as name, denoted by the label *$name* in (5).

(5) اني ساكن ب# شقة يم ال# مدرسة ب# $name_(خضراء)
(Any sAkn b# $qp ym Al# mdrsp b# $name_(xDrA'))

The token $name_(خضراء) is decomposed into the class label *$name* and the source word بخضراء, creating a simple confusion network for decoding. The beam search phrase decoder computes the translation costs for all possible input phrases including the phrase pair "*$name* | *$name*",[2] using all of

[2] The source phrase *$name* translates to the target phrase *$name*.

the decoder feature functions. Assuming that the translation cost for *$name* being translated into *$name* is the lowest, the decoder produces the translation (6), where the name classed source word خضراء retains its Arabic spelling .

(6) I live in an apartment near the school in خضراء

The Arabic word خضراء in (6) is transliterated into *khadraa* by the NAME/OOV transliteration module. And the system produces the final translation output (7).

(7) I live in an apartment near the school in khadraa

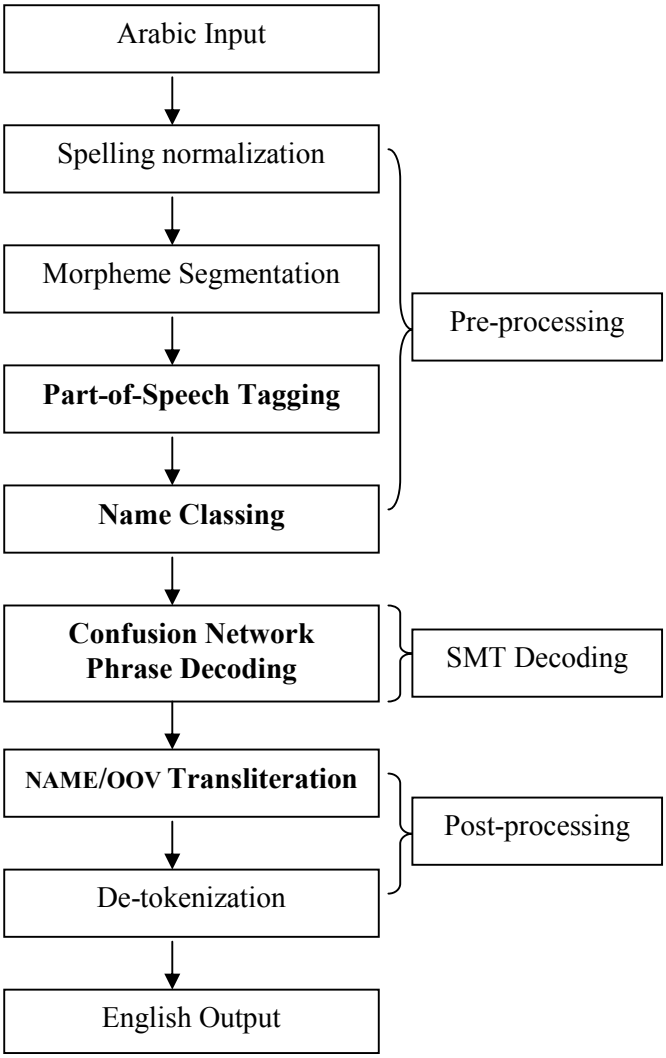


Figure 1. Process Flow of Arabic to English Phrase Translation Decoding

We use an in-house implementation of the maximum entropy part-of-speech tagger described in Adwait (1996) for name classing. The part-of-speech tagger is trained on the combination of LDC-released Arabic Treebank data containing about 3 million morpheme tokens from MSA (modern standard Arabic) and in-house annotated TransTac Iraqi Arabic data containing about 63k morpheme tokens.

F-score of the tagger on proper noun tags, NOUN_PROP, is about 93% on 2,044 MSA name tokens derived from Arabic Treebank: Part 3 v 3.2 (LDC2010T08), and about 81.4% on 2,631 Iraqi Arabic name tokens derived from the DARPA TransTac corpus.

3 Model Training and Confusion Network Decoding

We train translation and language models with name classing to obtain proper translation and language model probabilities of the class label *$name*. We extend the baseline phrase beam search decoder to handle a relatively simple confusion network (CN hereafter) and incorporate the name part-of-speech tagging scores into the decoder feature functions.

3.1 Translation Model

For any name classed input word, *$name_(orig)* in (5), we would like to have the name translation, *$name* → *$name*, always available in addition to other translations of the input word obtainable from the parallel training corpus.

In order to estimate *$name* distributions without obfuscating the distributions of other training vocabulary, we apply name classing only to words that occur less than 3 times in the training corpus and part-of-speech tagged with NOUN_PROP. The reasons are three-fold: 1) we need to keep all non-name translations of the training vocabulary, 2) typical low frequency words include names and typos, 3) even with *$name* classing on low frequency words only, the overall *$name* count is high enough for a robust probability estimation.

After name classing of words occurring less than 3 times, *$name* occurs 6,944 times (122th most frequent token) in Arabic and 9,707 times (108th most frequent token) in English. We train both phrase translation and distortion models on the name classed parallel corpus. Note that the frequency restriction applies only to model training. During decoding, any word labeled with *$name* may be name transliterated regardless of its frequency in the training corpus, differentiating the current technique from (Li et al., 2013).

3.2 Language Models

To properly capture the name and non-name ambiguities, we interpolate two types language models: 1) 5-gram language model trained on the English side of the parallel corpus without name classing (LM1), 2) 5-gram language model trained on the English side of the parallel corpus and additional monolingual corpora with name classing (LM2).

Each language model is smoothed with modified Kneser-Ney (Chen and Goodman, 1998). The two sets of language models are interpolated, as in (8), where α is set to 0.1. We find the optimal interpolation weight on the basis of BLEU scores of the development test data set containing about 30k word tokens in Arabic and about 43k word tokens in English.

(8) $\alpha \cdot LM1 + (1-\alpha) \cdot LM2$

3.3 Confusion Network Decoding

The confusion network containing the class label *$name* and the source word is handled by an extension of the baseline phrase decoder. The baseline decoder utilizes 11 feature functions including those in (9)[3] through (14), where $\bar{f}$ denotes the source phrase and $\bar{e}$, the target phrase, and ***s***, the source sentence, ***t***, the target sentence and *a*, a word alignment. We use the in-house implementation of the simplex algorithm in Zhao et al. (2009) for decoder parameter optimization.

(9) Direct phrase translation model for $pr(\bar{e} \mid \bar{f})$
(10) Distortion models(Al-Onaizan and Papineni, 2006)
(11) Mixture language models

[3] We do not use $pr(\bar{f} \mid \bar{e})$

(12) Lexical weights $p_w(\bar{f}|\bar{e},a)$ & $p_w(\bar{e}|\bar{f},a)$, cf. (Koehn et al., 2003)
(13) Lexical weights p_w (**t**|**s**,a) & p_w (**s**|**t**,a)
(14) Word and phrase penalties (Zens and Ney, 2004)

Lexical weight $p_w(\bar{e}|\bar{f},a)$ in (12) is computed according to (15), where $j = 1, \ldots, n$ source word positions and $i = 1,\ldots, m$ target word positions within a phrase, N = source phrase length, $w(e|f)$ = the lexical probability distribution:[4]

(15) $$p_w(\bar{e}|\bar{f},a) = (\prod_{j=1}^{n} w_{\max(j,i)\in a}(e_i|f_j)) / N$$

Lexical weight $p_w(\mathbf{t}|\mathbf{s},a)$ in (13) is computed according to (16), where K = number of phrases in the input sentence, $k = k^{th}$ phrase, and $pr_{w\_k}(\bar{e}|\bar{f},a) = p_{w\_k}(\bar{e}|\bar{f},a)$ without normalization by the source phrase length N.

(16) $$p_w(\mathbf{t}|\mathbf{s},a) = \prod_{k=1}^{K} pr_{w\_k}(\bar{e}|\bar{f},a)$$

We augment the baseline decoder in two ways: First, we incorporate the maximum entropy part-of-speech tagging scores of names into the translation scores in (9), (12) and (13). We simply add the name part-of-speech tag cost, i.e. –log probability, to the translation model costs. Second, the decoder can activate more than one edge from one source word position to another, as shown in Figure 2.[5] The name classed input is split into two tokens *$name* and *xDrA'*, leading to two separate decoding paths. The choice between the two paths depends on the overall decoding cost of each path, computed from all of the decoder feature functions.

Since the decoding path to *$name* is always available when the input word is classed as *$name* at the pre-processing step, the technique can discover the name interpretation of an input word even if the name interpretation is absent in the parallel training corpus. Even when the input word occurs as a name in the training corpus but has a lower name translation probability than non-name translations in the baseline phrase table, it can be correctly translated into a name as long as the word is labeled as *$name* and the decoder feature functions support the *$name* path in the given context. When a non-name token is mistakenly labeled as *$name*, the confusion network decoder can recover from the mistake if the non-name path receives a lower decoding cost than the *$name* path.[6] If the input token is name classed and the correct name translation also exists in the baseline phrase table with a high probability, either path will lead to the correct translation, and the decoder chooses the path with the lower translation cost.

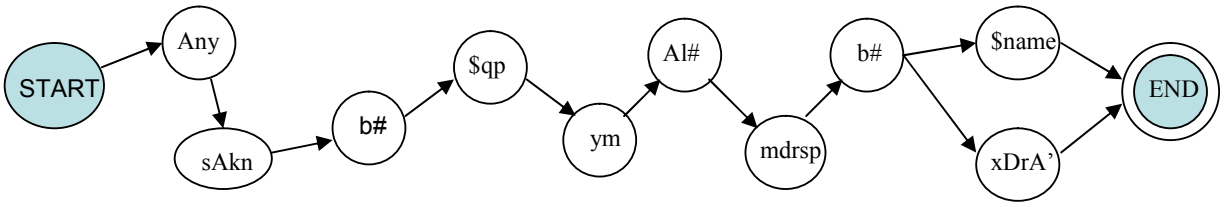


Figure 2. Confusion Network Decoding Paths for Name Classed Input

[4] Estimated in the manner described in Koehn et al. (2003).
[5] Arabic is represented by Buckwalter transliteration scheme.
[6] The decoding scores are computed as cost on the basis of –log likelihood of various component models. And therefore, a smaller decoding cost indicates a higher translation quality.

4 Character-Based Phrase Transliteration Models

All instances of un-translated input words, which include names and OOVs, are transliterated in the post-processing step. Character-based phrase transliteration models are trained on 9,737 unique name pairs. 965 name pairs are obtained from a name lexicon and the remaining 8,772 name pairs are automatically derived from the parallel training corpus as follows: 1) Take each side of the parallel corpus, i.e. Iraqi Arabic or English. 2) Mark names manually or automatically. 3) Apply word alignment to the name-marked parallel corpus in both directions. 4) Extract name pairs aligned in both directions. For name marking, we used the manual mark-up that was provided in the original data.

5-gram character language models are trained on about 120k entries of names in English. In addition to about 9.7k names from the English side of the parallel names, about 110k entries are collected from wiki pages, English Gigaword 5th Edition (LDC2011T07), and various name lexicons.

4.1 Phrase Extraction with English Vowels as Spontaneous Words

Short vowels are optional in written Arabic, whereas all vowels have to be obligatorily specified in English for a word to be valid (Stalls and Knight, 1998; Al-Onaizan and Knight, 2002b). We model the asymmetrical nature of vowels between the two languages by treating all instances of unaligned English vowels – *a, e, i, o, u* – as spontaneous words which can be attached to the left or to the right of an aligned English character for phrase extractions. An example GIZA++ (Och and Ney, 2003) character alignment is shown in Figure 3. Arabic name is written left to right to illustrate the monotonicity of the alignments.

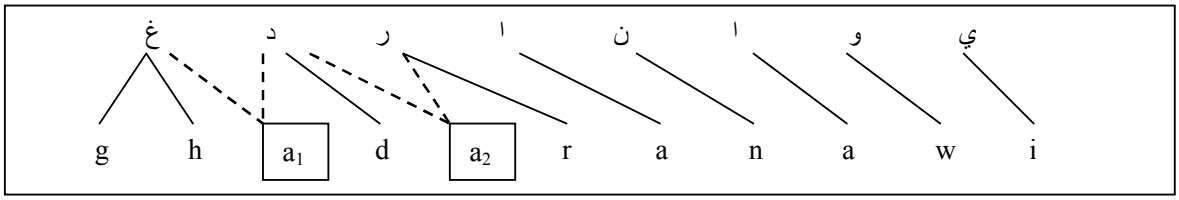


Figure 3. Automatic Character Alignment between Arabic and English names

In Figure 3, solid lines indicate the automatic machine alignments. English vowels in rectangular boxes indicate null alignments by the aligner. The dotted lines indicate the potential attachment sites of the unaligned vowels for phrase extractions. The first instance of unaligned *a* (denoted by a_1) may be a part of the phrases containing the preceding consonant sequence *g h*, or the following consonant *d*. The second instance of unaligned *a* (denoted by a_2) may be a part of the phrases containing the preceding consonant *d* or the following consonant *r*.[7]

4.2 Experiments

We use exact match accuracy[8] to evaluate transliteration qualities. Systems are tested on 500 unique name pairs including 52 names unseen in the training corpus. Experimental results are shown in Table 1.[9] Note that using English vowels as spontaneous words dramatically improves the accuracy from 21.6% to 89.2%.

Decoding is carried out by the baseline phrase decoder discussed in Section 3.3, using the same decoder feature functions except for the distortion models. Using only phrase translation and language model probabilities for decoding results in 74.4% accuracy on SYSTEM4, much lower than 90% accuracy with all decoder feature functions. The same language model is used for all experiments. For the end-to-

[7] Attachment of unaligned English vowels takes place after phrase extractions and should be distinguished from a heuristic alignment of unaligned English vowels to Arabic characters before phrase extractions.
[8] A transliteration is correct if and only if it exactly matches the truth, i.e. gold standard.
[9] GIZA++ word aligner is trained with 5 iterations of IBM MODEL 1, 5 iterations of HMM, 5 iterations of IBM MODEL 3 and 5 iterations of IBM MODEL 4. HMM word aligner (Vogel et al., 1996) is trained with 15 iterations of IBM MODEL 1 and 6 iterations of HMM.

end translation quality evaluations in Section 5, we use SYSTEM4. Exact match accuracy of SYSTEM4 on the 52 unseen name pairs is 46%.

Systems	Character Alignments	Symmetrization[10]	Target spontaneous words	Accuracy
SYSTEM1	GIZA++	Union	None	21.6%
SYSTEM2	HMM	Refined	None	86.8%
SYSTEM3	GIZA++	Union	All English vowels: *a, e, i, o, u*	89.2%
SYSTEM4	GIZA++ & HMM	Union	All English vowels: *a, e, i, o, u*	90.0%

Table 1. Name transliteration accuracy on 500 names according to various phrase extraction techniques

5 End-to-end Translation System Experimental Results

End-to-end translation quality experiments are carried out on 3 evaluation data sets shown in Table 2. TransTac.eval has a low out-of-vocabulary (OOV) and a low name ratios, and has been used as the test data for system development among DARPA BOLT–C[11] program partners. TransTac.oov has a high OOV and a high name ratios, and has been created in-house for OOV detection system developement. TransTac.name has a low OOV and a high name ratios, and was used for the TransTac 2008 name translation evaluations.

Evaluation Data Sets	TransTac.eval	TransTac.oov	TransTac.name
sentence count	3,138	344	79
token count	36,895	3,053	514
OOV ratio	0.4%	4.7%	0.6%
name ratio	~0.5%	~11.3%	~15.4%

Table 2. Translation Quality Evaluation Data Statistics

5.1 Systems, Metrics and Results

End-to-end translation system evaluation results are shown in Table 3. Bold-faced and italicized scores indicate that the system's translation quality is statistically significantly better than all other systems with over 95% confidence, i.e. two-tailed P value < 0.05 in paired t-tests.

Metrics	EvalSets / Systems	TransTac.eval	TransTac.oov	TransTac.name	TransTac.name_spnorm
Uncased BLEU (4-gram & 1 ref)	baseline	33.35	30.72	35.03	37.39
	OOVTranslit	33.35	31.93	35.03	37.54
	name_t	32.94	31.81	32.97	40.15
	CN	33.35	32.60	32.19	40.97
HUMAN (6-point scale)	baseline	3.16	1.45	3.19	3.19
	OOVTranslit	3.22	2.88	3.36	3.36
	name_t	2.16	2.79	3.58	3.58
	CN	3.20	***3.09***	***3.86***	***3.86***

Table 3. Translation Quality Evaluation Result

The system *baseline* is trained without name classing and decoded by the baseline decoder without name classing. The system OOVTranslit is trained and decoded the same way as the baseline except that all instances of un-translated OOVs are transliterated at the post-processing step. The system *name_t* is

[10] Bi-directional word alignment symmetrization methods, as defined in Och and Ney (2003), include union, intersection and refined.

[11] BOLT stands for Broad Operational Language Translation and BOLT-C focuses on speech-to-speech translation with dialog management.

trained without name classing and decoded by the baseline decoder with name classing.[12] The system *CN* is trained with name classing and decoded by the CN decoder with name classing.[13]

We evaluate the systems, using automatic BLEU (Papineni et al., 2002), and 6-point scale human evaluations. Lowercased BLEU scores are computed with 1 reference translation up to 4-grams. Scoring criteria for human evaluations are as follows. **0**: exceptionally poor; **1**: poor; **2**: not good enough; **3**: good enough; **4**: very good; **5**: excellent. Human evaluations are conducted on a subset of the automatic evaluation data containing names.[14] We exclude the input sentences for which all systems produce the same translation output. This leaves 201 sentences from TransTac.eval, 197 sentences from TransTac.oov, 64 sentences from TransTac.name.

5.2 Result Analysis

We observe that human evaluation scores are relatively consistent with BLEU scores on two data sets, TransTac.eval and TransTac.oov. TransTac.eval contains very few names. Therefore, incorrect name classing at the pre-processing step hurts the translation quality for the system *name_t*. The CN decoder can improve the translation quality by recovering from a name classing error by choosing the non-name path. Transliteration of OOVs (OOVTranslit) can improve the translation quality if any of the OOVs are names. Human evaluations capture the behaviors of the CN decoder and OOVTranslit by giving a slightly higher (statistically insignificant) score to OOVTranslit, 3.22, and the CN decoder, 3.20, than to the baseline, 3.16. All three systems, baseline, OOVTranslit and CN, however, received the same BLEU scores, 33.35. This seems to reflect the fact humans can easily capture the spelling variation of names whereas the automatic evaluation with 1 reference cannot.

Transtac.oov has a high OOV and a high name ratios and all OOVs are names. Therefore, name classing improves the translation quality as long as the correctly classed names out-number the incorrectly classed ones, explaining the higher translation quality of *name_t* than the baseline. OOVTranslit improves the translation quality over the baseline because all OOVs are names. The CN decoder out-performs all three other systems by correctly disambiguating non-OOV names and transliterating name OOVs. BLEU scores and human evaluation scores show the same pattern.

For TransTac.name with a high name and a low OOV ratios, however, human evaluation and BLEU scores show the opposite pattern, although none of the BLEU scores are statistically significantly better than others (note the small evaluation data size of 79 segments and 514 tokens). Since most names in this data set are known to the translation vocabulary and is highly ambiguous, we expect the CN decoder to out-perform all other systems. This expectation is borne out in the human evaluations, but not in BLEU scores. Our analysis indicates that the apparent inconsistency between BLEU and human evaluation scores is primarily due to spelling variations of a name, which are not captured by BLEU with just one reference, cf. (Li et al., 2013). Out of the human evaluated 64 names in TransTac.name, the baseline system produced the same spelling as the reference 34 times (53.13%), which contrasts with 28 times (43.75%) by the CN decoder. Overall, the CN decoder produced 62 correct name translations, about 20% more than 49 correct translations by the baseline system. Table 4 shows the names for which the reference spelling agrees with the baseline system, but disagrees with the CN decoding followed by transliteration.

Reference	CN output	Reference	CN output	Reference	CN output
tikrit	*tikreet*	*mariam*	*maryam*	*mousa*	*moussa*
ajlan	*al-`ajlan*	*jaafar*	*gaafar*	*basra*	*al-basra*

Table 4. Name Spelling Variations

[12] We ensure that any name classed input word *$name* is translated into *$name* by adding *$name* to the translation vocabulary, and the input word for *$name* is transliterated in the post-processing stage.

[13] We also evaluated another system, called name_st, which is trained with name classing and decoded with name classing using the baseline decoder. BLEU scores on TransTac.eval and TransTac.oov indicated that model training and decoding with name classing (name_st) is only slightly better than model training without name classing and decoding with name classing (name_t).

[14] For TransTac.eval data, we selected the sentences containing words tagged as name, i.e. NOUN_PROP, by the automatic part-of-speech tagger. The name ratio around 0.5% in Table 2 is computed on the basis of human annotations on the reference translation.

To verify that the inconsistency between BLEU and human evaluation scores is due to name spelling variations which humans capture but automatic metrics does not, we recomputed BLEU scores after normalizing spellings of the system outputs to be consistent with the reference translation spelling. The recomputed BLEU scores are denoted by TransTac.name_spnorm in Table 3, which shows that the recomputed BLEU scores are indeed consistent with the human evaluation scores.[15] Also note that the translation quality improvement by transliterating OOV names is well captured in human evaluation scores, 3.19 in the baseline vs. 3.36 in the system OOVTranslit, but not in BLEU scores, 35.03 in both baseline and OOV-Translit.

We point out that the same name is often spelled differently in various parts of our training corpus and even in the same reference translation, e.g. *al-aswad* vs. *aswad*, *jassim* vs. *jasim*, *risha* vs. *rasha*, *mahadi* vs. *mehdi* vs. *mahdi*, etc., as had been noted in Al-Onaizan and Knight (2002b), Huang et al. (2008).

6 Related Work

Al-Onaizan and Knight (2002a) propose an Arabic named entity translation algorithm that performs at near human translation accuracy when evaluated as an independent name translation module. Hassan et al. (2007) propose to improve named entity translation by exploiting comparable and parallel corpora. Hermjakob et al. (2008) present a method to learn when to transliterate Arabic names. They search for name translation candidates in large lists of English words/phrases. Therefore, they cannot accurately translate a name if the correct English name is missing in the word lists. Their restriction of named entity transliteration to rare words cannot capture name interpretations of frequent words, e.g. صباح (Sabah/morning), if the name interpretations are absent in the parallel corpus. Li et al. (2013) propose a Name-aware machine translation approach which tightly integrates high accuracy name processing into a Chinese-English MT model. Similar to Hermjakob et al. (2008), they restrict the use of name translation to names occurring less than 5 times in the training data. They train the translation model by merging the name-replaced parallel data with the original parallel data to prevent the quality degradation of high frequency names.

Onish et al. (2010) present a lattice decoding for paraphrase translations, which can handle OOV phrases as long as their paraphrases are found in the training corpus. They build the paraphrase lattices of the input sentence, which are given to the Moses lattice decoder. They deploy the source-side language model of paraphrases as a decoding feature.

Stalls and Knight (1998) propose a back-transliteration technique to recover original spelling in Roman script given a foreign name or a loanword in Arabic text, which consist of three models: a model to convert an Arabic string to English phone sequences, a model to convert English phone sequences to English phrases, a language model to rescore the English phrases. They use weighted finite state transducers for decoding. Al-Onaizan and Knight (2002b) propose a spelling-based source-channel model for transliteration (Brown et al., 1993), which directly maps English letter sequences into Arabic letter sequences, and therefore overcomes Stalls and Knight's major drawback that needs a manual lexicon of English pronunciations. Sherif and Kondrak (2007) propose a substring-based transliteration technique inspired by phrase based translation models and show that substring (i.e. phrase) models out-perform letter (i.e. word) models of Al-Onaizan and Knight (2002b). Their approach is most similar to the current approach in that we both adopt phrase-based translation models for transliteration. The current approach and Sherif and Kondrak (2007), however, diverge in most technical details including word alignments, phrase extraction heuristics and decoding, although it is not clear how they estimate transliteration probabilities. Crucially, we use the same set of decoder feature functions (excluding distortion models) as the end-to-end phrase translation system including lexical weights for phrases and a sentence in both directions and word/phrase penalties, whereas Sherif and Kondrak (2007) use only transliteration and language models for substring

[15] The spellings of the CN decoder output are normalized as follows: 38 instances of names, 2 instances of *'s* to *is*, 2 instances of *the city of arar* to *arar city* and 1 instance of *talk with* to *speak to*. Only name spelling normalizations were necessary for other system outputs.

transducer. We noted in Section 4 that inclusion of all decoder feature functions improves the accuracy by 15.6% absolute, compared with using just translation and language models for decoding.

7 Conclusion

We proposed a confusion network decoding to disambiguate Arabic names between name and non-name interpretations of an input word and character-based phrase transliteration models for NAME/OOV transliteration.

Name classing at the pre-processing step, coupled with name transliteration at the post-processing step, enables the system to accurately translate OOV names. Robust TM/LM probability estimations of names on the class label *$name* enable the system to correctly translate names even when the name interpretation of an in-vocabulary word is absent from the training data. Confusion network decoding can recover from name classing errors by choosing an alternative decoding path supported by decoder feature functions, obviating the need for stipulating a count threshold of an input token for name translation. The character-based phrase transliteration system achieves 90% exact match accuracy on 500 unique name pairs, utilizing all of the phrase decoder feature functions except for distortion models. We capture the asymmetries of English and Arabic vowel systems by treating any instance of an unaligned English vowel as a spontaneous word that can be attached to the preceding or following target phrases for phrase acquisition.

Although we proposed the confusion network decoding and character-based phrase transliteration models in the contexts of Arabic name disambiguation and transliteration tasks, the techniques are language independent and may be applied to any languages.

Acknowledgements

This work has been funded by the Defense Advanced Research Projects Agency BOLT program, Contract No. HR0011-12-C-0015. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the view of DARPA. We would like to thank Lazkin Tahir for his tireless effort on human evaluations. We also thank anonymous reviewers for their helpful comments and suggestions.

References

Y. Al-Onaizan and K. Knight. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 400–408.

Y. Al-Onaizan and K. Knight. 2002. Machine Transliteration of Names in Arabic Text. In *Proceedings of the Association for Computational Linguistics Workshop on Computational Approaches to Semitic Languages*.

Y. Al-Onaizan and K. Papineni. 2006. Distortion models for Statistical Machine Translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 529–536.

N. Bertoldi, R. Zens, and M. Federico.2007. Speech translation by confusion network decoding. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1297–1300.

P. Brown, S. Della Pietra, V. Della Pietra and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. In *Computational Linguistics, 19(2)*, pages 263–311.

S. Chen and J. Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. TR-10-98. Computer Science Group. Harvard University.

C. Dyer, S. Muresan, and P. Resnik. 2008. Generalizing Word Lattice Translation. In *Proceeding of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1012–1020.

N. Habash and F. Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation, In *Proceedings of the North American Chapter of the Association for Computational Linguistics* (NAACL), pages 49–52.

A. Hassan, H. Fahmy, and H. Hassan. 2007. Improving Named Entity Translation by Exploiting Comparable and Parallel Corpora. In *Proceeding RANLP'07*, pages 1–6.

U. Hermjakob, K. Knight, and H. Daume III. 2008. Name Translation in Statistical Machine Translation:Learning When to Transliterate. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 389–397.

S. Hewavitharana, D. Mehay, S. Ananthakrishnan, and P. Natarajan. 2013. Incremental Topic-Based Translation Model Adaptation for Conversational Spoken Language Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 697-701.

F. Huang, A. Emami, and I. Zitouni. 2008. When Harry Met Harri, هاری and 亨利 : Cross-lingual Name Spelling Normalization. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 391–399.

P. Koehn, F. Josef Och, and D. Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – Volume 1*, pages 127–133.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 177–180.

Y. Lee, K. Papineni, S. Roukos, O. Emam and H. Hassan. 2003. In *Proceedings of the 41st Annual Meeting of Association for Computational Linguistics – Volume 1*, pages 399–406.

Y. Lee. 2004. Morphological Analysis for Statistical Machine Translation. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics: Short Papers*, pages 57–60.

H. Li, J. Zheng, H. Ji, Q. Li and W. Wang. 2013. Name-aware Machine Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 604–614.

F. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. In *Computational Linguistics 29(1)*, pages 19–51. MIT Press.

T. Onish, M. Utiyama and E. Sumita. 2010. Paraphrase Lattice for Statistical Machine Translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics Short Papers*, pages 1–5.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

A. Ratnaparkhi. 1996. A Maximum Entropy Model for Part-Of-Speech Tagging. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 133–142.

T. Sherif and G. Kondrak. 2007. Substring-Based Transliteration. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics,* pages 944–951.

B. G. Stalls and K. Knight. 1998. Translating Names and Technical Terms in Arabic Text. *In Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages*.

C. Tillmann. 2003. A Projection Extension Algorithm for Statistical Machine Translation. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 1–8.

C. Tillmann and H. Ney. 2003. Word Reordering and a Dynamic Programming Beam-Search Algorithm for Statistical MT. *Computational Linguistics 29(1)*, pages 97–133. MIT Press.

S. Vogel, H. Ney and C. Tillmann. 1996. HMM-based word alignment in statistical machine translation. In *Proceedings of the 16th International Conference on Computational Linguistics*, Volume 2, pages 836–841.

O. Zayed, S. El-Beltagy, O. Haggag. An Approach for Extracting and Disambiguating Arabic Person's Names Using Clustered Dictionaries and Scored Patterns. In *Natural Language Processing and Information Systems Lecture Notes in Computer Science*. Vol. 7934, 2013, pages 201–212.

B. Zhao and S. Chen. 2009. A Simplex Armijo Downhill Algorithm for Optimizing Statistical Machine Translation Decoding. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics Short Papers*, pages 21–24.

R. Zen and H. Ney. 2004. Improvements in phrase-based statistical machine translation. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics*, pages 257–264.

Fourteen Light Tasks for Comparing Analogical and Phrase-based Machine Translation

Rafik Rhouma
RALI / DIRO
Université de Montréal
rafikrhouma@live.fr

Philippe Langlais
RALI / DIRO
Université de Montréal
felipe@iro.umontreal.ca

Abstract

In this study we compare two machine translation devices on twelve machine translation medical-domain specific tasks, and two transliteration tasks, altogether involving twelve language pairs, including English-Chinese and English-Russian, which do not share the same scripts. We implemented an analogical device and compared its performance to the state-of-the-art phrase-based machine translation engine Moses. On most translation tasks, the analogical device outperforms the phrase-based one, and several combinations of both systems significantly outperform each system individually. For the sake of reproducibility, we share the datasets used in this study.

1 Introduction

A *proportional analogy* is a relation between 4 objets, x, y, z and t, noted $[x : y :: z : t]$, which reads x is to y as z is to t. A *formal proportional analogy*, hereafter analogy, is a proportional analogy which involves a relationship at the graphemic level, such as [*atomkraftwerken* : *atomkriegen* :: *kraftwerks* : *kriegs*] in German. *Analogical learning* is a holistic learning paradigm (sketched in Section 2) which relies on proportional analogies for generalizing a training set.

Lepage and Denoual (2005b) pioneered the application of analogical learning to Machine Translation (MT). Different variants of their system have been tested within the IWSLT evaluation campaigns (Lepage and Denoual, 2005a; Lepage and Lardilleux, 2008; Lepage et al., 2008; Lepage et al., 2009). Since then, a number of studies have been investigating analogical learning for performing more specific machine translation tasks. Langlais et al. (2009) applied it to translating medical terms, and Langlais and Patry (2007) investigated the more specific task of translating unknown words, a problem simultaneously investigated in (Denoual, 2007). Recently, Langlais (2013) applied formal analogies to transliterate English proper names into Chinese.

Those works suggest, at least on the tasks investigated, that analogical translation typically shows better precision than phrase-based Statistical MT (SMT), but at a much lower recall. Still, the analogical devices tested in these works vary from task to task, making it difficult to draw a clear picture of the strengths and weaknesses of analogy-based translation. In this study, we perform a systematic comparison of an analogical and a phrase-based MT engine for the translation of fourteen different testbeds. We also improve the state-of-the-art of analogical learning by revisiting the aggregation step of the process. In particular, we observe that ranking analogical candidates according to random forests improves the performance of the analogical device, over training a classifier, as proposed for instance in (Langlais, 2013). On each task we tackle, we report improvements to the state-of-the-art in analogical learning.

In the remainder of this paper, we describe the principle of analogical learning and sketch our analogical device in Section 2. We describe our experimental protocol in Section 3. We analyze the performance of several variants of our analogical device in Section 4 and compare it to a state-of-the-art phrase-base SMT engine. We conclude this work and discuss future avenues in Section 5.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 444–454, Dublin, Ireland, August 23-29 2014.

2 ANALOGICAL LEARNING

2.1 Principle

We note $[x : y :: z : ?]$ an *analogical equation*. It can have 0 or several solutions, depending on the definition of analogy being considered. We are given a training set (or memory) of pairs of *input* and *output* forms that are in (translation) relation: $L = \{\langle x_1, y_1\rangle, \ldots, \langle x_l, y_l\rangle\}$, and we note $\tau(x)$ the set of output forms to which the input form x corresponds in the training set: $\tau(x) = \{y : \langle x, y\rangle \in L\}$.

Given an input form u unseen at training time, analogical learning generates its associated output form (in our case its translation), by accomplishing 3 steps. First, analogies in the input space $[x : y :: z : u]$ are searched for. Second, output equations $[x' : y' :: z' : ?]$ are solved for all x', y', and z' in $\tau(x)$, $\tau(y)$, and $\tau(z)$ respectively. By applying those two steps (that we call the *generator*), a number of candidate solutions are typically produced. They need to be aggregated. This is the purpose of the third step, or *selector*. Note that for the mapping to happen between input and output strings, there is no attempt to align subsequences of forms in both spaces, as it is typically done in statistical MT. There is actually no alignment whatsoever: analogies are treated in each space separately, and the mapping is the result of the inductive bias which promotes that an analogy in the input space corresponds to an analogy in the output space.

Figure 1 depicts the overall process for the translation of the English term *proton pump inhibitors* into Spanish, given a memory of pairs such as $\langle$*blood coagulation factors*, *factores de coagulación sanguínea*$\rangle$ and $\langle$*proton pumps*, *bombas de protones*$\rangle$. 6 input analogies are being identified (2 are reported), therefore 6 (output) equations are being solved, yielding a total of 5268 different forms that are sorted in decreasing order of frequency with which they have been generated. This is the output of the generator. The reference translation (in bold) ranks 11$^{\text{th}}$ according to frequency. The aggregator finally selects two candidates from this list. The best ranked one according to the aggregator is the correct translation.

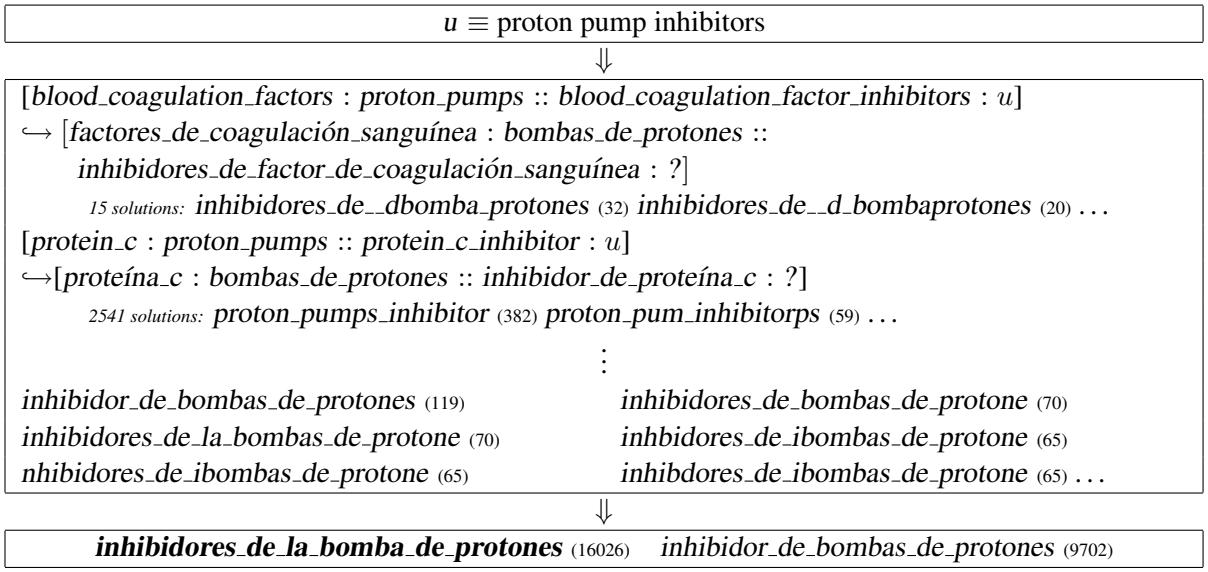


Figure 1: Excerpt of the translation session of the English term *proton pump inhibitors* into Spanish. The reference translation is in bold. Spaces are underlined for readability.

2.2 Implementation

Implementing such a learning procedure requires the definition of a formal analogy, the implementation of an analogical solver, as well as a way to handle computational issues: the identification of input analogies is an operation a priori cubic in the size of the input space. We describe each component of our implementation below. In practice, and for the tasks we consider in this work, our implementation allows the translation of an input form within a few seconds on average.

We would like to point out that analogical learning often suffers from a silence issue, that is, there are (input) forms for which no solution is provided. This may happen because no input analogy is identified, or because none yields an output equation with solutions. In contrast, there are many forms for which several candidate translations will be provided, thus the need for a good aggregator (see next section). This happens because an equation typically allows many solutions, and because many input analogies might be identified for solving a given input form.

Formal Analogy We used the definition of formal analogy proposed by Yvon et al. (2004), where an analogy is defined in terms of *d-factorizations*. A d-factorization of a string x over an alphabet Σ, noted f_X, is a sequence of d *factors* $f_X \equiv (f_X^1, \ldots, f_X^d)$, where $f_X^i \in \Sigma^*$ for all i, and such that $f_X^1 \odot f_X^2 \odot f_X^d \equiv x$; where $\odot$ denotes the concatenation operator.

Definition 1. *$\forall\, x, y, z$ and $t \in \Sigma^\star$, [x : y :: z : t] **iff** there exists a 4-uple of d-factorizations (f_X, f_Y, f_Z, f_t) of x, y, z and t respectively, such that $\forall i \in [1, d]$, $(f_Y^i, f_Z^i) \in \left\{(f_X^i, f_t^i), (f_t^i, f_X^i)\right\}$. The smallest d for which this holds is called the degree of the analogy.*

For instance, [*protein_c* : *proton_pumps* :: *protein_c_inhibitor* : *proton_pump_inhibitors*] because of the 4-uple of 3-factorizations shown in Fig. 2, whose factors are aligned column-wise for clarity, and where spaces (underlined) are treated as regular characters. There is no 4-uple of d-factorizations, with d smaller than 3. Therefore, the degree of this analogy is 3. Note that there are many 4-uple of d-factorizations for d greater than 3.

Figure 2: A 4-uple of 3-factorizations demonstrating that [*protein_c* : *proton_pumps* :: *protein_c_inhibitor* : *proton_pump_inhibitors*].

f_X	$\equiv$	(	*protein_c*	ϵ	ϵ)
f_Y	$\equiv$	(	*proton_pump*	ϵ	*s*)
f_Z	$\equiv$	(	*protein_c*	*_inhibitor*	ϵ)
f_t	$\equiv$	(	*proton pump*	*_inhibitor*	*s*)

Analogical Solver With the aforementioned definition, it has been showed by Yvon et al. (2004) that the set of solutions to an analogical equation is a rational language, therefore we can build a finite-state machine for encoding those solutions. In practice however, the automaton is non-deterministic, and in the worst case, enumerating the solutions can be exponential in the length of the forms involved in the equation. We adopted the solution proposed in (Langlais et al., 2009) which consists in sampling this automaton without building it. The more we sample this automaton, the more solutions we produce. It is sufficient to note that typically, a solver produces several solutions to an equation, many being simply spurious, which means that, while they obey the definition of formal analogy, they are not valid forms.

Figure 3: Three most frequent solutions to the equation [*protein_c* : *proton_pumps* :: *protein_c_inhibitor* : *?*] along with their frequency, as a function of the number of samples considered 10^n. nb stands for the total number of solutions produced.

n	nb	solutions		
1	43	*p_inhibitorroton_pumps* (2)	*proton_p_inhiubitomrps* (2)	*prot_ion_pnhibitumorps* (2)
2	320	*proton_pumps_inhibitor* (8)	*proton_pum_inhibitposr* (4)	*prot_inhibion_pumtorps* (4)
3	2 597	*proton_pumps_inhibitor* (121)	*roton_pumpps_inhibitor* (19)	*proton_pump_inhsibitor* (19)
4	16 006	*proton_pumps_inhibitor* (764)	*proton_pump_inhibsitor* (103)	*proton_pump_isnhibitor* (95)
5	72 610	*proton_pumps_inhibitor* (3706)	*proton_pump_sinhibitor* (501)	*proton_pump_inhibitosr* (481)

To illustrate this, Figure 3 reports the solutions produced to the equation [*protein_c* : *proton_pumps* :: *protein_c_inhibitor* : *?*] by our implementation of the solver, as a function of the number of samplings done in the automaton. Clearly, many solutions are not valid forms in English, although they define

proper solutions according to the aforementioned definition. Note that with enough sampling, the solution *proton_pumps_inhibitor* (involving a degree-2 analogy) is the most frequently generated one, while the solution *proton_pump_inhibitors* involved in the analogy illustrated in Figure 2 is generated less often (typically at the 10^{th} position).

Searching input analogies Identifying input analogies for an input term u is an operation a priori cubic in the size of the input space. Langlais and Yvon (2008) developed an algorithm for speeding up the search procedure that we adopted in this work. The main idea is to exploit a property of formal analogies (Lepage and Shin-ichi, 1996):

$$[x : y :: z : u] \Rightarrow |x|_c + |u|_c = |y|_c + |z|_c \quad \forall c \in \mathcal{A} \tag{1}$$

where $\mathcal{A}$ is the (input) alphabet, and $|x|_c$ stands for the number of occurrences of symbol c in x.

The strategy consists in first selecting a form x in the input space. This enforces a set of necessary constraints on the counts of symbols that any two forms y and z must satisfy for $[x : y :: z : u]$ to hold. By considering all forms x in turn, we collect a set of candidate triplets for u. We then have to find out which of these triplets form an analogy with u. Formally, we search for:

$$\begin{aligned}\{\langle x, y, z\rangle : \quad & x \in \mathcal{I}, \\ & \langle x, y\rangle : y \in \mathcal{I} \text{ and } |x|_c + |u|_c = |y|_c + |z|_c \quad \forall c \in \mathcal{A}, \\ & [x : y :: z : u]\}\end{aligned} \tag{2}$$

where $\mathcal{I} \equiv \{x_1, \ldots, x_l\}$. This strategy relies on the fact that one can efficiently identify the pairs $\langle y, z\rangle$ that satisfy a set of constraints on symbol counts. See (Langlais et al., 2009) for the *tree-count* solution we implemented in this work.

3 Experimental Protocol

3.1 Tasks

We use two families of tasks in this study. The first one concerns the translation of medical terms, the second one is about transliterating proper names. The main characteristics of the datasets we consider are reported in Table 1. If both tasks are of importance in practice, we admit that they are rather specific. The reason for this is that analogical learning is quite computationally intensive. Therefore, tackling broader tasks, such as those typically considered in MT evaluation campaigns is currently too challenging.

Medical term translation We use the datasets described in (Langlais et al., 2009). Part of the data comes from the Medical Subject Headings (MESH) thesaurus. This thesaurus is used by the US National Library of Medicine to index the biomedical scientific literature in the MEDLINE database. The MESH material concerns five language pairs with three relatively close European languages (English-French, English-Spanish and English-Swedish), a more distant one (English-Finnish) and one pair involving different scripts (English-Russian). The material was split in three randomly selected parts (TRAIN, DEV and TEST), so that the development and test material contain exactly 1000 terms each. Roughly a third of the examples are pairs of single-word terms.

For the Spanish-English language pair, a set of medical terms from the Medical Drug Regulatory Activities thesaurus (MEDDRA) is also available. This dataset contains roughly three times more terms than the Spanish-English material from the MESH dataset. Forms in the dataset are typically longer and the percentage of examples that are pairs of single-word terms is only 5.6%. This set is used for studying how the silence rate of analogical learning evolves with the size of the training set.

We are pleased to share those datasets. They can be downloaded at `http://rali.iro.umontreal.ca/rali/?q=en/12-medical-translation-tasks`.

Proper name transliteration This task is part of the NEWS evaluation campaign conducted in 2009 (Li et al., 2009). The organizers of this evaluation campaign kindly provided us with the Chinese-English dataset. This task has been investigated recently by Langlais (2013). This allows a direct comparison of our analogical system. We also consider the reverse transliteration direction, *i.e.*, the transliteration of

Chinese proper names into English. This was done by simply switching the source and target languages in the NEWS dataset.

	TRAIN		TEST	DEV	
	nb	$avg.$	nb	oov%	oov%	examples:
MESH					
FI	19 787	19.3	1 000	65.0	63.8	*orthodontic retainers*
FR	17 230	21.5	1 000	35.8	36.8	↪ FI:*tandregleringshjälpmedel, förankrade*
RU	21 407	38.5	1 000	42.3	45.1	
ES	19 021	21.5	1 000	37.4	34.9	*aid to families with dependent children*
SW	17 090	17.3	1 000	69.3	70.0	↪ SW:*bidrag till barnfamiljer*
MEDDRA						*poor urinary stream*
ES	65 276	34.6	1 000	7.1	7.1	↪ ES: *chorro de orina débil*
NEWS						*Abberley* → CN:阿伯利
CN	31 961	9.5	2 896	—	—	*Schemansky* → CN:谢曼斯基

Table 1: Main characteristics of our datasets. nb indicates the number of pairs of terms in a bitext, $avg.$ indicates the average length (in symbols) of the foreign forms; oov% indicates the percentage of out-of-vocabulary types (space-separated types of TEST or DEV unseen in TRAIN).

3.2 Evaluation Metrics

All the tasks we consider are characterized by a rather high out-of-vocabulary rate (see Table 1). Thus, word-based translation is not an adequate solution. Therefore, we devised engines which translate sequences of symbols (characters), without taking into account the notion of word.[1] In particular, spaces in forms were considered as any ordinary symbol. Measuring how close a candidate translation is to a reference is of little interest here, since typically, a medical term only has one reference translation that we seek to discover. Therefore, rewarding partially correct translations (like a metric such as BLEU (Papineni et al., 2002) does) is not especially useful. Therefore we report the *accuracy* of the first candidate proposed by a translation device for each source term. Accuracy is measured as the percentage of test forms for which the first candidate is the sanctioned one. So in the example of Figure 1, the aggregator illustrated in the bottom frame would get one point since the first translation produced is the sanctioned one, while an aggregator that would pick the most frequently generated candidate would receive no point. Accuracy is the main metric of the NEWS evaluation campaign, and we used the NEWS 2009 official evaluation script[2] in order to compute it. Also of interest for the analogical devices, is the *silence rate*, computed as the percentage of input forms for which no output is generated. As we will see, on some tasks, this ratio can be rather high, a clear limitation of the analogical approach we discuss in Section 5.

3.3 Systems

3.3.1 Reference System

We compare a number of analogical devices to the state-of-the-art statistical translation engine Moses (Koehn et al., 2007). In a nutshell, SMT seeks to find the optimal translation $\hat{e}$ of a sentence f using to a log-linear combination of models (h_i), including a language model $p(e)$ which scores how likely a hypothesis is in the target language, and a translation model $p(f|e)$ which predicts the likelihood that two sentences are translations:

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, p(f|e)p(e) \approx \underset{e}{\operatorname{argmax}} \exp\left(\sum_i \lambda_i h_i(e, f)\right) \quad (3)$$

[1] We tried it, but the results are very low.

[2] http://translit.i2r.a-star.edu.sg/news2009/

We trained such a system at the character level,[3] very similarly to the approach described in (Finch and Sumita, 2010). Such a system has been massively used as a key component by the participants of the NEWS 2009 evaluation campaign. We used the default configuration of Moses for training and testing the SMT engine. We trained a 5-gram character-based language model on the target part of the TRAIN material.[4] We used the DEV corpus for tuning the coefficients (λ_i) given to each model. The resulting system have high BLEU scores (*e.g.*, 55.7 for the CN–EN NEWS task). A random extract of the phrase-table learnt by Moses for the English-Swedish system is shown in Figure 4.

Figure 4: Phrases stored in the SW–EN phrase-table, along with 4 estimations of their likelihood

```
eckos          | | | echos          | | | 0.303 0.006    0.303 0.002
, _ kvinn      | | | , _ fema       | | | 0.101 8.3e-09  0.303 2.5e-11
eckrina        | | | eccrine        | | | 0.151 0.009    0.303 0.001
edel           | | | ator           | | | 0.002 4.6e-06  0.002 1.9e-06
```

3.3.2 Analogical Systems

We ran our analogical generator for translating the DEV set, using the TRAIN set as a memory. The candidate translations generated were used for training our aggregators in a supervised way. Then, we generated the translation of the TEST terms with our analogical device, making use of the TRAIN and the DEV set as a memory. Adding the DEV corpus to the memory used by the generator is acceptable since it does not involve training. We only consider the (at most) 100 most frequently generated forms for each input term. This certainly decreases the recall of the analogical device, but simplifies the overall process. These candidates are passed on to the aggregator, and one candidate is finally selected.

Aggregators A number of aggregators have been proposed in the literature. Lepage and Denoual (2005b; Stroppa and Yvon (2005) keep the candidate that has been generated the most frequently. We call this aggregator FREQ henceforth. Langlais et al. (2009) trained a binary classifier to recognize good examples from bad ones. A training instance in their case was constituted by an input analogy, and the corresponding output equation along with one solution produced. Therefore, for the translation of the input form u, any pair ($[x : y :: z : u]$, $[x' : y' :: z' : c]$), with x', y', and z' in $\tau(x')$, $\tau(y')$, and $\tau(z')$ respectively, and c a candidate translation would be considered for classification. The authors had to face a particularly unbalanced classification task. Indeed, when translating a test form, a large number of input analogies can be considered (hundreds) and therefore a large number of output equations, each generating potentially numerous solutions (recall the translation session in Figure 1). They reported for instance that on the English-to-Finnish translation direction, they had over 2.7 million instances to classify among which slightly less than 4200 were positive ones. Not only is this task very unbalanced, it is also challenging to train a classifier on that many instances.

In this work, we reframe the classification task as one of identifying the correct candidate among the 100 most frequently generated ones. An instance in this setting is simply a candidate form, and not a pair of analogies as in (Langlais et al., 2009). This is still an unbalanced task, since typically at most one candidate will be correct, but the ratio 1:100 is more manageable, and the classification task is easier to deploy. A total of 81 features are computed for each candidate form:

ANA is a set of 59 features (mostly analogical ones, therefore the name). Some features are characterizing the candidate solution thanks to a character-based language model (the same 5-gram language model used by Moses). Others are characterizing the process with which a given candidate is generated, such as the number of input analogies involved, the number of target equations that generated the candidate, the average degree of the analogies involved, etc. The remaining features are cohort-based ones, such as the rank of the candidate according to frequency, to the language model, etc.

[3] This was done by separating each character in the training material by a space; true spaces being previously substituted by a special character not belonging to the alphabet.

[4] A Markov model of order 4. We tried higher order models, without gains.

IBM is a set of 18 features that are capitalizing on statistical word alignment. The alignment models being used are word-based generative models that are exploited by Moses in order to build the phrase table, namely IBM models, therefore the name of the feature set. Different likelihood-based features were computed, as well as rank features (the rank of the likelihood of the candidate in the cohort of candidates, the ratio of its likelihood over the highest likelihood in the cohort, etc.). To our knowledge, this is the first attempt to capitalize on such features in the analogical sphere.

MOS is a set of 4 features that are exploiting the n-best solutions we asked Moses to produce. The idea being that if Moses ranks a given analogical candidate well (in rank or in score), this is a good indicator of the salience of this candidate. The two main features are the rank of the candidate in the n-best list and its score as given by Moses (or 0 if Moses does not produce the candidate).

We point out that an analogical device with an aggregator that uses the features ANA and IBM is basically making use of the same models (language and IBM) as those used by Moses. It is therefore interesting to compare this configuration to Moses. Also, the aggregators that are making use of the MOS features are performing a kind of combination that has not been explored so far. Note also that we did not engineer task-specific features. For instance, for the medical term translation task, terms and their translation often share the same latin root, which could be exploited to boost performance.

We investigated two families of classifiers: voted-perceptrons (Freund and Schapire, 1999) and support vector machines (Cortes and Vapnik, 1995). We investigated all the metaparameters that `LibSVM` (Chang and Lin, 2011) offers (penalization, kernels, etc.), but did not manage to outperform the performance of the former classifier (an in-house implementation) that we trained with 500 epochs. Therefore we only report the results of the voted-perceptron classifier (VP). Classifying each candidate solution separately is not optimal. This is why we also investigated reranking algorithms in this study. To our knowledge, this is the first time reranking is applied in analogical learning. We tested the algorithms implemented in `RankLib`[5] and `SVMRank`[6] toolkits, and found random forests (Breiman, 2001) to be the most beneficial. We note it RF in the sequel. We only considered bipartite ranking in this work (Argarwal, 2005).

4 Results

4.1 MESH

The accuracy of the translation devices we trained are summarized in Table 2 for the 10 translation directions we tested. This table calls for several comments. First, it is noticeable that our implementation of analogical learning with the FREQ aggregator (line LYZ) outperforms the equivalent configuration in (Langlais et al., 2009) by roughly 10 absolute points in accuracy. We also observe a slight reduction of the silence rate, which still remains high, since on average 54.6% of the test forms do not receive any candidate solution. Second, we observe that Moses slightly outperforms the FREQ variant at a silence rate of 0 (a decision is always returned by Moses). This suggests that FREQ is actually more precise than Moses and calls for a simple combination where the analogical device is trusted whenever it produces a candidate solution, and Moses otherwise. This is illustrated in line CASC(FREQ,MOSES). We observe a clear improvement over each system: almost 10 absolute accuracy points on average are gained by this combination (38.6%). Third, we observe that the aggregators that are relying on a classifier or a reranker offer better performance than picking the most frequently generated form (as done by FREQ). The gains are not especially high, but are consistent over all translation directions. Overall, it seems that the random forest reranker we investigated (the best reranker we tried) offers the best performance on average. This represents 92% of the reachable accuracy according to line ORACLE which involves a perfect classifier. This validates the usefulness of the features we designed. As far as features are concerned, it seems that using all of them leads to better performance overall, and that the configurations that are making use of the ANA and IBM feature sets are comparable or higher than Moses. Cascading the best analogical device with Moses (last line) finally gives a slight boost in accuracy. In the end, the best system we tested correctly translated 41.9% of the test terms in the first position on average across translation directions.

[5] `http://people.cs.umass.edu/~vdang/ranklib.html`

[6] `http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`

	→ EN					EN →					
	FR	RU	FI	ES	SW	FR	RU	FI	ES	SW	*avg.*
LYZ	18.1	20.8	16.4	20.3	18.2	14.6	18.7	14.9	19.5	15.4	17.7
	(61.5)	*(57.9)*	*(55.2)*	*(57.4)*	*(55.4)*	*(58.8)*	*(53.8)*	*(52.9)*	*(53.0)*	*(57.2)*	*(56.3)*
FREQ	27.3	29.1	28.5	30.5	28.3	21.8	29.0	24.7	29.8	26.3	27.5
	(59.3)	*(56.7)*	*(53.7)*	*(55.6)*	*(54.3)*	*(56.0)*	*(52.5)*	*(50.9)*	*(51.6)*	*(55.2)*	*(54.6)*
MOSES	22.3	33.4	27.0	29.0	38.8	20.0	30.5	26.4	28.6	37.0	29.3
VP(ANA)	28.4	29.8	29.8	31.9	29.7	23.2	31.0	27.2	32.3	27.9	29.1
VP(ANA+IBM)	28.8	31.8	31.6	32.4	31.2	24.5	32.3	28.4	34.2	29.2	30.4
VP(ANA+IBM+MOS) †	29.2	32.3	31.6	32.8	31.9	25.0	32.6	28.8	34.0	30.1	30.8
RF(ANA)	28.3	29.8	30.7	32.0	29.5	23.0	31.2	27.4	31.6	28.3	29.2
RF(ANA+IBM)	29.1	31.6	31.8	32.8	31.0	24.4	32.4	28.7	33.5	30.1	30.5
RF(ANA+IBM+MOS)	29.4	31.8	32.2	32.9	32.4	24.9	32.5	29.9	34.0	31.1	31.1
ORACLE	31.3	34.0	34.9	35.2	34.9	28.2	35.7	33.2	37.3	33.3	33.8
	(68.7)	*(66.0)*	*(65.1)*	*(64.8)*	*(65.1)*	*(71.8)*	*(64.3)*	*(66.8)*	*(62.7)*	*(66.7)*	*(66.2)*
casc(FREQ,MOSES)	36.9	42.4	37.7	41.6	43.8	29.6	38.9	34.3	40.7	39.9	38.6
casc(†,MOSES)	38.8	45.6	40.8	43.9	47.4	32.8	42.5	38.4	44.9	43.7	41.9

Table 2: Accuracy on the MESH tasks. Figures in parenthesis are silence rates. LYZ stands for the system described in (Langlais et al., 2009), reproduced according to Table 4, p. 492. *avg.* indicates the average over the 10 translation directions.

4.2 MEDDRA

The results presented so far show that the analogical device is more accurate than the statistical one, but that it suffers from a high silence rate. We tested whether increasing the size of the training set would lower the silence rate. We used the datasets of MEDDRA for this. The results are reported in the left column of Table 3. We observe that the silence rate decreases drastically, since less than a fourth of the test forms do not receive a candidate translation. We also observe that the analogical devices, even the simplest FREQ, are far more accurate than Moses (over 30 absolute points on average). The poor performance of the SMT engine might be explained by the fact that the forms in the MEDDRA datasets are longer in terms of characters, therefore reducing the chance of getting the full translation right. Again, combining both approaches does improve accuracy, but the improvement is small since Moses is much less accurate on this task. Also, we observe that using a classifier is preferable to picking the most frequently generated form, and again, the random forest reranker delivers the best performance on average. It is noticeable however, that the performance is far less than the oracle's, therefore, there is still room for improvement.

4.3 NEWS

The right column of Table 3 summarizes the performance of the transliteration devices we trained on the NEWS tasks. The silence rate is rather low (less than 4%). Here again, we observe that aggregating by classifying or reranking is preferable to picking the most frequent solution. There is no clear difference between random forest and voted perceptron here. On the English-to-Chinese transliteration tasks, Moses outperforms the analogical devices, but the opposite is observed for the reverse transliteration direction. Our best configuration slightly outperforms the best analogical device reported in (Langlais, 2013), but the gain is likely not significant.

	MEDDRA				NEWS			
	ES–EN		EN–ES		CN–EN		EN–CN	
FREQ	52.2	*(25.1)*	45.5	*(16.7)*	17.2	*(2.5)*	43.3	*(3.7)*
MOSES	10.2		11.0		15.4		66.6	
VP(ANA)	55.1		46.8		20.0		57.3	
VP(ANA+IBM)	56.2		46.9		20.9		59.5	
VP(ANA+IBM+MOS) †					21.4		64.2	
RF(ANA)	54.1		49.3		20.9		57.8	
RF(ANA+IBM)	55.7		49.5		21.6		59.2	
RF(ANA+IBM+MOS)					22.3		64.1	
ORACLE	64.3	*(34.4)*	61.8	*(38.2)*	64.9	*(32.9)*	81.5	*(18.5)*
casc(FREQ,MOSES)	53.2		46.7		17.5		44.9	
casc(†,MOSES)	—		—				68.9	
(Langlais, 2013)							68.5	

Table 3: Accuracy on the MEDDRA and NEWS tasks. The performance of (Langlais, 2013) is taken from Table 1 p. 687.

4.4 Examples of translations

We conducted a random inspection of the outputs produced by Moses and by the analogical device which uses a voted perceptron classifier trained on the ANA and the IBM features.[7] We report in Figure 5 a few examples that we found representative of the problems each translation device faces. The FI–EN example shows a case where Moses fails to produce a valid sequence of words. The EN–ES example illustrates the weakness of Moses at reordering words. The CN–EN example shows the incorrect transliterations made by both systems, and the EN–CN one illustrates a failure of the analogical engine were *ph* and *us* are transliterated separately.

MESH*(FI–EN)*	*hammasytimen sairaudet*	NEWS*(CN–EN)*	本尼迪克特
Analog	*dental marrow diseases*	Analog	Bennidickt
MOSES	*dental ne diseases*	MOSES	BenniDickert
Reference	*dental pulp diseases*	Reference	Benedict
MEDDRA*(EN–ES)*	*intrinsic asthma with status asthmaticus*	NEWS*(EN–CN)*	Adolphus
Analog	*asma intrínseca con estatus asmático*	Analog	阿道夫厄斯
MOSES	*intrínseco asmático con estatus asmático*	MOSES	阿道弗斯
Reference	*asma intrínseca con estatus asmático*	Reference	阿道弗斯

Figure 5: Examples of analogical and phrase-based outputs

5 Discussion

We have applied analogical learning on a number of key tasks involving various language pairs. Overall, we confirm the findings of Langlais et al. (2009) and Langlais (2013) that analogical devices are typically more accurate than statistical phrase-based SMT, but that they are too often silent. We also verified that cascading the analogical device with Moses increases accuracy. We compared a number of classification algorithms and rerankers, and observed that overall, reranking by random forest

[7] This variant fares well compared to Moses in terms of information used (same language and IBM models).

delivers the best performance. Our implementation outperforms previously reported ones. Our generator is more efficient than the one described in (Langlais et al., 2009). Reranking candidate solutions is preferable to their classification, as proposed in (Langlais, 2013). In order to foster reproducibility, the datasets related to the medical-translation tasks we investigated can be downloaded at `http://rali.iro.umontreal.ca/rali/?q=en/12-medical-translation-tasks`.

We believe this systematic comparison shows the high potential of analogical learning as a translation engine. Still, this work raises a number of issues that we must address. First, we need to find ways to remedy analogical learning's high silence rate. Lepage and Denoual (2005b) describe a recursive process where the input form is split into two parts whenever no solution is returned in the first place. This process is at the very least costly and deserves further investigations. Lepage and Lardilleux (2008) augments the training set with sub-sentential alignment (bootstrapping). Second, the solver we use is producing many solutions that are currently ranked according to frequency. We are addressing the issue of producing less, but more accurate solutions, by integrating structured learning in the solver. Last, we investigated here the translation of sequences of characters on modestly sized tasks. We want to tackle broader translation tasks, *e.g.*, translating plain sentences, as done in (Lepage and Denoual, 2005b), to see whether our analogical device is still beneficial.

Acknowledgements

We thank the reviewers for their valuable comments and apologize for having failed to taking all of them into account in this version. This work has been partially funded by the Natural Science and Engineering Research Council of Canada. We are grateful to Fabrizio Gotti for his advice.

References

Shivani Argarwal. 2005. A study of the bipartite ranking problem in Machine Learning. Technical report, University of Illinois.

Leo Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):1–27, May.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Étienne Denoual. 2007. Analogical translation of unknown words in a statistical machine translation framework. In *MT Summit XI*, pages 135–141, Copenhagen, Denmark.

Andrew Finch and Eiichiro Sumita. 2010. Transliteration Using a Phrase-based Statistical Machine Translation System to Re-score the Output of a Joint Multigram Model. In *2nd Named Entities Workshop (NEWS'10)*, pages 48–52, Uppsala, Sweden.

Yoav Freund and Robert Schapire. 1999. Large Margin Classification Using the Perceptron Algorithm. *Mach. Learn.*, 37(3):277–296.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *45th ACL*, pages 177–180. Interactive Poster and Demonstration Sessions.

Philippe Langlais and Alexandre Patry. 2007. Translating Unknown Words by Analogical Learning. In *EMNLP*, pages 877–886, Prague, Czech Republic.

Philippe Langlais and François Yvon. 2008. Scaling up Analogical Learning. In *22nd International Conference on Computational Linguistics (COLING 2008), Poster session*, pages 51–54, Manchester, United Kingdom, Aug.

Philippe Langlais, François Yvon, and Pierre Zweigenbaum. 2009. Improvements in Analogical Learning: Application to Translating multi-Terms of the Medical Domain. In *12th EACL*, pages 487–495, Athens.

Philippe Langlais. 2013. Mapping Source to Target Strings without Alignment by Analogical Learning: A Case Study with Transliteration. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 684–689, Sofia, Bulgaria.

Yves Lepage and Étienne Denoual. 2005a. Aleph: an EBMT system based on the preservation of proportional analogies. In *2nd IWSLT*, pages 47–54, Pittsburgh, USA.

Yves Lepage and Étienne Denoual. 2005b. Purest ever example-based machine translation: Detailed presentation and assesment. *Mach. Translat*, 19:25–252.

Yves Lepage and Adrien Lardilleux. 2008. The GREYC Translation Memory for the IWSLT 2007 Evaluation Campaign. In *4th IWSLT*, pages 49–54, Trento, Italy.

Yves Lepage and Ando Shin-ichi. 1996. Saussurian Analogy: A Theoretical Account and Its Application. In *7th COLING*, pages 717–722.

Yves Lepage, Adrien Lardilleux, Julien Gosme, and Jean-Luc Manguin. 2008. The GREYC Translation Memory for the IWSLT 2008 Evaluation Campaign. In *5th IWSLT*, pages 39–45, Hawai, USA.

Yves Lepage, Adrien Lardilleux, and Julien Gosme. 2009. The GREYC Translation Memory for the IWSLT 2009 Evaluation Campaign: one step beyond translation memory. In *6th IWSLT*, pages 45–49, Tokyo, Japan.

Haizhou Li, A. Kumaran, Vladimir Pervouchine, and Min Zhang. 2009. Report of NEWS 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, NEWS '09, pages 1–18.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proc. of the 40th ACL*, pages 311–318, Philadelphia, Pennsylvania, USA.

Nicolas Stroppa and François Yvon. 2005. An Analogical Learner for Morphological Analysis. In *9th Conf. on Computational Natural Language Learning (CoNLL)*, pages 120–127, Ann Arbor, USA.

François Yvon, Nicolas Stroppa, Arnaud Delhay, and Laurent Miclet. 2004. Solving Analogies on Words. Technical Report D005, École Nationale Supérieure des Télécommuncations, Paris, France.

Finding Zelig in Text: A Measure for Normalising Linguistic Accommodation

Simon Jones[1], **Rachel Cotterill**[2], **Nigel Dewdney**[2], **Kate Muir**[3], **and Adam Joinson**[3]

[1]Department of Computer Science, University of Bath, BA2 7AY. s.jones2@bath.ac.uk
[2]Department of Computer Science, University of Sheffield, S1 4DP. r.cotterill@sheffield.ac.uk, acp08njd@sheffield.ac.uk
[3]Behavioural Research Lab, Faculty of Business and Law, University of the West of England, Bristol, BS16 1QY. kate.muir@uwe.ac.uk, adam.joinson@uwe.ac.uk

Abstract

Linguistic accommodation is a recognised indicator of social power and social distance. However, different individuals will vary their language to different degrees, and only a portion of this variance will be due to accommodation. This paper presents the *Zelig Quotient*, a method of normalising linguistic variation towards a particular individual, using an author's other communications as a baseline, thence to derive a method for identifying accommodation-induced variation with statistical significance. This work provides a platform for future efforts towards examining the importance of such phenomena in large communications datasets.

1 Introduction

"Zelig...protects himself by becoming like whoever he is around."
- The Narrator, Zelig (Allen, 1983)

When people converse, they often become more alike in their language in many different dimensions (Garrod and Pickering, 2004). This can include similarity in pronunciation (Giles, 1973), speech rates (Street, 1984), pause and utterance duration (Cappella, 1979), and volume (Natale, 1975). Similarly, in written communications people often converge in terms of features such as linguistic style (Danescu-Niculescu-Mizil and Lee, 2011), vocabulary, and syntax (Scissors et al., 2008). Communication Accommodation Theory (Giles and Ogay, 2007) proposes that interactants can adjust their communication style, such as accent, vocabulary, and use of jargon to sound more (convergence) or less (divergence) like the other person. Individuals typically converge to signal affinity with their interlocutor, and diverge to show interpersonal or social distance.

One area which has been largely overlooked, to date, is the role played by an individual's inherent tendency to accommodate (or not). We propose that some people are more apt than others to change their typical linguistic style to converge to that of their conversational partner. This paper introduces the Zelig Quotient, which is a new method for capturing the degree to which the variation in an individual's language use can be explained by their accommodation towards the style of their interlocutor. Using this score, it is then possible to measure the significance of an individual's accommodation within a specific communication pair: does each individual accommodate more or less than their personal norm?

In this paper, we firstly consider existing computational measures of linguistic accommodation. Although useful in measuring accommodation of specific linguistic features within dialog, current measures

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 455–465, Dublin, Ireland, August 23-29 2014.

do not permit examination of the role of an individual's inherent (or latent) predisposition to accommodate their linguistic style. Further, another area that has yet to be explored is the influence of social status and relationships between interlocutors on the likelihood of accommodation. We therefore demonstrate the applicability of the Zelig Quotient by applying the technique to large datasets of communications from three online community forums, in which social status and relationships between interlocutors are clearly defined. We close with a discussion of potential future directions in which the Zelig Quotient could be applied.

1.1 Computational Measures of Linguistic Accommodation

Several computational measures of linguistic accommodation already exist. These measures typically capture the extent to which language use increases in similarity or becomes 'adapted' either within a piece of text or between individuals in dialog. Church (2000) developed a method for determining lexical adaptation in text, by examining the probability of one word appearing in the later half of a document when it appears in the earlier half. This method has been used and extended by other researchers in the examination of lexical adaptation over time (Reitter et al., 2006), adaptation of syntactic constructions (Dubey et al., 2005) and to measure the prevalence and strength of linguistic feature adaptation in dialogs (Stenchikova and Stent, 2007). Along similar lines, linguistic style matching (LSM) techniques with LIWC measures (quantitative analysis of standard language categories) (Pennebaker et al., 2001) reveal the extent to which language use is coordinated between group members, either on a whole conversation or turn-by-turn level (Niederhoffer and Pennebaker, 2002). Ireland et al. (Ireland et al., 2011) used LSM techniques to study the predictive value of stylistic similarity in a social setting. They found that similarity of a few stylistic categories (such as the distribution of pronouns and determiners) was a good indicator of whether two individuals would vote to see one another again in a speed dating scenario.

The work of Huffaker et al. (2006) is particularly relevant to our examination of accommodation within online community forums. Huffaker et al compared three different measures of lexical convergence to assess message similarly in an online community over time. These included: Spearman's Rank Correlation, which has been used to determine message similarity between corpora (Kilgarriff, 2001), 'Zipping', referring to data compression algorithms, which has been used to measure the complexity of documents (Benedetto et al., 2002), and Latent Semantic Analysis, which has been used to measure semantic similarity across corpora (Coccaro and Jurafsky, 1998). All three measures showed divergence in message similarity both between individuals, and in the community as a whole, across time.

However, the common theme with all these techniques is that although they can effectively measure adaptation of linguistic feature use within and between dialogs, they fail to capture the precise direction of convergence or divergence between individuals (i.e., do both interactants within a conversational pair accommodate their language use to the same extent?) Thus, existing computational measures of linguistic accommodation fail to provide a fine-grained view of the dynamics of convergence within dyads or large groups. The measures discussed above only capture the extent to which members of the group match one another, and overlook precise details of an individual's movement from their existing language use towards that of the group.

1.2 Individual's Propensity to Accommodate

Further, whilst accommodation within dyads and groups has been measured extensively, one area which has been largely overlooked, to date, is the role played by an individual's inherent tendency to accommodate (or not). Some individuals may have a relatively stable linguistic style, whereas other individuals may be more likely to accommodate their linguistic style towards that of their conversational partner. We have been unable to find any methods for reliably measuring an individuals propensity to accommodate towards their interlocutors. We hypothesize that individuals are not equal with respect to their accommodation and propose the Zelig measure, detailed within this paper, as a means for quantifying this characteristic.

One factor which could conceivably influence an individual's tendency to accommodate is *social power*. Giles and Coupland (1991) state that "the power variable is one that emerges a number of times in the accommodation literatures and in ways that support the model's central predictions" (p.

Category	Examples	Category	Examples
Personal pronouns	I, his, their	Auxillary verbs	shall, be, was
Impersonal pronouns	it, that, anything	High-frequency adverbs	very, rather, just
Articles	a, an, the	Negations	no, not, never
Conjunctions	and, but, because	Quantifiers	much, few, lots
Prepositions	in, under, about		

Table 1: Word Categories Used for Calculating Linguistic Style

19). Demonstrations of the role of social power in accommodation include interviewees converging their speech style towards that of their interviewers during employment interviews (Willemyns et al., 1997), students accommodating their verbal and non-verbal behaviours to academic faculty members (Jones et al., 1999) and witnesses in courtrooms accommodating to the linguistic style of the questioning legal professional (Gnisci, 2005). Thus, individuals with low social power are more likely to accommodate their linguistic style. The Zelig Quotient allows explicit examination of research questions of this nature concerning accommodation and divergence associated with demographic variables such as social power.

2 The Zelig Quotient

"Wanting only to be liked, he distorted himself beyond measure."
- The Narrator, Zelig (Allen, 1983)

We propose the Zelig Quotient, a measure for normalizing linguistic variation. The Zelig Quotient is named for Leonard Zelig, the central character of the Woody Allen film Zelig, who is described as "the human chameleon" due to his propensity for taking on the characteristics of other people. This is the logical extreme of accommodating to one's audience. An author who always adopts the language style of the intended reader is totally Zelig-like, whereas an author who does not adapt at all has zero likeness. Opposite behaviour to Zelig (moving away from the audience) is also possible. Over-accommodation occurs when the author adopts elements of linguistic style of their intended reader, but emphasises to the point of overuse. In extreme cases this would be detected as parody. We need, therefore, to distinguish not only the distance between author and reader, but also the orientation. The Zelig Quotient thus shows the extent to which an individual changes their linguistic style from their 'typical' or baseline style, to move either towards or away from each of their conversational partners. The average Zelig score across all conversational partners can then be used to demonstrate the individual's general tendency to accommodate their language use to that of others.

2.1 Feature Selection

We have selected to study a set of features which are stylistic rather than semantic in nature; although consideration of whether two people are talking about the same topic is a valid research question, we currently wish to focus on their linguistic style. The best features for our purposes are those able to be varied with comparative freedom, without affecting the meaning of a message. We use a set of nine such features, taken from the linguistic style matching study conducted by Ireland et al. (2011) (see Table 1).

We used LIWC dictionaries for each category. LIWC processes a text file word by word, comparing each word to the dictionary and providing a count of the words in the file which match each category in the dictionary. Sums of words in each category are presented as percentage of total words in the file to correct for differences in text length between text files (Pennebaker et al., 2001). The use of LIWC is the basis of much recent work on linguistic style accommodation (Niederhoffer and Pennebaker, 2002; Danescu-Niculescu-Mizil et al., 2011; Danescu-Niculescu-Mizil and Lee, 2011) to which we want to relate.

2.2 Calculating the Zelig Quotient

We assume an author has a baseline linguistic style resulting in a baseline value for each of our stylistic features. However, we expect variation in the observed values due to sampling, an author's natural variation, constraints of message content and format, etc. as well as any movement due to accommodation.

We can estimate a baseline value for a specific feature, μ_f, by averaging over all the messages we have for an author a. Previous research has used a similar technique for establishing the baseline level of a lexical item in a dialog in order to study accommodation (Church, 2000; Stenchikova and Stent, 2007).

$$\mu_f(a) = \sum_{m=1}^{n_a} f_m(a)/n_a \tag{1}$$

where m is a message, n_a is the number of messages for a and $f_m(a)$ is the feature value in m.

We can further estimate the proportion of variance due to 'noise' and that due to accommodation by also calculating the average feature values on an author-reader (a, r) pairwise basis.

$$f(a, r) = \sum_{m=1}^{n_{ar}} f_m(a, r)/n_{ar} \tag{2}$$

where n_{ar} is the number of messages written by a to reader r.

Measuring the variance within a pair and then averaging over all pairs that author is party to gives an estimate of the natural variation in feature value for an author.

$$\sigma_f^2(a) = \frac{1}{R_a} \sum_r \sum_{m=1}^{n_{ar}} (f_m(a) - \mu_f(a))^2/n_{ar} \tag{3}$$

where R_a is the number of recipients of messages from author a.

The movement in a feature due to accommodation is simply taken to be the difference between the value seen within a communicative pair, and the author's baseline value.

Having calculated scores for several features, some of which may change more readily than others, we can consider authors as having corresponding points in an F-dimensional feature space described by the vector of feature values. The generalised phenomenon of accommodation can then be measured in terms of movement in this feature space, rather than movement in individual features. Note that to avoid bias towards particular features when considering overall movement, feature scales must be comparable. Movement in an author's language may be large, but it may not necessarily be towards the reader.

We represent movement and distances between the author's baseline position, accommodated position, and the reader's position as vectors in the feature space:

$$\mu = \{\mu_1(a), \mu_2(a), ..., \mu_F(a)\}$$
$$a = \{f_1(ar), f_2(a, r), ..., f_F(a, r)\}$$
$$r = \{f_1(r), f_2(r), ..., f_F(r)\}$$

We use the law of cosines to yield the cosine of the angle between the vector connecting the reader to the author's baseline position, and that connecting the reader to the author's accommodated position. The angle will be greater than 90° if the author has over-accommodated, and will therefore have a negative cosine value. However, the dot product of these two vectors gives the cosine of the inner angle. Therefore, normalising by this gives a value of +/- 1 according to whether accommodation movement is less or more than the amount required to meet the reader.

Multiplying the accommodated distance by this +/-1 factor gives us a definition of an accommodation metric that expresses the accommodation as the change in directed distance from the reader, proportional to the amount required from the author's unaccommodated position. This may be greater than 1 (over-accommodation) or less than zero (divergence). In vector notation we define accommodation as:

$$Acc(a, r) = 1 - \left(\frac{|\vec{ar}|}{|\vec{\mu r}|}\right)\left(\frac{|\vec{\mu r}|^2 + |\vec{ar}|^2 - |\vec{\mu a}|^2}{2(\vec{\mu r} \cdot \vec{ar})}\right) \tag{4}$$

The dot product of $\vec{\mu r}$ and $\vec{ar}$ is zero if the two vectors are orthogonal. However this is matched by a zero value in the numerator and we take the final parentheses value in equation (4) to be 1 in this case. In the other pathological case where $|\vec{\mu r}|$ is zero, the implication is that author and reader have the same preferred position, i.e. there is nothing meaningful to say about accommodation between the two.

Having estimated author to reader accommodation, we are now in a position to estimate how readily the author adapts to others, by averaging over the set of readers. This gives us our Zelig factor, Z.

$$Zelig(a) = \frac{1}{R_a}\sum_{r=1}^{R_a} 1 - \left(\frac{|\vec{ar}|}{|\vec{\mu r}|}\right)\left(\frac{|\vec{\mu r}|^2 + |\vec{ar}|^2 - |\vec{\mu a}|^2}{2(\vec{\mu r}\cdot\vec{ar})}\right) \tag{5}$$

A positive (+) Zelig Quotient signifies the author readily accommodates, with a Zelig Quotient of 1 indicating the author always adapts their linguistic style to that of their audience. A negative (-) Zelig Quotient suggests divergence in the authors linguistic style (moving away from the audience).

Significance of values can be estimated from the variance. Here we take movement beyond one standard deviation of the authors total message distribution. The significance of an author's Zelig Quotient then follows from averaging the variance seen over the pairs the author is party to.

$$Zelig_{min}(a) = \sqrt{\frac{1}{R_a}\sum_{r=1}^{R_a}\frac{\sum_{f=1}^{F}(\sigma_f(a) - \mu_f(a))^2}{\sum_{f=1}^{F}(f(r) - \mu_f(a))^2}} \tag{6}$$

This model assumes that there are latent baseline distributions for feature values but does not suggest a generative function. Further work will determine appropriate distribution models for features, to be parameterised from the estimation methods presented here.

3 Zelig in Online Communications

To demonstrate the utility of the Zelig Quotient, our study uses scraped forum data from three large online communities (note, the names of the forums have been anonymised to protect the identity of the community members).

The first (ForumA) contains circa 2800 threads, 21000 posts, 250 currently active members and three years of historical data. The second (ForumB) contains approximately 160,000 threads, 2.25 million posts, 1500 currently active members and historical data is available for a period of approximately 10 years. The third (ForumC) contains approximately 50,000 threads, 550,000 posts, and currently 824 active members. Historical data is available for a period of approximately seven years.

All three of the online community forums are powered by vBulletin, a system which allows users to earn reputation points for their activity. Users can 'up-vote' or 'down-vote' each others' posts, which either adds or subtracts reputation points from that user. The number of reputation points received or deducted depends on who is casting their up/down-vote. Having more reputation enables a voter to have a greater influence on the reputation of others. Reputation can also be earned as the number of posts made by a user, or the age of their account, increases. This system essentially enables a power structure within the community, and is useful for differentiating between veteran communities members who contribute a lot to the community and, based on their up-votes, have a considerable amount of expertise or valuable information/opinions to share (i.e. Leaders), from relatively new and inexperienced community members whose contributions are less significant (i.e. Non-Leaders).

From each community we sample the top 10% of all members from the complete historical data (ForumA n = 70, ForumB n = 98, ForumC n = 169) based on reputation score and assign them to our 'Leader' category. For our 'Non-Leader' category we select an equally sized sample group (same n), which are evenly distributed across the remaining 90%, based on reputation score. One-time posters were removed prior to sampling.

3.1 Hypotheses

We hypothesise that, in accordance with Communication Accommodation Theory (Giles and Ogay, 2007), the Zelig Quotients for high power individuals (which we will refer to as Leaders) and low power

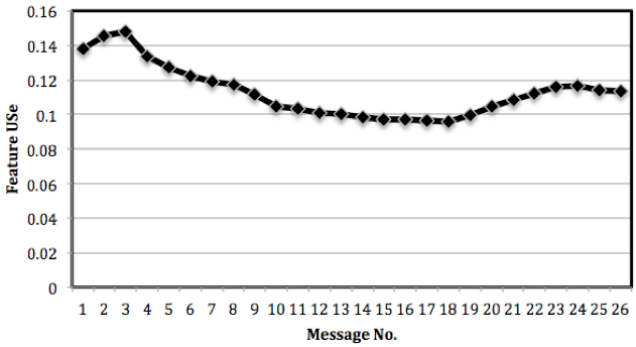


Figure 1: Moving average for an individual's linguistic feature use

individuals (referred to as Non-Leaders) will differ significantly. The power variable has been shown to have a strong influence on communication accommodation. Since people with lower status have greater cause to try to gain social approval by converging towards others, we hypothesise that:

H1: Non-Leaders (i.e. people with low power) will exhibit greater linguistic style accommodation/more Zelig-like behaviour than Leaders (people with high power)

Furthermore, it has been shown that those with low power often show greater convergence when communicating with somebody in a superior position. For example, foremen converge more to managers than to workers, and managers converge more to higher managers than to foremen (Taylor et al., 1978). Similarly, salespeople converge more to customers than vice versa, as the customers in these settings hold greater economic power (Van den Berg, 1986).

H2: People accommodate more with interlocutors who have higher power (Leaders) than those who have lower power (Non-Leaders)

3.2 Method

One challenge when working with scraped data from online community discussion threads, is accurately reconstructing who is talking to whom. Unlike in e-mail communication, vBulletin forums lack a mechanism for explicitly stating who a post is a reply to. Posters therefore append their post to the end of an ever-growing thread, regardless of whether they are addressing the first post, last post or any post in between. Of course, their communication may not even be aimed at any single person, and instead might be intended for a whole community audience.

Since the Zelig measure we have presented requires dyadic comparisons of linguistic style features, it is necessary to reconstruct a dyadic conversation structure for all of the forum threads. Previous work has examined features for accurate reply reconstruction of threaded conversations (Aumayr et al., 2011); re-building the correct structure from a collapsed conversation thread without explicit reply mechanisms. Many features are useful for reply graph reconstruction, for example: reply distance (how closely a post appears to that which it is responding), time difference (how soon after a post a response is written), quotation links (how explicit citations of previous posts are used) and cosine similarity (how closely the contents of two posts match). Aumayr et al. (2011) demonstrated that accuracy (as indicated by measurements of precision and recall) can be achieved by simply combining the use of reply distance and quotation links. That is to say, posts are typically responses to those which they appear closest to within a thread, or those which they explicitly cite. Therefore, for our analysis we treat each post as a response to the author of the closest post (the one directly preceding it within the thread) or the author that is cited within the post.

In order to calculate variations in each individuals' linguistic style, we calculate their baseline style as the moving average of their previous communications (either globally or within each particular dyad). Figure 1 illustrates this moving average for a particular LIWC feature changing with each message sent by an individual. Our moving average approach has the advantage that accommodation is calculated according to the movement towards a persons' linguistic style at a given point in time, rather than simply an average of their linguistic style in their entire communications (including those that occur later).

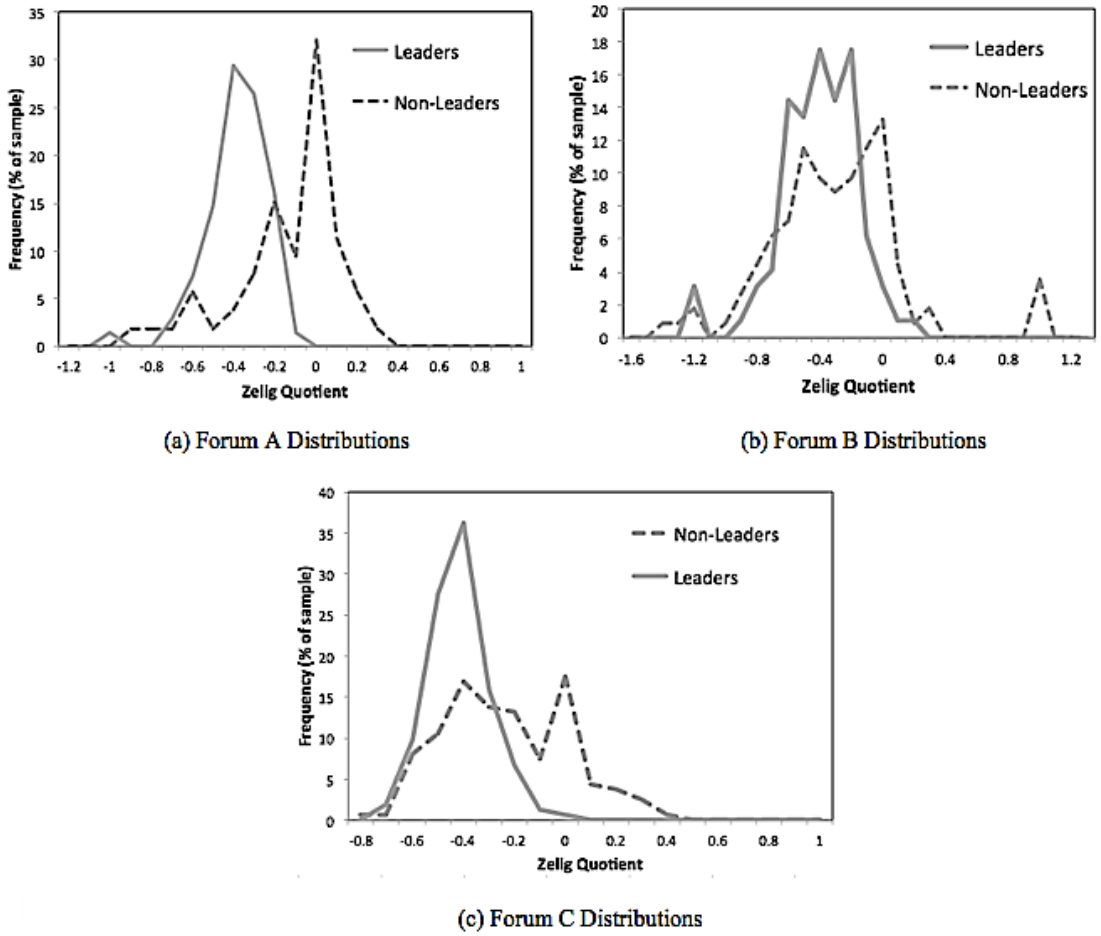


Figure 2: Zelig Quotient Distributions for members of Forum A, Forum B and Forum C

4 Results

Figure 2 shows the frequency distributions of Zelig Quotients for individuals within each of the three communities.

Our results show that the Zelig Quotients of community members follow a relatively normal distribution, centered around a Zelig Quotient of approximately -0.4 in all three communities, though there are obvious differences between the Zelig Quotient distributions for our two sample groups of Leaders vs. Non-Leaders.

Our analysis reveals that divergent communication is a common behaviour for a large proportion of each online community. That is, many community members receive a negative Zelig Quotient. Our results, therefore, go against the prevailing findings in linguistics and psychology, which suggest that individuals often constitute themselves as a community, speaking in a collective voice, and converging in terms of linguistic style.

4.1 Divergence is common

Within communication accommodation theory, convergence is generally regarded as positive and divergence as negative. Divergent communicators are often evaluated as insulting, impolite, and hostile (Bradac and Giles, 2005). Convergent speakers are evaluated as more competent, attractive, likeable and cooperative (Giles et al., 1991). Divergence is the typically the result of communicators wanting to differentiate themselves from each other and emphasize distinct identities. This can be the case particularly where there are power or status differences between interlocutors, as individuals attempt to communicate their social differences by engaging in dissimilar communication behaviours (Street, 1991). If their

points of view start deviating, so too do their communication styles (McPherson et al., 2001). Observing the Zelig distribution of a community is therefore likely to provide a valuable insight into the overall 'unity' of its members. Our results are consistent with Huffaker et al. (2006), who found increasing dissimilarity with the words used by community forum users over a six-week period. Thus, our results suggest that a large proportion of individuals within online communities have a tendency to differentiate themselves from others.

However, divergent communication is not always inherently negative. Attributions of the speaker's motives by the recipient can influence the extent to which convergent and divergent communications are perceived to be positive or negative. For instance, convergence can be evaluated as positive when attributed to speaker's internal positive motives; however convergence can also be viewed negatively when attributed to external factors. The same is true for divergent communications. When divergence is perceived by the recipient as being unintentional or positively motivated, recipients evaluate the speaker and their communications more favourably than if it is evaluated as being intentional or negatively motivated (Gasiorek and Giles, 2012). Thus, divergent communication does not neccessarily have negative connotations for the relationship between speakers and recipients.

Whilst linguistic style convergence does occur within the communities we have examined, a relatively small proportion of members (typically less than 10%) demonstrate a tendency to accommodate to others in a near Zelig-like way. A larger proportion of each community (between 15 - 35%) tend to maintain their typical linguistic style, with their Zelig Quotient of 0 indicating that fluctuations in style are due to noise rather than convergence or divergence.

The following results describe in more detail the differences in Zelig Quotients for our two sample groups, Leaders and Non-Leaders.

4.2 Non-Leaders are more Zelig-like than Leaders

We conducted independent samples t-tests in order to compare the Zelig Quotient means for Leaders and Non-Leaders in each of the online communities. There were common significant differences across all three communities: Non-Leaders in ForumA had significantly greater Zelig Quotients (M= -0.181, SD= 0.270) than Leaders (M= -0.432, SD= 0.148); $t(119)= 6.492$, $p < 0.001$. Similarly, ForumB Non-Leaders (M= -0.345, SD= 0.425) were more Zelig-like than Leaders (M= -0.461, SD= 0.254); $t(206)= 2.346$, $p < 0.05$, and ForumC Non-Leaders (M= -0.306, SD= 0.276), were also more Zelig-like than Leaders (M= -0.468, SD= 0.121); $t(327)= 6.885$, $p < 0.001$.

These results lead us to accept H1; Non-Leaders' linguistic style variation can be attributed to their accommodation towards the style of their interlocutor, to a greater degree than for Leaders. Furthermore, our results illustrate that analysis using the Zelig Quotient uncovers important accommodation trends within textual conversation data, which are in accordance with Communication Accommodation Theory.

As well as a comparison between high and low reputation groups (Leaders and Non-Leaders, respectively), a Spearman's Rank Order correlation was run to determine the relationship between reputation rank and Zelig rank within the community. We found weak but statistically significant negative correlations between reputation and Zelig Quotient within ForumB ($rs(316) = -0.1406$, $p = 0.011889$) and ForumC ($rs(179) = -0.1729$, $p = 0.019863$), further suggesting that Zelig-like behaviour is most common within the lowest reputation ranks.

4.3 Interactions between Leaders and Non-Leaders

In order to test H2 and examine how interactions between Leaders and Non-Leaders influenced accommodation, each individual within a dyad was classified as either communicating 'Up' or 'Down' the reputation hierarchy (either from Non-leader to Leader, or Leader to Non-Leader, respectively). Independent samples t-tests were conducted in order to compare the mean Zelig Quotients for dyads from each category. The tests revealed a statistically significant difference within two of the three communities, with Zelig Quotients significantly lower in 'downward' communications for ForumA - Upwards (n = 72, M= -0.283 , SD= 0.456), Downwards (n = 72, M= -0.588, SD= 0.711); $t(142)= 3.066$, $p < 0.01$, and ForumC - Upwards (n=1280, M= -0.388, SD= 0.570), Downwards (n=1280, M= -0.530, SD= 0.723); $t(2558)= 5.528$, $p < 0.001$. Within ForumB, those communicating up the hierarchy were also more

Zelig-like, however the difference was considered not to be statistically significant - Upwards (n=295, M= -0.383, SD= 0.634), Downwards (n=295, M= -0.444, SD= 0.601); t(588)= 1.208, p = 0.228.

These results lead us to accept H2; community members are more Zelig-like when communicating with people above them in terms of reputation/status.

4.4 Finding the Zeligs: Who are they?

To conclude our analysis, we address the question: 'Who are the Zelig characters within our corpora?'. By focusing our attention on the individuals that have positive $Zelig$ values ($Z > 0$), our results point to a clear and consistent answer across all three datasets - *almost all Zelig-like individuals are Non-Leaders*, however, Non-Leaders are *not* all Zeligs. Within our sample populations for each community Non-Leaders account for the vast majority of those with Zelig values greater than 0 (100% in ForumA, 91.6% in ForumB, and 100% in ForumC). These results are consistent with the idea that individuals with lower power are sensitive to the language used by a higher power interlocutor (Niederhoffer and Pennebaker, 2002); the Zelig Quotient has captured the greater tendency of Non-Leaders to alter their baseline linguistic style in order to converge with the linguistic style of Leaders, instead of the other way around.

The Zelig-like behaviour of non-leaders could potentially be attributable to their acclimatisation to community expectations; shifting their behaviour more frequently at the earliest stages of their community life and converging more towards the linguistic styles of others, until they gradually stabilise and become more attuned to the community norms, perhaps even progressing to leadership roles themselves. A useful future investigation would be to track the progress of the Zelig-like Non-Leaders over time, to see if their propensity to adapt and change their linguistic style affects their ability to progress within the community, e.g. does linguistic style convergence enable them to earn reputation more quickly, or provide an indicator of longevity within the community?

5 Conclusions and Future Work

> *"...and it shows exactly what you can do if you're a total psychotic."*
> - Leonard Zelig, Zelig (Allen, 1983)

We have presented a metric for measuring linguistic accommodation in a systematic manner, considering not only the context of an individual pair's communications, but the background models for both individuals. Thus, the Zelig Quotient provides an objective, quantifiable measure of convergence and divergence in language use between individuals, as defined by the movement in an individual's typical linguistic style towards or away from the typical linguistic style of their conversational partner. The Zelig Quotient is meaningful for differentiating between those who typically accommodate their language use towards many people (Zelig-like individuals) from those who don't. In addition, the metric includes calculation of pair-wise author to reader accommodation scores. Thus, a full picture of how an individual is behaving in terms of convergence and divergence can be gained by examining these pair-wise scores. In combination, these two scores together provide a comprehensive and illuminating picture of an individual's accommodation behaviour. This provides a framework for investigating the circumstances surrounding such variation in language use, over large datasets, in a manner which has not previously been undertaken.

We acknowledge that community forums such as these may not be the ideal dataset for evaluating dyadic accommodation, as authors may be addressing multiple people. However, it is worth noting the use of the Zelig metric seems to be effective even in this community forum dataset. Work on testing the metric on a wide range of existing datasets, including courtroom interactions and dyadic therapeutic interactions, is ongoing.

There are a number of additional possibilities for future work. Firstly, although in the current study we focussed on linguistic features that are stylistic in nature, it would be simple to alter the Zelig metric to use greater, fewer or entirely different linguistic features, for instance to explore semantic or content aspects of language. We have so far considered only a small set of linguistic style features, and it may be worth expanding this to a greater variety; in particular, concentrating on features which have been

shown to be the subject of accommodation in sociolinguistic studies (such as Bunz & Campbell, 2004). We also intend to test the metric with languages other than English in future research. Secondly, the separation between our two sample groups in terms of Zelig distributions suggests that this Quotient may be useful as a predictor of group membership/reputation score. A considerable body of work has attempted to accurately classify community members based on their communication behaviours and our results suggest that Zelig may be useful in this domain. We also intend to investigate in more detail what kinds of relationships are characterized by higher levels of accommodation, to see whether this accords with underpinning theories of politeness and social identity. Group dynamics, including linguistic accommodation to group norms in multiparty communication, and the individual's contribution to constructing a group identity, is also a large area ripe for further study.

References

Erik Aumayr, Jeffrey Chan, and Conor Hayes. 2011. Reconstruction of threaded conversations in online discussion forums. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM-11)*, pages 26–33.

Dario Benedetto, Emanuele Caglioti, and Vittorio Loreto. 2002. Language trees and zipping. *Physical Review Letters*, 88(4):1 – 4.

James Bradac and Howard Giles. 2005. Language and social psychology: Conceptual niceties, complexities, curiosities, monstrosities, and how it all works. In K. L. Fitch and R. E. Sanders, editors, *Handbook of language and social interaction*, pages 201 – 230. Mahwah, NJ: Lawrence Erlbaum Associates Publishers.

Joseph N Cappella. 1979. Talk-silence sequences in informal conversations i. *Human Communication Research*, 6(1):3–17.

Kenneth W Church. 2000. Empirical estimates of adaptation: The chance of two noriegas is closer to p/2 than p2. In *Proceedings of the 18th Conference on Computational Linguistics (COLING2000)*, volume 1, pages 180 – 186.

Noah Coccaro and Daniel Jurafsky. 1998. Towards better integration of semantic predictors in statistical language modeling. In *Proceedings of International Conference on Spoken Language Processing (ICSLP-98)*, pages 2403 – 2406.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 76 – 87. Association for Computational Linguistics.

Cristian Danescu-Niculescu-Mizil, Michael Gamon, and Susan Dumais. 2011. Mark my words!: Linguistic style accommodation in social media. In *Proceedings of the 20th international conference on World wide web*, pages 745–754.

Amit Dubey, Patrick Sturt, and Frank Keller. 2005. Parallelism in coordination as an instance of syntactic priming: evidence from corpus-based modeling. In *Proceedings of the Human Language Technology conference and the conference on Empirical Methods in Natural Language Processing*, pages 827 – 834.

Simon Garrod and Martin. J Pickering. 2004. Why is conversation so easy? *Trends in Cognitive Sciences*, 8(1):8 – 11.

Jessica Gasiorek and Howard Giles. 2012. Effects of inferred motive on evaluations of nonaccommodative communication. *Human Communication Research*, 38(3):309 – 331.

Howard Giles and Tania Ogay. 2007. Communication accommodation theory. In Bryan B Whaley and Wendy Samter, editors, *Explaining communication: Contemporary theories and exemplars*, pages 325 – 345. Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Howard Giles, Nikolas Coupland, and Justine Coupland. 1991. Accommodation theory: Communication, context, and consequence. In Howard Giles, Justine Coupland, and Nikolas Coupland, editors, *Contexts of accommodation: Developments in applied sociolinguistics*, pages 1 – 68. New York: Cambridge University Press.

Howard Giles. 1973. Accent mobility: A model and some data. *Anthropological linguistics*, 15(2):87–105.

Augusto Gnisci. 2005. Sequential strategies of accommodation: A new method in courtroom. *British Journal of Social Psychology*, 44(4):621 – 643.

David Huffaker, Joseph Jorgensen, Francisco Iacobelli, Paul Tepper, and Justine Cassell. 2006. Computational measures for language similarity across time in online communities. In *Proceedings of the HLT-NAACL 2006 Workshop on Analyzing Conversations in Text and Speech*, pages 15–22. Association for Computational Linguistics.

Molly E Ireland, Richard B Slatcher, Paul W Eastwick, Lauren E Scissors, Eli J Finkel, and James W Pennebaker. 2011. Language style matching predicts relationship initiation and stability. *Psychological Science*, 22(1):39–44.

Elizabeth Jones, Cynthia Gallois, Victor Callan, and Michelle Barker. 1999. Strategies of accommodation: Development of a coding system for conversational interaction. *Journal of Language and Social Psychology*, 18(2):123 – 152.

Adam Kilgarriff. 2001. Comparing corpora. *International Journal of Corpus Linguistics*, 6(1):97 – 133.

Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology*, pages 415–444.

Michael Natale. 1975. Convergence of mean vocal intensity in dyadic communication as function of social desirability. *Journal of Personality and Social Psychology*, 32(5):790–804.

Kate G Niederhoffer and James W Pennebaker. 2002. Linguistic style matching in social interaction. *Journal of Language and Social Psychology*, 21(4):337–360.

James W Pennebaker, Martha E Francis, and Roger J Booth. 2001. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*.

David Reitter, Frank Keller, and Johanna Moore. 2006. Computational modelling of structural priming in dialogue. In *Proceedings of the Human Language Technology conference of the North American chapter of the Association for Computational Linguistics*, pages 121– 124.

Lauren E Scissors, Alastair J Gill, and Darren Gergle. 2008. Linguistic mimicry and trust in text-based cmc. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 277–280. ACM.

Svetlana Stenchikova and Amanda Stent. 2007. Measuring adaptation between dialogs. In *Proceedings of the 8th SIGdial workshop on Discourse and Dialogue*, pages 166 – 173.

Richard L Street. 1984. Speech convergence and speech evaluation in fact-finding interviews. *Human Communication Research*, 11(2):139–169.

Richard L Street. 1991. Accommodation in medical consultations. In Howard Giles, Justine Coupland, and Nikolas Coupland, editors, *Contexts of accommodation: Developments in applied sociolinguistics*, pages 131 – 156. New York: Cambridge University Press.

Donald M Taylor, Lise M Simard, and Danielle Papineau. 1978. Perceptions of cultural differences and language use: A field study in a bilingual environment. *Canadian Journal of Behavioural Science/Revue canadienne des sciences du comportement*, 10(3):181.

Marinus Van den Berg. 1986. Language planning and language use in taiwan: social identity, language accommodation, and language choice behavior. *International journal of the sociology of language*, (59):97–116.

Michael Willemyns, Cynthia Gallois, Victor Callan, and J Pittam. 1997. Accent accommodation in the employment interview. *Journal of Language and Social Psychology*, 15(1):3 – 22.

The annotation of the Central Unit in Rhetorical Structure Trees: A Key Step in Annotating Rhetorical Relations

Mikel Iruskieta
Dept. Language and Literature Didactics
mikel.iruskieta@ehu.es

Arantza Díaz de Ilarraza
Dept. Computer Languages and Systems
a.diazdeilarraza@ehu.es

Mikel Lersundi
Dept. Basque Language and Communication
mikel.lersundi@ehu.es

IXA NLP Group, Manuel Lardizabal 1, 48014 Donostia

Abstract

This article aims to analyze how agreement regarding the central unit (macrostructure) influences agreement when establishing rhetorical relations (microstructure). To do so, the authors conducted an empirical study of abstracts from research articles in three domains (medicine, terminology, and science) in the framework of Rhetorical Structure Theory (RST). The results help to establish a new criteria to be used in RST-based annotation methodology of rhetorical relations. Furthermore, a set of verbs which can be utilized to detect the central unit of abstracts was identified and analyzed with the aim of designing a preliminary study of an automatic system for identifying the central unit in rhetorical structures.

1 Credits

This study was carried out within the framework of the following projects: IXA group, Research Group of type A (2010-2015): IT344-10 (Basque Government); SKaTeR: Scenario Knowledge Acquisition by Textual Reading: TIN2012-38584-C06-02 (Spanish Ministry of Economy and Competitiveness); Hibrido Sint: Rule-based and Statistical-based syntactic analyzers. Corpus management in an XML standard based framework: TIN2010-20218 (Spanish Ministry of Science and Innovation); TACARDI: Context-aware Machine Translation Augmented using Dynamic Resources from Internet: TIN2012-38523-C02-01 (Spanish Ministry of Science and Innovation).

2 Introduction

One of the biggest challenges in annotating the rhetorical structure of discourse has to do with the reliability of annotation. When two or more individuals annotate a text, discrepancies generally arise as a result of the way each human annotator interprets the text (Taboada and Mann, 2006). Besides, markers specifying the rhetorical relations between discourse units do not always exist (Taboada, 2006). Even if they appear in the text, these markers do not always establish rhetorical relations unequivocally (van Dijk, 1998; Mann and Thompson, 1987). Despite this ambiguity, discourse markers are considered to be a form of linguistic evidence which are used to signal coherence relations and which are useful in detecting certain rhetorical relations (Georg et al., 2009; Iruskieta et al., 2009; Pardo and Nunes, 2004).

In searching for linguistic evidence to determine the rhetorical structure of texts, scholars have analyzed not only discourse markers but also verbs. For example, Pardo and Nunes (2004) first rhetorically annotated their Corpus TCC (a Portuguese corpus containing scientific texts in the computational domain) and then analyzed verbs related to certain rhetorical relations, finding that verbs such as *buscar* 'search, look for', *objetivar* 'objectify, intend', *pretender* 'intend, mean', *procurar* 'search, look for', *servir* 'serve, meet the requirements of', and *visar* 'aim, drive' are related to the PURPOSE relation.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 466–475, Dublin, Ireland, August 23-29 2014.

They also found that other rhetorical relations such as CAUSE, EVIDENCE and RESULT are indicated by other types of verbs.

This paper aims to answer the following research questions:

(i) Does agreement about the central unit affect inter-annotator reliability when annotating rhetorical relations?

(ii) Are there some types of verbs that can be used as "indicators" (Paice, 1980) to identify the central unit of a rhetorical structure?

Besides we focus on how to identify the unit associated with the main node in the rhetorical structure tree or, in other words, the "central unit" (CU) (Stede, 2008), the "central proposition" (Pardo et al., 2003), the "central subconstituent" (Egg and Redeker, 2010) or the "salient unit of the root node" (Marcu, 1999). To our knowledge, no other research has attempted to identify this unit, the central unit of a rhetorical structure tree, by semantically studying the verb within the framework of RST. This topic, however, could have both theoretical and methodological implications.

The structure of the paper is as follows: Section 3 describes the theoretical framework, corpus and methodology utilized in this study. Section 4 lays out the results obtained. Section 5 presents a preliminary study on the semantic classes of the verbs beloging to central unit. The final section presents conclusions and suggests directions for future research.

3 Theory, corpus and methodology

3.1 Theory

Various theories describe the relational structure of a text (Asher and Lascarides, 2003; Grosz and Sidner, 1986; Mann and Thompson, 1987). This study is based on Mann and Thompson's (1987) Rhetorical Structure Theory (RST), an applied, language-independent theory that describes coherence between text fragments. It combines the idea of nuclearity –that is, the importance of an individual fragment from within the discourse– with the presence of rhetorical relations (RR) (hypotactic and paratactic relations) between these fragments. Mann and Thompson (1987) argue that nuclear units play a more important role for text coherence than satellites.

This has significant implications for automatic text summarization. Ono et al. (1994) and Rino and Scott (1996) suggest that the summary of a text can be obtained by deleting optional satellites, an argument based on the property of nuclearity in hypotactic relations. Da Cunha (2008) describes rules based on nuclearity which can be used to summarize medical texts. For a more in-depth, critical explanation of nuclearity, see Stede (2008) and for additional information on RST, see Taboada and Mann (2006) and Mann and Taboada (2010).

According to RST, hypotactic and paratactic relations connect elementary discourse units (EDUs) or groups of discourse units (span). Elementary units cannot be divided into simpler units. In this paper, a "central unit" is defined as the clause which best expresses the topic or main idea of the text. The central unit of a rhetorical structure tree is the elementary unit or group of elementary units which comprise the nucleus of its main node. Hypotactic units have a single nucleus in the central unit, while paratactic units contain multiple nuclei.

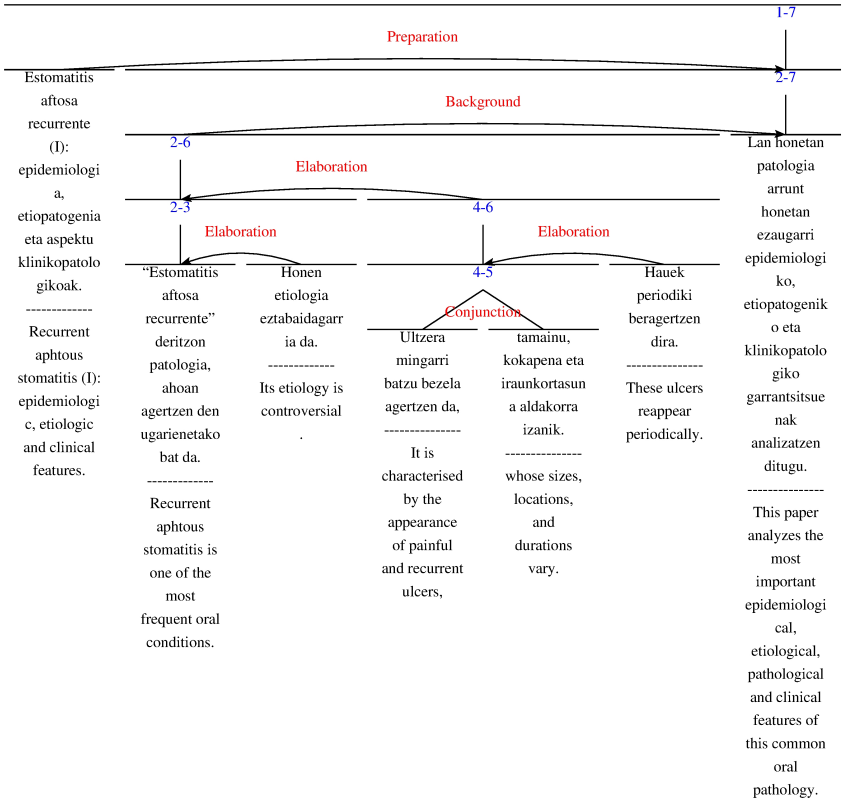
For example,[1] in the rhetorical structure tree presented in Figure 1, unit 7 is the central unit of the elementary units that are numbered from 1 to 7, since it is the nuclear unit of the root node which and has the relation PREPARATION associated to it. The root node covers the entire structure of the text, and since it is not linked to any other unit, no other associated nuclei have the same degree of central importance (Marcu, 1999). The central unit indicates the most important unit in the structure, which is indicated in Figure 1 by the verb *analizatzen* 'analyze'.

Determining nuclearity (that is, deciding which of the two associated spans has a more central role based on the intentions of the writer) is key in assigning rhetorical relations. In fact, Stede (2008) has questioned the way in which rhetorical structure is represented in RST based on several reasons:

i) It is not clear what grounds are used to make the decision: is it because of nuclearity or because of the effect of a rhetorical relation?

[1] Examples are extracted from the Basque corpus used in this study (Iruskieta, 2014).

Figure 1: A rhetorical structure tree for text GMB0301 (Annotator 1)



ii) Nuclearity poses challenges for annotation. This led Carlson et al. (2001) to present multi-nuclear versions of some nuclear relations from the classic extended classification.

We also identified the same problems. Examples (1) and (2) demonstrate how different choices of nuclearity affect agreement in rhetorical relations.

(1) [*Emaitza:*]$_1$ [*Erabiltzaileen perfil orokorra ondokoa dela esan daiteke: gizonezkoa (% 51,4), heldua (43,2 urteko media) eta patologia traumatologikoagatik kontsultatzen duena (% 50,5).*]$_2$
GMB0401
[Results:]$_1$ [The average user is as follows: male (51.4%), middle-aged (43.2 years old), and treated for trauma (50.5%).]$_2$

Annotator 1 (A1) decides that the second unit in Example (1) is more important than the first unit. The second annotator (A2), however, makes the exact opposite decision. Both annotators arrive reach their conclusions based on structural reasons. Disagreements about the importance of each text fragment influence the rhetorical relation: A1 annotates the relation as PREPARATION while A2 chooses to label the relation as ELABORATION.

Example (2) demonstrates how different interpretations of nuclearity affect agreement with regard to the rhetorical relation.

(2) [*Erabiltzaileen % 80ak bere kabuz erabakitzen dute larrialdi zerbitzu batetara jotzea*]$_1$ [*eta kontsulta hauen % 70a larritasun gutxikotzat jotzen dituzte zerbitzu hauetako medikuek.*]$_2$
GMB0401
[It is calculated that about 80% of users come to emergency services on their own initiative]$_1$ [and that 70% of visits are considered minor by health care personnel.]$_2$

A1 believes that the second unit in Example (2) provides more detailed characteristics about the users (e.g. the second unit is a satellite of the first unit) and therefore annotates the relation as hypotactic

(ELABORATION). A2, on the other hand, annotates the same discourse segment as a paratactic relation (CONJUNCTION), considering the marker *eta* 'and' to be the most significant element, indicating that she or he believes that two different elements of emergency services are being discussed.

According to Bateman and Rondhuis (1997), when determining nuclearity at the higher levels of a tree structure, RST clearly establishes a global view of a text, since an analysis is by definition incomplete until all units in the text have a function which is depicted by a single structure. It is logical that if nuclearity plays a role in determining rhetorical relations at the lower levels of a rhetorical structure, it will also affect the structure's higher levels. If two annotators have a different global point of view (e.g. they annotate different central units), they will also annotate different rhetorical relations. Therefore, our hypothesis is that trees which have the same global interpretation of text structure will have greater agreement in the annotation process; i.e., in the labeling of rhetorical relations, while those with differing global structures will have lower agreement. This hypothesis underpins the methodology used to answer the first research question of this study.

The next subsection describes the corpus used for this study.

3.2 Corpus

This study sought to analyze short but well structured texts written in Basque in order to determine linguistic evidence which could be used to indicate the central unit of rhetorical structure. The corpus utilized in this study consists of three corpora from the same genre (abstracts) from three different specialized domains: medicine, terminology and science. The communicative goal of these texts is to present specialized knowledge, since both the writer and readers are experts. Medical texts include the abstracts of all medical articles written in Basque in the *Gaceta Médica de Bilbao* (GMB) 'Medical Journal of Bilbao' between 2000 and 2008. Terminology texts are abstracts from the proceedings of the *Congreso Internacional de Terminología* (TERM) 'International Conference on Terminology' organized by UZEI –the Basque Centre for Terminology– in 1997, while scientific articles are abstracts of papers from the University of the Basque Country's *Jornadas de Investigacin de la Facultad de Ciencia y Tecnología* (ZTF) 'Research Conference of the Faculty of Science and Technology', which took place in 2008.

After the annotation process (central unit and rhetorical relations among others), the annotated corpus was evaluated (Iruskieta et al., Forthcoming) and harmonized by a judge (Iruskieta, 2014). The harmonized corpus can be consulted in the RST Basque TreeBank[2] (Iruskieta et al., 2013a).

3.3 Methodology

Before presenting the process followed to get our goals, let us explain that, when we began this research, the GMB corpus had previously been annotated manually (Iruskieta et al., 2013b) by two linguists using the extended classification of RST (Mann and Taboada, 2010) while the other two corpora (TERM and ZTF) were not tagged. The results of the comparison done about the relationship of agreement between the annotation of the central unit and the annotation of the rhetorical structure in GMB led us to redefine the annotation strategy for TERM and ZTF in the sense that we asked annotators to identify the central unit (one or more) before tagging the rhetorical structure.

The steps carried out for the annotation of the corpora were the following:

A. Elementary Discourse Units segmentation. The corpus was segmented at intra-sentential level using a minimal set of criteria (Iruskieta et al., 2011a) by each annotator using the RSTTool program (O'Donnell, 1997)

B. Central unit identification (TERM and ZTF). Both annotators determined the central unit[3] and the verbs present in the central unit of a scientific abstract in TERM and ZTF domains.[4]

[2]The RST Basque TreeBank is available at `http://ixa2.si.ehu.es/diskurtsoa/en/fitxategiak.php`.

[3]We calculate a baseline to illustrate the complexity of the central unit selection reporting the average number of EDUs: average number of 22.58 EDUs per central unit candidates per text. The average was calculated based on the number of EDUs, over the number of texts.

[4]The central units (CU) can be consulted also in RST Basque TreeBank.

C. Rhetorical tree structure annotation. Rhetorical relations were annotated by each annotator using the RSTTool program with the extended classification (Mann and Taboada, 2010) of RST.

D. Evaluation. Agreement in rhetorical tree structures were manually evaluated following the qualitative methodology proposed in Iruskieta et al. (Forthcoming), but taking into account the structures with the same central unit and distinguishing between the rhetorical relations linked or not to central unit.

E. Interpretation. We compared the results of central unit agreement and disagreemens to check for possible correlations using a t-test formula at 99.5% confidence.

4 Results

Our main hypothesis is that an agreement on central unit leads us to a higher agreement on rhetorical relations; in other words, identifying the main idea of the text helps the human annotator in the identification of the structure of the text and, therefore, the agreement between annotators is higher.[5]

4.1 Correlation between agreement on rhetorical relations and agreement on central unit

The observation made about the GMB, where we argued that annotators agree more on rhetorical relations when they annotated the same central unit, remained after considering results of a more extended corpus with two new corpora (TERM and ZTF) and two additional annotators.

Results confirm this fact even when the difference has been substantially reduced from 0.1497 to 0.0426 when more data (all the corpus) were considered. Table 1 presents the global results of the comparison between the agreement on central unit ('= CU')[6] and mean agreement on rhetorical relations for the corpus as a whole.

	GMB			**Corpus**		
	= CU	**≠ CU**	**Diff.**	**= CU**	**≠ CU**	**Diff.**
Mean	0.7456	0.5959	**0.1497**	0.5915	0.5489	**0.0426**
SD	0.1833	0.1749		0.1429	0.1125	

Table 1: Mean agreement (and standard deviation) of the central unit and rhetorical relations

We perform a significant test for the differences. We confirmed that the populations being compared have a normal distribution following the Kolmogorov-Smirnov test (p-value of K-S test was 0.913) and have the same variance (p-value of F-test was 0.063). Therefore, two tail independent samples t-test was used with a 0.013 p-value, denying the null hypothesis.

Other hypothesis and combinations were analyzed with positive results: a significant agreement was observed when we compared agreement in rhetorical relation linked to central unit when annotators tagged the same central unit and when they tagged different central units. It is very difficult to establish which rhetorical relation are linked to central unit when annotators do not tag the same central unit.

4.2 Correlation between agreement on rhetorical relations linked or not to central unit

After our main hypothesis was confirmed, we went ahead in the tree structure and we checked whether there is higher agreement in rhetorical relations linked to the central unit (considering the structures where there was agreement in central unit), than in the other relations of the tree structure. For example, in the rhetorical structure tree presented in Figure 1, we consider two relations linked to central unit PREPARATION (1>2-7) and BACKGROUND (2-6>7), while the other four relations are not linked to central unit (ELABORATION (2<3), ELABORATION (2-3<4-6), ELABORATION (4-5<6) and CONJUNCTION (4=5). Table 2 presents the results of relations linked to central unit with relation not linked to central unit:

In structures with the same central unit we compare between the agreement in rhetorical relations linked to the central unit and all the other relations. Percent agreement is substantially higher when we

[5]The results of all corpora considered indicate that the change in methodology improved central unit agreement between annotators slightly in TERM and ZTF. This highlights the benefits of a first step followed in TERM and ZTF which entails detecting the central unit.

[6]And '≠ CU' for disagreement on central unit.

	GMB			Corpus		
	Linked	**Not**	**Diff.**	**Linked**	**Not**	**Diff.**
Mean	0.7454	0.5881	**0.1573**	0.7179	0.5449	**0.1730**
SD	0.2695	0.3344		0.2107	0.1850	

Table 2: Comparison between rhetorical relations linked and no-linked to central unit in structures with the same central unit

observe the relations linked to the central unit: 17.3% higher than the agreement on the relations that are not linked to the central unit. Populations being compared follow a normal distribution (p-value of K-S test was 0.93) but they do not have the same variance (p-value of F-test is 0.09). The result of the null hypothesis was rejected (p-value of t-test was smaller than 0.001), so we can establish a correlation. The average rhetorical relation agreement on a text according to the central unit, is no different to the average percentage of agreement in the rhetorical relations linked to the UC to those not linked.

4.3 Discussion of results

To illustrate the results on agreement (or not) on central unit and average agreement on rhetorical relations linked (or not) from Tables 1 and 2, we present comparisons of the populations in Figure 2:

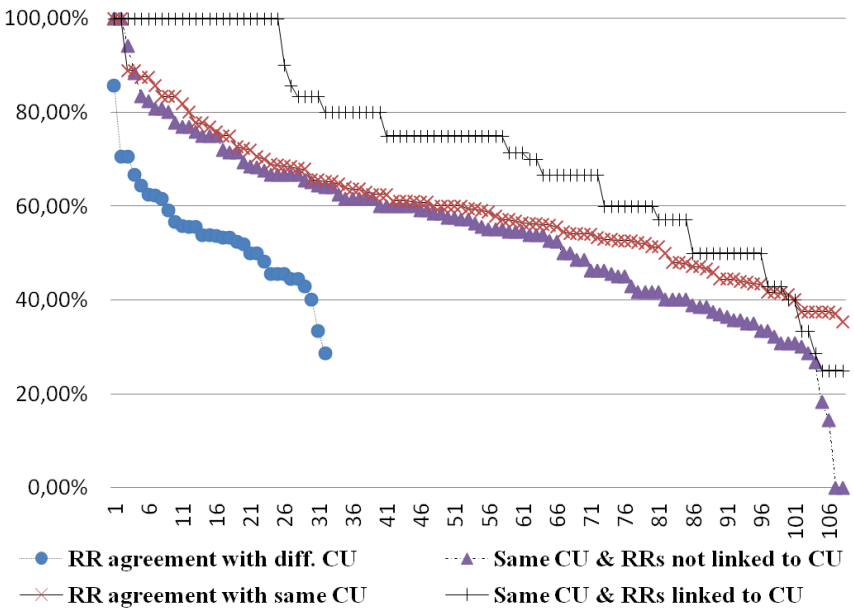
a. When the central unit was the same, the average agreement on relations is represented with red crosses.

b. When the central unit was different, the average agreement on relations is represented with blue circles.

c. When the central unit was the same and the relations are linked to central unit with black crosses.

d. When the central unit was the same and the relations are not linked to central unit with violet triangles.

Figure 2: Representation of mean agreement between RR (vertical) and the number of relations considered in a structure (horizontal) according to the central unit.



These results help to answer the first research question of this study and seem to indicate that there is a correlation between these two kinds of agreement: i) greater agreement on detecting the central unit correlates with greater agreement on the annotation of rhetorical relations (results from Table 1 are ilustrated in Figure 2 comparing the distance of the red croses [a] with blue circles [b]), ii) also on those which are linked to the central unit (results from Table 2 are ilustrated in Figure 2 comparing the distance

of the black croses $[c]$ with the violet triangles $[d]$).

This analysis leads to two conclusions:

i) When considering the methodology for labeling rhetorical structure, annotating the central unit is an important first step before labeling rhetorical relations at least in short texts such as abstracts.

ii) In Computational Linguistics, a process which helps to automatically identify the central unit is important for determining some restrictions in rhetorical structure mainly determined by the genre/domain structure.

In order to discuss these results, first of all we have to consider that the central unit is a nuclear unit and that relations are linked at various levels (intra-sentential level and inter-sentential level); there are more relations linked at inter-sentential level. For example, in Figure 1 two relations linked to central unit are only at inter-sentential level. This seems to show that these results (rhetorical relations linked to central unit) are not so trivial, since the degree of agreement expected at higher level tree structures is lower. In other words, the agreement at lower levels is higher than in the high level. For example, Marcu and Echihabi (2002) argue that automatic annotation of certain rhetorical relations should be addressed first at intra-sentential level because they are less ambiguous. Soricut and Marcu (2003) mention that some of the rhetorical relations are derived from syntactic structures. These results (11.50% higher agreement at intra-sentential level, than at inter-sentential level in the GMB corpus) were confirmed in Basque by Iruskieta et al. (2011b).

5 Identifying the semantic class of verbs in the central unit

Our final goal is the automatic detection of central unit. To this end, we wanted to find lexical-semantic markers in the central unit[7] in each domain in greater detail. The meanings of the main verbs were analyzed and their semantic class determined as per the SUMO ontology (Niles, 2003). The relation between meaning and semantic class was obtained by means of the MCR semantic database, which includes various lexical-semantic and ontological databases. Data from the GMB, TERM, and ZTF corpora are grouped in Table 3 by semantic classes at the most general level, e.g. "Intentional Psychological Process" (IPP), "Social Interaction" (SI), "Internal Change" (IC) and "Predicate".

SUMO	SUMO	MCR synset	GMB	TERM	ZTF
IP-IPP	Reasoning	analyze$_1$, show$_2$, base$_1$	**0.4615**	0.2273	0.0870
	Comparing	value$_2$, compare$_1$	**0.2692**		
	Classifying	classify$_1$			0.0870
	Learning	review$_1$	0.0385		
	Guiding	take$_3$		0.0455	
	Process	gain$_4$			0.1739
IP-IPP		recognize$_2$, determine$_8$, hold$_6$, focus$_1$	0.0385	0.0909	0.0435
IP-SI	Communication	present$_2$, addres$_9$, recount$_1$, propose$_1$	0.0385	**0.4545**	0.0435
IP		perform$_1$, target$_1$, set-up$_{15}$, work$_1$, make$_3$, use$_1$	0.1154	0.0909	0.0870
IP	Searching-Investigating	investigate$_1$			0.0435
IP	Organizational Process	serve$_2$			0.0435
IC		palliate$_2$		0.0455	
Predicate		be$_1$, develop$_5$, constitute$_1$, hold$_4$	0.0385	0.0455	0.3913

Table 3: Summary comparison of verbs by domain

The results of this empirical study indicate that each domain tends to use verbs from the same semantic class. For example, in the GMB corpus, the central unit was usually marked with verbs from the IPP category. On the other hand, in the TERM corpus, verbs from the IPP and SI category. Verbs in the central unit of the ZTF corpus are marked with IPP and Predicate class.

Therefore, the results demonstrate that:

[7]Results show that there are multiple EDUs functioning as the central unit of the text in the three corpora: 9 multiple EDU functioning as central unit in GMB, 2 multiple EDUs in TERM and 3 multiple EDUs in ZTF.

i) A study is needed to identify the SUMO class of the verbs used in a specific domain. For example in our corpus the central units is indicated with verbs that belong to the IPP class for all three domains. However, other classes also have to be considered, SI for TERM and Predicate for ZTF.

ii) In the case of weak verbs, other indicators[8] help to identify the central unit. The TERM and ZTF corpora are more marked by noun class indicators than the GMB corpus (Iruskieta, 2014). Another reason is that the direct observation of the central unit makes the central unit selection more consistent. An evidence of that is that all the verbs in central unit are from the same SUMO class in TERM and ZTF corpora by both annotators. Futhermore, it could also be argued that the use of different verbs has to do not only with the field but also with the medium: the GMB corpus derives from texts published in a periodical while the TERM and ZTF corpora include texts published in Conference proceedings. In other words, it could be argued that the medium influences the writing style, and consequently, impacts the verb classes used in the texts. This is in line with the main argument of this study, since different verbs are used to indicate the central unit in the TERM and ZTF corpora, which share the same medium but belong to different fields.

So far, this paper has provided a partial answer to the second research question. However, to automatically detect the central unit by means of verbs (with the help of other types of signals) it is necessary to consider these three issues:

i) The verb form which is used in the central unit might also be used in non-central units in the rhetorical structure tree.

ii) Tools which disambiguate the sense of analyzed verbs are necessary in order to know what SUMO class they belong to.[9]

iii) The central unit is not always indicated with a verb and, therefore, other types of signals (or combinations) can help in the automatic identification of the central unit.

The next phase of this research considered whether verb forms which appear in the central unit unequivocally indicate this unit or whether they can also appear in other types of units. This entailed calculating the frequency with which each studied verb appeared and counting the percentage of appearances which correspond to the central unit.

From the results obtained so far we can't establish any clear tendency but rather some preliminary conclusions that must be ratified with the analysis of more data.

Phenomena related to the central unit appeared in this study of ambiguity:

i) In GMB corpus verbs that indicate the central unit with a high enough frequency are from IPP category *baloratu* 'value2'; there exist other verbs that can be considered but they are not so frequent, e.g. *alderatu* 'compare$_1$', *gainbegiratu* 'review$_1$', *aztertu* and *analizatu* 'analyze$_1$', and *ezagutu* 'recognize$_2$'.

ii) In TERM corpus, the second sense of the verb present in MCR, 'present$_2$' (its equivalents in Basque are the verbs *plazaratu*, *aurkeztu*, *aipatu*, *berri eman* and *jardun*), has a high frequency but a high degree of ambiguity. We can't identify the central unit on the basis of its occurrence.

iii) In the ZTF corpus, the central unit was not always indicated with a verb.

6 Conclusions and future research

After considering the relationship between identifying the central unit in a text and annotating its rhetorical structure, it has been demonstrated that a correlation exists between these two tasks, since a greater degree of agreement with regard to the central unit leads to a greater degree of agreement in rhetorical. Besides there is more agreement in rhetorical relations linked to the central units than in relations that are not linked.

This study has investigated verbs which mark the central unit of a rhetorical structure and the correlation of the agreement in central unit with the agreement in rhetorical relations. Its goal has been

[8] According to Paice (1980) indicators can be nouns ('paper', 'method', 'result'), determiners ('this', 'the', 'a') and pronouns ('we', 'I'), among others.

[9] In attempting to automatically detect coherence relations which are not indicated or vaguely indicated using WordNet (Miller et al., 1990) Sporleder and Lascarides (2007) obtained better results using morphological strategies than using semantic generalization strategies. This is due to the fact that, as far as we know, NLP has yet to focus on disambiguating words.

to consider aspects which are relevant for establishing a methodology to help set general criteria for identifying the central unit of texts.

This study also considered which verbs appear in the central units, their semantic classes (according to SUMO categories), and how they identify the central unit. Verbs used to indicate the central units vary in different domains: in the GMB corpus, the central unit was more frequently and the least ambiguously indicated with verbs from the IPP category (SUMO), while in the TERM, SI verbs were most frequent and the least ambiguous.

Testing these results in a larger corpus (and different domains and text structures) could lead to applications for automatic text summarization tasks (classifying clauses), since the central unit is the most important unit in the text.

Furthermore, this study has explained the steps to automatically detect the central unit based on the ambiguity of the verb which marks the central unit. More studies about other indicators (and their combinations) are necessary to automatically detect the central unit.

References

[Asher and Lascarides2003] Asher, Nicholas and Alex Lascarides. 2003. *Logics of conversation*. Cambridge Univ Pr, Cambridge.

[Bateman and Rondhuis1997] Bateman, John A. and Klaas Jan Rondhuis. 1997. Coherence relations: Towards a general specification. *Discourse Processes*, 24(1):3–49.

[Carlson et al.2001] Carlson, Lynn, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In *2nd SIGDIAL Workshop on Discourse and Dialogue, Eurospeech 2001*, Aalborg, Denmark, 1-2 September. Association for Computational Linguistics.

[da Cunha2008] da Cunha, Iria. 2008. Hacia un modelo lingüístico de resumen automático de artículos médicos en español. Phd-thesis, IULA, Universitat Pompeu Fabra.

[Egg and Redeker2010] Egg, Markus and Gisela Redeker. 2010. How complex is discourse structure? In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, page 16191623, Valletta, Malta, 19-21 May.

[Georg et al.2009] Georg, Georg, Hugo Hernault, Marc Cavazza, Helmut Prendinger, and Mitsuru Ishizuka. 2009. From rhetorical structures to document structure: shallow pragmatic analysis for document engineering. In *9th ACM symposium on Document engineering*, pages 185–192, Munich, Germany, 16-18 September. ACM.

[Grosz and Sidner1986] Grosz, Barbara J. and Candance L. Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational linguistics*, 12(3):175–204.

[Iruskieta et al.2009] Iruskieta, Mikel, Arantza Diaz de Ilarraza, and Mikel Lersundi. 2009. Correlaciones en euskera entre las relaciones retóricas y los marcadores del discurso [Correlations between rhetorical relations and discourse markers]. In *27th AESLA Conference*, pages 963–971, Ciudad Real, Spain.

[Iruskieta et al.2011a] Iruskieta, Mikel, Arantza Diaz de Ilarraza, and Mikel Lersundi. 2011a. Bases para la implementación de un segmentador discursivo para el euskera [Bases for an Implementation of a Discourse Parser for Basque]. In *Workshop A RST e os Estudos do Texto*, Mato Grosso, Brazil, 24-26 October.

[Iruskieta et al.2011b] Iruskieta, Mikel, Arantza Diaz de Ilarraza, and Mikel Lersundi. 2011b. Unidad discursiva y relaciones retóricas: un estudio acerca de las unidades de discurso en el etiquetado de un corpus en euskera. *Procesamiento del Lenguaje Natural*, 47:144.

[Iruskieta et al.2013a] Iruskieta, Mikel, Mara Jesus Aranzabe, Arantza Diaz de Ilarraza, Itziar Gonzalez, Mikel Lersundi, and Oier Lopez de la Calle. 2013a. The RST Basque TreeBank: an online search interface to check rhetorical relations. In *4th Workshop "RST and Discourse Studies"*, Brasil, October 21-23.

[Iruskieta et al.2013b] Iruskieta, Mikel, Arantza Diaz de Ilarraza, and Mikel Lersundi. 2013b. Establishing criteria for RST-based discourse segmentation and annotation for texts in Basque. *Corpus Linguistics and Linguistic Theory*, 0(0):132.

[Iruskieta et al.Forthcoming] Iruskieta, Mikel, Iria da Cunha, and Maite Taboada. Forthcoming. A qualitative comparison method for rhetorical structures: Identifying different discourse structures in multilingual corpora. *Language Resources and Evaluation*.

[Iruskieta2014] Iruskieta, Mikel. 2014. Pragmatikako erlaziozko diskurtso-egitura: deskribapena eta bere ebaluazioa hizkuntzalaritza konputazionalean (a description of pragmatics rhetorical structure and its evaluation in computational linguistic). Phd-thesis, Euskal Herriko Unibertsitatea, Donostia. http://ixa2.si.ehu.es/~jibquirm/tesia/tesi_txostena.pdf.

[Mann and Taboada2010] Mann, Willian C. and Maite Taboada. 2010. RST web-site. http://www.sfu.ca/rst/.

[Mann and Thompson1987] Mann, Willian C. and Sandra A. Thompson. 1987. Rhetorical Structure Theory: A Theory of Text Organization. *Text*, 8(3):243–281.

[Marcu and Echihabi2002] Marcu, Daniel and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 368–375. Association for Computational Linguistics.

[Marcu1999] Marcu, Daniel, 1999. *Discourse trees are good indicators of importance in text*, pages 123–136. Advances in Automatic Text Summarization. MIT, Cambridge.

[Miller et al.1990] Miller, George A., Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of lexicography*, 3(4):235–244.

[Niles2003] Niles, Ian. 2003. Mapping WordNet to the SUMO ontology. In *Proceedings of the IEEE International Knowledge Engineering conference*, pages 23–26.

[O'Donnell1997] O'Donnell, Michael. 1997. RSTTool: An RST analysis tool. In *Proceedings of the 6th European Workshop on Natural Language Generation*, Duisburg, Germany.

[Ono et al.1994] Ono, Kenjl, Kazuo Sumita, and Seijl Miike. 1994. Abstract generation based on rhetorical structure extraction. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 344–348. Association for Computational Linguistics.

[Paice1980] Paice, Chris D. 1980. The automatic generation of literature abstracts: An approach based on the identification of self-indicating phrases. In *3rd annual ACM conference on Research and development in information retrieval*, pages 172–191, Cambridge, June. Butterworth and Co.

[Pardo and Nunes2004] Pardo, Thiago A. S. and Maria G. V. Nunes. 2004. Relações Retóricas e seus Marcadores Superficiais: Análise de um Corpus de Textos Científicos em Português do Brasil [Rhetorical relations and its surface markers: an analysis of scientific texts corpus in Portuguese of Brazil]. Technical Report NILC-TR-04-03.

[Pardo et al.2003] Pardo, Thiago A. S., Lucia H. M. Rino, and Maria G. V. Nunes. 2003. GistSumm: A summarization tool based on a new extractive method. *Computational Processing of the Portuguese Language*, pages 196–196.

[Rino and Scott1996] Rino, Lucia H. M. and Donia R. Scott. 1996. A discourse model for gist preservation. *Advances in Artificial Intelligence*, pages 131–140.

[Soricut and Marcu2003] Soricut, R. and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, volume 1, pages 149–156. Association for Computational Linguistics.

[Sporleder and Lascarides2007] Sporleder, Caroline and Alex Lascarides. 2007. Exploiting linguistic cues to classify rhetorical relations. In *Recent Advances in Natural Language Processing*, pages 532–539, Borovets, Bulgaria, 27-29 September.

[Stede2008] Stede, Manfred, 2008. *RST revisited: Disentangling nuclearity*, pages 33–57. 'Subordination' versus 'coordination' in sentence and text. John Benjamins, Amsterdam and Philadelphia.

[Taboada and Mann2006] Taboada, Maite and Willian C. Mann. 2006. Rhetorical Structure Theory: looking back and moving ahead. *Discourse Studies*, 8(3):423–459.

[Taboada2006] Taboada, Maite. 2006. Discourse markers as signals (or not) of rhetorical relations. *Journal of Pragmatics*, 38(4):567–592.

[van Dijk1998] van Dijk, Teun A. 1998. *Texto y contexto: semántica y pragmática del discurso*. Cátedra.

Measuring Lexical Cohesion: Beyond Word Repetition

Anna Kazantseva & Stan Szpakowicz
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Ontario, Canada
{ankazant,szpak}@eecs.uottawa.ca

Abstract

This paper considers the problem of finding topical shifts in documents and in particular at what information can be leveraged to identify them. Recent research on topical segmentation usually assumes that topical shifts in discourse are signalled by changes in vocabulary. This information, however, is not always a sufficient indicator of a topical shift, especially for certain genres. This paper explores an additional source of information. Our hypothesis is that the type of a referring expression is an indicator of how accessible its antecedent is. The shorter and less informative the expression (*e.g.,* a personal pronoun *versus* a lengthy post-modified noun phrase), the more accessible the antecedent is likely to be and the more likely it is that the topic under discussion has remained constant between the two mentions. We explore how this information can be used to augment a lexically-based topical segmenter. We test our hypothesis on two types of data, literary narratives and lecture notes. The results suggest that our similarity metric is useful: depending on the settings it either slightly improves the performance or leaves it unchanged. They also suggest that certain types of referring expressions are more useful than others.

1 Introduction

In the past 10 years, research on topical segmentation has mostly centred on using surface vocabulary to identify topical shifts. The intuition is that if the vocabulary changes perceptibly, so does *the topic* under discussion. One popular way to model this assumption is by probabilistic graphical models. A document may be modelled as a sequence of strings (*e.g.,* sentences) generated by a latent topic variable, where the topic variables correspond to distributions over a finite vocabulary. Similarity-based methods are an alternative methodology. The segmenter explicitly measures the amount of lexical similarity between sentences. Places where similarity is low are likely to indicate shifts of topic. The common thread among these approaches is that they rely almost exclusively on the explicitly mentioned words.

The idea that vocabulary shifts indicate topical shifts dates back to Youmans (1991). Indeed, by and large, introducing new concepts almost necessarily requires that the concepts be named and described. How densely the concepts are explicitly mentioned and how often the mentions are repeated depends to a large degree on the genre and on the cognitive complexity of the document. In scientific papers or legal documents clarity is paramount, so the author will endeavour to state things explicitly and avoid ambiguity. The less complicated the document, however, the less it is necessary to explicitly repeat terminology. In literature, for example, word repetition is not only uncommon, but it is usually a sign of bad writing. In casual conversations, the topic can easily be never mentioned explicitly. How can we identify topical shifts in a document whose author does not "hold the reader's hand"?

It turns out that lexical cohesion (or, put simply, word repetition) is only one of several devices of cohesion (Halliday and Hasan, 1976, p. 29) Other possibilities are reference, substitution, ellipsis and conjunction. In this paper we mainly explore referential cohesion.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 476–485, Dublin, Ireland, August 23-29 2014.

Figure 1: An example dialogue from the *Moonstone* corpus

"What's wrong now?" I said once more.

"Rosanna's late again for dinner," says Nancy. "And I'm sent to fetch her in. All the hard work falls on my shoulders in this house. Let me alone, Mr. Betteredge!"

The person here mentioned as Rosanna was our second housemaid. "Where is she?" I inquired. [. . .]

"At the sands, of course!" says Nancy, with a toss of her head. "She had another of her fainting fits this morning, and she asked to go out and get a breath of fresh air. I have no patience with her!"

"Go back to your dinner, my girl," I said. "I have patience with her, and I'll fetch her in."

Figure 1 shows a snippet of a dialogue from the publicly available *Moonstone* corpus (Kazantseva and Szpakowicz, 2012). The two speakers discuss a specific person, *Rosanna*, yet her name is mentioned explicitly only twice. In the remainder of the dialogue the author uses pronouns to refer to this person, whose identity is evident from the context. Running an automatic segmenter on such a document would likely be challenging since focal concepts – characters – are often referred to by pronouns or definite noun phrases (NPs) instead of explicit repetition.

The focal entity *Rosanna* is introduced once and then it is referred to by nominal and pronominal anaphora, not by explicit repetition. Simplifying things somewhat, we can say that merely by the virtue of encountering a referring expression (*e.g., she* or *the person*), we know that it refers to something that must be clear from the context. The type of the referring expression also contains information about the availability of the antecedent. A *she* implies that the 'she' in question is rather obvious, that is to say, the antecedent is nearby and, more important for our purposes, the topical thread continues. A more verbose referring expression (*e.g., the woman in red*) is more likely in situations where the antecedent is less obvious and the reader needs additional information to disambiguate the expression.

The idea that the type of referring expression tells a lot about the accessibility of its antecedent dates back to Givón (1981). He postulated that the more informative the referring expression is, the less accessible the antecedent will be. Figure 2 shows the list of expressions from the least to the most informative. Projecting this information onto our task, we can say that the more informative the expression is, the less continuity there will be in the topic.

The main contribution of this work is to show how such information can be used to improve the quality of text segmentation. We extract NPs and classify them by informativeness. This is achieved with the help of a syntactic parser, but a lighter form of processing might do, perhaps even if it captured personal pronouns. Using this information, we augment and correct a matrix of lexical similarities between sentences, a structure frequently used as an input to a topical segmenter.

The results of using coreferential similarity are evaluated on a dataset of manually segmented chapters from a novel (Kazantseva and Szpakowicz, 2012) and on transcripts of lectures in Artificial Intelligence (Malioutov and Barzilay, 2006). We try the new similarity matrix on two publicly available similarity-based segmenters APS (Kazantseva and Szpakowicz, 2011) and MinCutSeg (Malioutov and Barzilay, 2006). The results suggest that the new matrix never hurts, and in several case improves, the performance of the segmenter, especially for the novel. We also check whether this metric would still be useful if instead of the traditionally used lexical similarity we used a similarity metric which took synonymy into account. In this case, the margin of improvement is lower, but still the coreferential similarity metric never hurts the performance and often improves it.

Section 2 of the paper gives an overview of related work. Section 3 describes our similarity metric and how we compute it. Section 4 shows the details of the experiments, while Section 5 discusses the results. We conclude in Section 6 with a discussion of how our metric can be improved and simplified.

2 Background and related work

Much of research on topical segmentation of text is based on the idea that changes of topic are usually accompanied by vocabulary changes. Introduced by Youmans (1991), it has since formed the backbone

of research on topical segmentation. We now briefly review recent work on text segmentation. Since the focus of this research is on what information is useful for text segmentation, this review emphasizes representations rather than algorithms.

Perhaps the simplest way of estimating topical similarity between sentences is to measure cosine similarity between corresponding feature vectors. It has been used extensively in text segmentation. Hearst (1994; 1997) describes TextTiling, an algorithm which identifies topical shifts by sliding a window through the document and measures cosine similarity between adjacent windows. The drops in similarity signal shifts of topic. More recently, Malioutov and Barzilay (2006) as well as Kazantseva and Szpakowicz (2011) use graph cuts and factor graph clustering for text segmentation. Both systems rely on cosine similarity between bag-of-word vectors as an underlying representation.

While cosine similarity between vectors is easy to compute, it is hardly a reliable metric of topical similarity. Several researchers have used *lexical chains* – first introduced by Halliday and Hasan (1976) – to improve the performance of topical segmenters.[1] The intuition behind using lexical chains for text segmentation is that the beginning and end of a chain tend to correspond to the beginning and end of a topically cohesive segment. One version of TextTiling (Hearst, 1997) uses lexical chains manually constructed using Roget's Thesaurus. Okumura and Honda (1994) apply automatically created lexical chains to segment a small set of documents in Japanese. More recently, Marathe (2010) tried to build lexical chains using distributional semantics and apply the method to text segmentation.

Other proposals to move beyond word repetition in topical segmentation include the use of bigram overlap in (Reynar, 1999), information about collocations in (Jobbins and Evett, 1998), LSA (Landauer and Dumais, 1997) in (Choi et al., 2001; Olney and Cai, 2005) and WordNet in (Scaiano et al., 2010).

It should be noted that much of the recent work on topical segmentation revolves around generative models. For example Blei and Moreno (2001) use HMM,while Eisenstein and Barzilay (2008), Misra et al. (2011) and Du et al. (2013) use higher-order models. We do not review this work in detail here because it centers on algorithms for text segmentation and not on the information supplied to those algorithms, which is the focus of this research. Fundamentally, the text is modelled as a sequence of tokens generated by latent topic variables. Although probabilistic segmenters can be extended to use additional information (*e.g.,* Eisenstein and Barzilay (2008) augment their segmenter with information about discourse markers), it is not trivial to change these models to include information such as synonymy, co-reference and so on. That is why we do not review them in detail here.

As this brief review shows, a number of approaches have been proposed to measure cohesion between sentences, that is to say, to describe to what extent a pair of sentences is "about the same thing". Most of them have a common denominator: they use explicit lexical information, sometimes augmented by semantic relations from thesauri or ontologies.

Lexical resources, such as ontologies and knowledge-bases, may help improve the quality of segmentations, but such resources are not always available. They also may cause problems with precision. More important, however, they do not solve a more fundamental problem: a text may be highly cohesive and coherent without being tightly bound by either lexical cohesion or synonymy.

The main ideas developed in this work originate in (Givón, 1981). The author looks at the functional domain of topical accessibility. A number of coding devices affect this property. They are listed in Figure 2, ordered from the devices used to mark the most continuous topics to those which mark the least continuous topics. The order in Figure 2 is governed by a simple principle: the more accessible the topic is, the less information is used to code it. The author argues that the continuum is applicable in many languages. He also mentions that while the exact values of the phenomenon in question are difficult to predict or even estimate, their relative order can be predicted with certainty, even if some devices are unavailable in some languages.

In a similar spirit, Ariel (2014) groups non-initial NPs into expressions with *low accessibility* (definite NPs and proper names), those with *intermediate accessibility* (personal and demonstrative pronouns) and those with *high accessibility* (pronouns).

In this work, we propose to leverage the presence and type of co-referential relations to improve

[1] Very simply put, a lexical chain is a sequence of related words in a text.

the results of two recent similarity-based segmenters. Instead of resolving anaphoric references, we assume that their mere presence often indicates topic continuity. With this augmented model, we segment fiction and spoken lecture transcripts, the two types of data where low rates of lexical cohesion preclude achieving segmentation of good quality using only surface information about token types.

3 Estimating coreferential similarity

In order to see whether knowledge about types of referential expressions is useful for measuring topical similarity, we incorporate this information into two publicly available similarity-based topical segmenters, *MCSeg* (Malioutov and Barzilay, 2006) and *APS* (Kazantseva and Szpakowicz, 2011). Normally, both *MCSeg* and *APS* measure similarity between sentences by computing cosine similarity between the vectors corresponding to bag-of-words representation for each sentence:

$$sim(s_1, s_2) = \frac{s_1 \bullet s_2}{||s_1|| \times ||s_2||} \quad (1)$$

Each atomic unit of text is represented as a vector of features corresponding the occurrences of each token type. The vectors are weighted using $tf.idf$ values for each token type. Next, a segmenter measures cosine similarity between vectors according to Equation 1. That is the fundamental representation in both *MCSeg* and *APS*. *MCSeg* identifies segment boundaries by creating a weighted cyclic graph and cutting it so as to maximize the sum of edges within segments and to minimize the sum of severed edges. *APS* segments the sequence by finding segment centres – points which best capture the content of a segment – and assigning data points to best segment centres so as to maximize net similarity.

The proposed similarity metric relies on the following idea: in order to measure how many concepts two sentences share, we do not need to resolve anaphoric expressions in full, but only to map them onto sentences which contain their most recent antecedent (without actually naming the antecedents). We do that by parsing the documents with the Connexor parser (Tapanainen and Järvinen, 1997) and extracting all NPs with their constituents. Next, we attempt to classify the NPs into categories which would roughly correspond to those listed in Figure 2 and to those in (Ariel, 2014).

A manual study by Brown (1983) suggests that the average referential distance for animate and inanimate entities differs widely within the same document.[2] That is why it makes sense to distinguish between these two types. In the end, then, we classify each identified NP into one of the categories listed in Figure 3. The list is not exhaustive and in some cases an NP may belong to more than one type. In practice, however, an NP is always assigned a single type dictated by the implementation.

[2]Brown (1983, pp. 323-324) compares referring expressions which denote human and non-human entities. She uses three measurements: average distance to the nearest antecedent, average ambiguity and persistence. On all three counts, human and non-human entities appear to have different distributions.

Figure 2: Linguistic coding devices which signal topic accessibility (Givón, 1981)

Most continuous (least surprising)

1. zero anaphora
2. unstressed pronouns (*e.g., He* was speaking loudly.)
3. right-dislocated definite noun phrases (NPs) (*e.g.,* It is no good, *that book*.)
4. neutral-ordered definite NPs (*e.g., That book* is no good.)
5. left-dislocated definite NPs (*e.g., That book*, it is no good.)
6. Y-moved NP's (*e.g., The book* they read in turns.)
7. cleft/focus constructions (*e.g.,* It was *that book*, that was on her mind for weeks.)
8. referential indefinite NPs (*e.g.,* He picked up *a book* and left.)

Least continuous (most surprising)

Figure 3: Categories of noun phrases taken into account when computing coreferential similarity

1. animate personal pronouns (*he, she, they*)
2. inanimate pronouns (*it*)
3. demonstrative pronouns (*that, those*)
4. animate proper names (*John Hernecastle*)
5. inanimate proper names (*London*)
6. animate definite noun phrases (*the man*)
7. inanimate definite noun phrases (*the jewel*)
8. animate indefinite noun phrases (*a man*)
9. inanimate indefinite noun phrases (*a jewel*)

Finally, coreferential similarity between sentences S_i and S_j is measured as follows:

$$coref\_sim(S_i, S_j) = \left(\frac{\sum_{t=0}^{|T|} count_t^{S_j} \times weight_t}{|S_1| \times |S_2|}\right)^{(j-i-1)\times decayFactor} \quad (2)$$

T is the set of of all types of referring expressions which we consider – those given in Figure 3. $count_t^{S_j}$ is the number of times when an expression of type t appears in the most recent sentence, S_j. Note that we only consider the referring expressions in the most recent sentence, because a referring expression, by its nature, must refer to something previously mentioned. The "tightness" of the link is controlled by setting $weight_t$ for each expression type t. $weight_t$ effectively specifies how likely it is that the antecedent for an expression of a type t appears in sentence s_i. The values of the weights are set experimentally on the holdout data. They can almost certainly be further fine-tuned. Intuitively, the settings of the weights reflect the logic behind Givón's theory. Consider an example vector of weights for expressions, where a higher weight corresponds to a more accessible antecedent (for animate and inanimate entities respectively).

<personal_pronouns_anim: 4, demonstr_pronouns_anim: 2, proper_names_anim: 1,
def_np_anim: 0.5, indef_np_anim: 0, pronouns_inanim: 2, demonstr_pronouns_inanim: 2,
proper_names_inanim: 0, def_np_inanim: 0, indef_np_inanim: 0>

The denominator of Equation 2 normalizes the value by the product of the lengths of sentences S_1 and S_2. The exponent $(j - i - 1) \times decayFactor$ is responsible for decreasing similarity as the distance between sentence S_i and S_j increases. The decay factor, $0 < decayFactor < 1$, is set experimentally, and $j - i$ is the distance between sentences S_i and S_j, $i < j$.

Figure 4 contains a walk-through example of computing referential similarity between two sentences.

The coreferential similarity as defined by Equation 2 is rather limited. The first limitation is the range: it can only measure similarity between nearby sentences or paragraphs, because it only makes sense between the closest occurrences of an antecedent and a subsequent referring expression. For example, it does not make sense to measure coreferential similarity between sentences that are several paragraphs apart. Even if they indeed talk about the same entities, the topic has most likely been re-introduced several times in between. That is why we only compute coreferential similarity for sentences no more than $decayWindow$ sentences apart. The value of $decayWindow$ is usually between 2 and 6 and it is set experimentally on the holdout set for each corpus.

The values of $coref_sim$ are usually quite small and the information used is rather one-sided. We use it, therefore, in addition to, not instead of, lexical similarity. In our experiments, we first compute lexical similarity between sentences (or paragraphs) and then modify the lexical matrix by adding to it the matrix of coreferential similarity.

Figure 4: An example of computing coreferential similarity

$$coref\_sim(S_i, S_j) = \left(\frac{\sum_{t=0}^{|T|} count_t^{S_j} \times weight_t}{|S_1| \times |S_2|}\right)^{(j-i-1)\times decayFactor}$$

S1: "At the sands, of course!" says Nancy, with a toss of her head.
S2: "She had another of her fainting fits this morning, and she asked to go out and get a breath of fresh air."

Expression counts:	Weights:
personal_pronouns_anim: 2 (she, she)	4
demonstr_pronouns_anim: 0	2
proper_names_anim: 1	1
def_np_anim: 0	0.5
indef_np_anim: 0	0
pronouns_inanim: 0	2
demonstr_pronouns_inanim: 1	2
proper_names_inanim: 0	0
def_np_inanim: 2 (this morning, fainting fits)	0
indef_np_inanim: 1 (a breath)	0

$$coref\_sim(S_2, S_1) = \frac{2 \times 4 + 1 \times 1 + 1 \times 2}{21 \times 22}^{(2-1-1)\times 0.5} = 0.0234$$

4 Experimental results

The effectiveness of coreferential similarity metric has been tested in practice. A set of experiments compared how much the metric improves the quality of topical segmentations. To this end, we ran *APS* and *MCSeg* with and without adding coreferential similarity to lexical similarity, and compared the results. We chose these segmenters for comparison because $coreferential\_similarity$ can only be naturally incorporated into a similarity-based segmenter.

Data. In our experiments we used two publicly available datasets. The first one is a set of lectures on Artificial Intelligence (Malioutov and Barzilay, 2006). The dataset contains 22 documents which were manually annotated for the presence of topical shifts. The second dataset is the *Moonstone* dataset described in (Kazantseva and Szpakowicz, 2012). It contains 20 chapters from Wilkie Collins's novel, each annotated by 4-6 people. To reconcile these multiple reference annotations, we create a majority gold standard. It only contains segment breaks which were marked by at least 30% of the annotators. Both segmenters are compared against this gold standard. There is a fair amount of disagreement between the annotators of this dataset. The average inter-annotator *windowDiff* is 0.38 (Kazantseva and Szpakowicz, 2012, pp. 215-216), but if one takes into account near-hits, then at least 50% of the boundaries are marked by more than two annotators.

Both datasets are quite challenging. The lecture dataset contain a lot of rather informal speech and there is not as much lexical repetition as would be in a more formal text. The *Moonstone* dataset is an example of literary language, full of small digressions, dialogue and so on.

The first dataset is annotated at the level of individual sentences. The second dataset is annotated at the level of paragraphs. We segment both datasets at the level of the gold-standard annotations (sentences for lectures, paragraphs for the novel).

When working with paragraphs, $coref\_sim$ is computed slightly differently:

$$coref\_sim(p_i, p_j) = \left(\frac{\sum_{t=0}^{|T|} count_t^{p_j} \times weight_t}{|p_1| \times |p_2|}\right)^{(j-i-1)\times decayFactor} \quad (3)$$

In this case, $count_t^{p_j}$ refers to the number of occurrences of expression of type t in the first $paragraphCutOff$ sentences of the paragraph p_j, instead of the whole paragraph. The rationale behind this heuristic is that the referring expressions in the opening sentences of the paragraph are likely to refer

to entities from the previous paragraph, while expressions in the middle or the end of the paragraph are likely to refer to entities introduced inside the paragraph.

Segmenters and baselines. We use two publicly available topical segmenters in our experiments: *MCSeg* and *APS*. The default version of each segmenter computes a similarity matrix between sentence in the input document. The values in the matrix correspond to cosine similarity (Equation 1) computed after the removal of stop words and weighting the bag-of-word vectors by $tf.idf$. The results obtained using these default matrices are our first baseline.

In our experiments, we modify this matrix by adding to it the matrix of coreferential similarities. The values of coreferential similarities are rather small and most modifications are localized. That is because the value of $decayWindow$ is set between 2 and 6 (see Section 3).

In addition to the matrices based on cosine similarity, we wanted to see if using a more intelligent measure of topical similarity improves the results. We built one more flavour of similarity matrices using the *DKPro Similarity* framework (Bär et al., 2013). The framework contains a model of textual similarity which has been used by the winning system at the SemEval Textual Similarity 2012 shared evaluation. We use this model (further *STS-2012*) as a more competitive baseline for computing topical similarity.

The *STS-2012* baseline consists of a log-linear regression model trained on the SemEval 2012 training data. It combines an assortment of measures of textual similarity to come up with its judgments. The metrics include n-gram overlap, semantic similarity measures (based on both corpora and lexical resources) and several measures of stylistic similarity. We chose to use this relatively complicated metric because of its competitive performance at SemEval 2012. The system, however, was not designed to measure topical similarity *per se*, especially between many sentences coming from the same source document. By default, the *STS-2012* baseline outputs values between 1 and 5. These were normalized to be between 0 and 1.

Similarly to the experimental design with cosine similarity matrices, we try running the segmenters using *STS-2012* with and without adding coreferential similarity matrix to it.

On both datasets we set the weights for various types of referential expressions using hold-out sets of two files. When setting the weights, we were guided by the principle captured in Figure 2: personal pronouns suggest the tightest link, followed by demonstrative pronouns, proper names, and so on.

It should be noted that because we had to modify the native representation of both segmenters by supplying a matrix computed using non-native code, we could not use the proper training scripts which come with the segmenters. In effect, the results are likely to be lower than they could have been. Even so, this is acceptable for our purposes because we are interested in the improvement gained by using coreferential similarity, not in obtaining the best possible segmentation via the setting of the best parameters.

Processing. We computed the underlying lexical similarity matrices using the same procedure as described in (Malioutov and Barzilay, 2006; Kazantseva and Szpakowicz, 2011), but using our own code. In other words, we built a matrix of cosine similarities after removing stop words and weighting the underlying vectors by $tf.idf$ values.

In order to compute coreferential similarity, all documents were parsed using the Connexor parser (Tapanainen and Järvinen, 1997). The parser was chosen because it produces high-quality partial parses of long sentences often encountered in the *Moonstone* dataset. We also tagged named entities and labelled NPs as animate or inanimate using the Stanford Core NLP suite.[3]

Metrics. We compare topical segmentations using the *windowDiff* metric:

$$winDiff = \frac{1}{N-k} \sum_{i=1}^{N-k} (|ref - hyp| \neq 0) \tag{4}$$

windowDiff slides a window of size k through the input sequence of length N. At every position of the window, the metric compares the number of boundaries in the reference sequence and in the hypothetical sequence. The number of erroneous windows is normalized by the total number of windows to obtain the final value. *windowDiff* is a penalty metric: lower values correspond to better segmentations.

[3] `http://nlp.stanford.edu/software/corenlp.shtml`

	AI Lectures	*Moonstone*
APS	0.420 (± 0.014)	0.441 (± 0.075)
APS-$coref\_sim$	0.411 (± 0.025)	0.391 (± 0.060)
APS-STS	0.428 (± 0.049)	0.479 (± 0.041)
APS-STS -$coref\_sim$	0.429 (± 0.020)	0.478 (± 0.035)
MCSeg	0.431 (± 0.045)	0.470 (± 0.095)
MCSeg-$coref_sim$	0.410 (± 0.060)	0.413 (± 0.030)
MCSeg-STS	0.451 (± 0.023)	0.441 (± 0.051)
MCSeg-STS-$coref\_sim$	0.433 (± 0.070)	0.430 (± 0.025)

Table 1: Results of comparing *APS* and *MCSeg* using four different matrix types (*windowDiff* values and standard deviation)

5 Evaluation

Table 1 presents the results of running *APS* and *MCSeg* using four different input matrices each. The first column shows the combination of the name of the segmenter and the specific input matrix. *APS* and *MCSeg* refer to the cases where both segmenters were run using simple cosine similarity matrices. STS refers to matrices computed using *STS-2012* from the *DKPro Similarity* framework. $coref\_sim$ refers to cosine similarity matrices modified by adding a matrix with coreferential similarities. $STS-coref\_sim$ are matrices computed using *STS-2012* which had coreferential similarity added to them.

In all experiments, we set the weights for different types of referring expressions on two hold-out files. The remainder of the data is divided into five folds. Standard deviation reported in the tables is computed across folds.

Coreferential similarity improves the results of the cosine matrix for both segmenters, but the improvement on the AI dataset is rather small (1% for *APS* and 2% for *MCSeg*).

It is interesting to see that in most cases using STS matrices slightly hurts the performance of the segmenters compared to using simple cosine similarity matrices. The only exception is running *MCSeg* on the *Moonstone* dataset which improves the performance by 3%.

Adding a matrix of coreferential similarities to STS matrices slightly improves the performance on the *Moonstone* dataset and leaves it practically unchanged on the dataset of AI lectures.

It is somewhat surprising that using *STS-2012* for similarity computation does not improve, and occasionally worsens, the results compared to using simple cosine similarity. Coreferential similarity, on the other hand, produces a small but consistent improvement.

We have examined the vectors of weights used in these experiments (set using hold-out data). On the *Moonstone* dataset, the results are the best when personal animate pronouns get the highest weight, followed by demonstrative animate pronouns, as well as inanimate pronouns, both regular and demonstrative. Other expression types are assigned either a very small weight or the value 0, effectively making them inconsequential. We hypothesize that this is due to the fact that the novel discusses people, their relations and interactions, making animate entities central for estimating topical links.

The vectors used on the AI lecture dataset are similar, except that here the highest weights are given to demonstrative and regular inanimate pronouns. These are followed by demonstrative and then regular animate pronouns. This distribution is likely due to the fact that the lecture dataset discusses abstract concepts, while people are likely to be noted more tangentially. We are not sure how to explain the fact that in this dataset demonstrative pronouns have a slightly higher weight than the regular ones.

Identifying and categorizing noun phrases requires either high-quality NP-tagging or parsing. On the other hand, most pronouns can be captured very easily, perhaps even using a list of words. It is interesting to note that the most gain is due to these "cheap" types of referring expressions. In the future, we plan to implement a lighter version of the coreferential similarity metric which only considers pronouns.

6 Conclusions and future work

This paper has presented a method for improving the quality of topical segmentations by using information about referential expressions in nearby sentences. The method slightly improves the quality of segmentations and, what is even more important, seems never to worsen the results.

The necessity to perform complete parsing of the input document is a drawback of the current approach. We note in Section 5, however, that the only types of referential expressions which improve performance are personal and demonstrative pronouns. Those can be easily captured without parsing. In the near future we plan to investigate such a light-weight version of $coref\_sim$ metric.

Another way to improve our current implementation would be a more objective method of setting the weights for different types of referring expressions. At present, the expressions are set by hand on a small hold-out set of documents. This is far from ideal. We plan to investigate if using logistic regression or expectation maximization would make the system more robust.

References

Mira Ariel. 2014. *Accessing Noun-Phrase Antecedents*. Routledge, London and New York.

Daniel Bär, Torsten Zesch, and Iryna Gurevych. 2013. DKPro Similarity: An Open Source Framework for Text Similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 121–126, Sofia, Bulgaria.

David Blei and Pedro Moreno. 2001. Topic segmentation with an aspect hidden Markov Model. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 343–348.

Elizabeth Brown. 1983. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Freddy Y. Y. Choi, Peter Wiemer-Hastings, and Johanna Moore. 2001. Latent Semantic Analysis for Text Segmentation. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 109–117, Pittsburgh, Pennsylvania.

Lan Du, Wray Buntine, and Mark Johnson. 2013. Topic Segmentation with a Structured Topic Model. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200, Atlanta, Georgia.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian Unsupervised Topic Segmentation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 334–343, Honolulu, Hawaii.

Talmy Givón. 1981. Typology and Functional Domains. *Studies in Language*, 5(2):163–193.

M.A.K Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London and New York.

Marti A. Hearst. 1994. Multi-paragraph Segmentation of Expository Text. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, ACL '94, pages 9–16, Las Cruces, New Mexico.

Marti A. Hearst. 1997. TextTiling: segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

Amanda C. Jobbins and Lindsay J. Evett. 1998. Text Segmentation Using Reiteration and Collocation. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, COLING '98, pages 614–618, Montréal, Québec.

Anna Kazantseva and Stan Szpakowicz. 2011. Linear Text Segmentation Using Affinity Propagation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 284–293, Edinburgh, Scotland.

Anna Kazantseva and Stan Szpakowicz. 2012. Topical Segmentation: a Study of Human Performance and a New Measure of Quality. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 211–220, Montréal, Canada.

Thomas K Landauer and Susan T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, pages 211–240.

Igor Malioutov and Regina Barzilay. 2006. Minimum Cut Model for Spoken Lecture Segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, Sydney, Australia.

Meghana Marathe. 2010. Lexical Chains Using Distributional Measures of Concept Distance. Master's thesis, University of Toronto.

Hemant Misra, François Yvon, Olivier Cappé, and Joemon M. Jose. 2011. Text segmentation: A topic modeling perspective. *Information Processing and Management*, 47(4):528–544.

Manabu Okumura and Takeo Honda. 1994. Word Sense Disambiguation and Text Segmentation Based On Lexical Cohesion. In *COLING 1994 Volume 2: The 15th International Conference on Computational Linguistics*, pages 775–761, Kyoto, Japan.

Andrew Olney and Zhiqiang Cai. 2005. An Orthonormal Basis for Topic Segmentation in Tutorial Dialogue. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing –HLT '05*, pages 971–978, Vancouver, Canada.

Jeffrey C. Reynar. 1999. Statistical Models of Text Segmentation. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 357–364.

Martin Scaiano, Diana Inkpen, Robert Laganière, and Adele Reinhartz. 2010. Automatic Text Segmentation for Movie Subtitles. In *Advances in Artificial Intelligence*, volume 6085 of *Lecture Notes in Computer Science*, pages 295–298. Springer.

Pasi Tapanainen and Timo Järvinen. 1997. A non-projective dependency parser. *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64–71.

Gilbert Youmans. 1991. A new tool for discourse analysis: The vocabulary-management profile. *Language*, 67(4):763–789.

Fast Tweet Retrieval with Compact Binary Codes

Weiwei Guo* **Wei Liu†** **Mona Diab‡**
*Computer Science Department, Columbia University, New York, NY, USA
†IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
‡Department of Computer Science, George Washington University, Washington, D.C., USA
weiwei@cs.columbia.edu weiliu@us.ibm.com mtdiab@gwu.edu

Abstract

The most widely used similarity measure in the field of natural language processing may be cosine similarity. However, in the context of Twitter, the large scale of massive tweet data inevitably makes it expensive to perform cosine similarity computations among tremendous data samples. In this paper, we exploit binary coding to tackle the scalability issue, which compresses each data sample into a compact binary code and hence enables highly efficient similarity computations via Hamming distances between the generated codes. In order to yield semantics sensitive binary codes for tweet data, we design a binarized matrix factorization model and further improve it in two aspects. First, we force the projection directions employed by the model nearly orthogonal to reduce the redundant information in their resulting binary bits. Second, we leverage the tweets' neighborhood information to encourage similar tweets to have adjacent binary codes. Evaluated on a tweet dataset using hashtags to create gold labels in an information retrieval scenario, our proposed model shows significant performance gains over competing methods.

1 Introduction

Twitter is rapidly gaining worldwide popularity, with 500 million active users generating more than 340 million tweets daily[1]. Massive-scale tweet data which is freely available on the Web contains rich linguistic phenomena and valuable information, therefore making it one of most favorite data sources used by a variety of Natural Language Processing (NLP) applications. Successful examples include first story detection (Petrovic et al., 2010), local event detection (Agarwal et al., 2012), Twitter event discovery (Benson et al., 2011) and summarization (Chakrabarti and Punera, 2011), etc.

In these NLP applications, one of core technical components is tweet similarity computing to search for the desired tweets with respect to some sample tweets. For example, in first story detection (Petrovic et al., 2010), the purpose is to find an incoming tweet that is expected to report a novel event not revealed by the previous tweets. This is done by measuring *cosine* similarity between the incoming tweet and each previous tweet.

One obvious issue is that cosine similarity computations among tweet data will become very slow once the scale of tweet data grows drastically. In this paper, we investigate the problem of searching for most similar tweets given a query tweet. Specifically, we propose a binary coding approach to render computationally efficient tweet comparisons that should benefit practical NLP applications, especially in the face of massive data scenarios. Using the proposed approach, each tweet is compressed into short-length binary bits (*i.e.*, a *compact binary code*), so that tweet comparisons can be performed substantially faster through measuring Hamming distances between the generated compact codes. Crucially, Hamming distance computation only involves very cheap NOR and popcount operations instead of floating-point operations needed by cosine similarity computation.

Compared to other genres of data, similarity search in tweet data is very challenging due to the short nature of Twitter messages, that is, a tweet contains too little information for traditional models to extract

[1] http://en.wikipedia.org/wiki/Twitter

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 486–496, Dublin, Ireland, August 23-29 2014.

Symbol	Definition
n	Number of tweets in the corpus.
d	Dimension of a tweet vector, *i.e.*, the vocabulary size.
$\boldsymbol{x}_i$	The sparse *tf-idf* vector corresponding to the i-th tweet in the corpus.
$\bar{\boldsymbol{x}}_i$	The vector subtracted by the mean $\boldsymbol{\mu}$ of the tweet corpus: $\bar{\boldsymbol{x}}_i = \boldsymbol{x}_i - \boldsymbol{\mu}$.
$X, \bar{X}$	The tweet corpus in a matrix format, and the zero-centered tweet data.
r	The number of binary coding functions, *i.e.*, the number of latent topics.
f_k	The k-th binary coding function.

Table 1: Symbols used in binary coding.

latent topical semantics. For instance, in our collected dataset, there exist only 11 words per tweet on average. We address the sparsity issue pertaining to tweet data by converting our previously proposed topic model*Weighted Textual Matrix Factorization* (WTMF) (Guo and Diab, 2012) to a binarized version. WTMF maps a tweet to a low-dimensional semantic vector which can easily be transformed to a binary code by virtue of a sign function. We consider WTMF a good baseline for the task of tweet retrieval, as it has achieved state-of-the-art performance among unsupervised systems on two benchmark short-text datasets released by Li et al. (2006) and Agirre et al. (2012).

In this paper, we improve WTMF in two aspects. The first drawback of the WTMF model is that it focuses on exhaustively encoding the local context, and hence introduces some overlapping information that is reflected in its associated projections. In order to remove the redundant information and meanwhile discover more distinct topics, we employ a gradient descent method to make the projection directions nearly orthogonal.

The second aspect is to enrich each tweet by its neighbors. Because of the short context, most tweets do not contain sufficient information of an event, as noticed by previous work (Agarwal et al., 2012; Guo et al., 2013). Ideally, we would like to learn a model such that the tweets related to the same event are mapped to adjacent binary codes. We fulfill this purpose by augmenting each tweet in a given training dataset with its neighboring tweets within a temporal window, and assuming that these neighboring (or similar) tweets are triggered by the same event. We name the improved model *Orthogonal Matrix Factorization with Neighbors* (OrMFN).

In our experiments, we use Twitter hashtags to create the gold (*i.e.*, groundtruth) labels, where tweets with the same hashtag are considered semantically related, hence relevant. We collect a tweet dataset which consists of 1.35 million tweets over 3 months where each tweet has exactly one hashtag. The experimental results show that our proposed model OrMFN significantly outperforms competing binary coding methods.

2 Background and Related Work

2.1 Preliminaries

We first introduce some notations used in this paper to formulate our problem. Suppose that we are given a dataset of n tweets and the size of the vocabulary is d. A tweet is represented by all the words it contains. We use notation $\boldsymbol{x} \in \mathbb{R}^d$ to denote a sparse d-dimensional *tf-idf* vector corresponding to a tweet, where each word stands for a dimension. For ease of notation, we represent all n tweets in a matrix $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n] \in \mathbb{R}^{d \times n}$. For binary coding, we seek r binarization functions $\left\{ f_k : \mathbb{R}^d \rightarrow \{1, -1\} \right\}_{k=1}^{r}$ so that a tweet $\boldsymbol{x}_i$ is encoded into an r-bit binary code (*i.e.*, a string of r binary bits). Table 1 illustrates the symbols used in this paper for notation.

Hamming Ranking: In the paper we evaluate the quality of binary codes in terms of Hamming ranking. Given a query tweet, all data items are ranked in an ascending order according to the Hamming distances between their binary codes and the query's binary code, where a Hamming distance is the number of bit positions in which bits of two codes differ. Compared with cosine similarity, computing Hamming distance can be substantially efficient. This is because fixed-length binary bits enable very cheap logic operations for Hamming distance computation, whereas real-valued vectors require floating-point op-

erations for cosine similarity computation. Since logic operations are much faster than floating-point operations, Hamming distance computation is typically much faster than cosine similarity computation[2]

2.2 Binary Coding

Early explorations of binary coding focused on using random permutations or random projections to obtain binary coding functions (aka, hash functions), such as Min-wise Hashing (MinHash) (Broder et al., 1998) and Locality-Sensitive Hashing (LSH) (Indyk and Motwani, 1998). MinHash and LSH are generally considered *data-independent* approaches, as their coding functions are generated in a randomized fashion. In the context of Twitter, the simple LSH scheme proposed in (Charikar, 2002) is of particular interest. Charikar proved that the probability of two data points colliding is proportional to the angle between them, and then employed a random projection $\boldsymbol{w} \in \mathbb{R}^d$ to construct a binary coding function:

$$f(\boldsymbol{x}) = \text{sgn}\big(\boldsymbol{w}^\top \boldsymbol{x}\big) = \begin{cases} 1, & \text{if } \boldsymbol{w}^\top \boldsymbol{x} > 0, \\ -1, & \text{otherwise.} \end{cases} \quad (1)$$

The current held view is that *data-dependent* binary coding can lead to better performance. A data-dependent coding scheme typically includes two steps: 1) learning a series of binary coding functions with a small amount of training data; 2) applying the learned functions to larger scale data to produce binary codes.

In the context of tweet data, Latent Semantic Analysis (LSA) (Landauer and Dumais, 1997) can directly be used for data-dependent binary coding. LSA reduces the dimensionality of the data in X by performing singular value decomposition (SVD) over X: $X = U\Sigma V^\top$. Let $\bar{X}$ be the zero-centered data matrix, where each tweet vector $\boldsymbol{x}_i$ is subtracted by the mean vector $\boldsymbol{\mu}$, resulting in $\bar{\boldsymbol{x}}_i = \boldsymbol{x}_i - \boldsymbol{\mu}$. The r coding functions are then constructed by using the r eigenvectors $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r$ associated with the r largest eigenvalues, that is, $f_k(\boldsymbol{x}) = \text{sgn}\big(\boldsymbol{u}_k^\top \bar{\boldsymbol{x}}\big) = \text{sgn}\big(\boldsymbol{u}_k^\top (\boldsymbol{x} - \boldsymbol{\mu})\big)$ $(k = 1, \cdots, r)$. The goal of using zero-centered data $\bar{X}$ is to balance 1 bits and -1 bits.

Iterative Quantization (ITQ) (Gong and Lazebnik, 2011) is another popular unsupervised binary coding approach. ITQ attempts to find an orthogonal rotation matrix $R \in \mathbb{R}^{r \times r}$ to minimize the squared quantization error: $\|B - RV\|_{\text{F}}^2$, where $B \in \{1, -1\}^{r \times n}$ contains the binary codes of all data, $V \in \mathbb{R}^{r \times n}$ contains the LSA-projected and zero-centered vectors, and $\|\cdot\|_{\text{F}}$ denotes Frobenius norm. After R is optimized, the binary codes are simply obtained by $B = \text{sgn}(RV)$.

Much recent work learns nonlinear binary coding functions, including Spectral Hashing (Weiss et al., 2008), Anchor Graph Hashing (Liu et al., 2011), Bilinear Hashing (Liu et al., 2012b), Kernelized LSH (Kulis and Grauman, 2012), etc. Concurrently, supervised information defined among training data samples was incorporated into coding function learning such as Minimal Loss Hashing (Norouzi and Fleet, 2011) and Kernel-Based Supervised Hashing (Liu et al., 2012a). Our proposed method falls into the category of *unsupervised, linear, data-dependent* binary coding.

2.3 Applications in NLP

The NLP community has successfully applied LSH in several tasks such as first story detection (Petrovic et al., 2010), and paraphrase retrieval for relation extraction (Bhagat and Ravichandran, 2008), etc. This paper shows that our proposed data-dependent binary coding approach is superior to data-independent LSH in terms of the quality of generated binary codes.

Subercaze et al. (2013) proposed a binary coding approach to encode user profiles for recommendations. Compared to (Subercaze et al., 2013) in which a data unit is a whole user profile consisting of all his/her Twitter posts, we tackle a more challenging problem, since our data units are extremely short – namely, a single tweet.

[2]We recognize that different hardware exploiting techniques such as GPU or parallelization accelerate cosine similarity. However, they don't change the inherent nature of the data representation. They can be equally applied to Hamming distance and we anticipate significant speed gains. We relegate this exploration of different implementations of Hamming distance to future work.

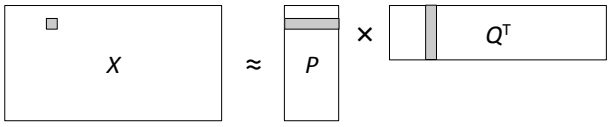


Figure 1: Weighted Textual Matrix Factorization. The $d \times n$ matrix X is approximated by the product of a $d \times r$ matrix P and an $n \times r$ matrix Q. Note in the figure we used the transpose of the Q matrix.

3 Weighted Textual Matrix Factorization

The WTMF model proposed by Guo and Diab (2012) is designed to extract latent semantic vectors for short textual data. The low-dimensional semantic vectors can be used to represent the tweets in the original high-dimensional space. WTMF achieved state-of-the-art unsupervised performance on two short text similarity datasets, which can be attributed to the fact that WTMF carefully handles missing words (the missing words of a text are the words with 0 values in a data vector $\boldsymbol{x}$).

Assume that there are r latent dimensions/topics in the data, the matrix X is approximated by the product of a $d \times r$ matrix P and an $n \times r$ matrix Q, as in Figure 1. Accordingly, a tweet $\boldsymbol{x}_j$ is represented by an r-dimensional vector $Q_{j,\cdot}$; similarly, a word w_i is generalized by the r-dimensional vector $P_{i,\cdot}$ (the ith row in matrix P). The matrix factorization scheme has an intuitive explanation: the inner-product of a word profile vector $P_{i,\cdot}$ and a tweet profile vector $Q_{j,\cdot}$ is to approximate the TF-IDF value X_{ij}: ${P_{i,\cdot}}^\top Q_{j,\cdot} \approx X_{ij}$ (as illustrated by the shaded parts in Figure 1).

Intuitively, $X_{ij} = 0$ suggests that the latent topics of the text $\boldsymbol{x}_j$ are not relevant to the word w_i. Note that 99% of the cells in X are 0 because of the short contexts, which significantly diminishes the contribution of the observed words to the searching of optimal P and Q. To reduce the impact of missing words, a small weight w_m is assigned to each 0 cell of X in the objective function:

$$\sum_i \sum_j W_{ij} \left({P_{i,\cdot}}^\top Q_{j,\cdot} - X_{ij} \right)^2 + \lambda ||P||_2^2 + \lambda ||Q||_2^2,$$
$$W_{i,j} = \begin{cases} 1, & \text{if } X_{ij} \neq 0, \\ w_m, & \text{if } X_{ij} = 0. \end{cases} \tag{2}$$

where λ is the regularization parameter. Alternating Least Squares (Srebro and Jaakkola, 2003) is used to iteratively compute the latent semantic vectors in P and Q:

$$P_{i,\cdot} = \left(Q^\top \tilde{W}^{(i)} Q + \lambda I \right)^{-1} Q^\top \tilde{W}^{(i)} X_{i,\cdot}^\top,$$
$$Q_{j,\cdot} = \left(P^\top \tilde{W}^{(j)} P + \lambda I \right)^{-1} P^\top \tilde{W}^{(j)} X_{\cdot,j} \tag{3}$$

where $\tilde{W}^{(i)} = \text{diag}(W_{i,\cdot})$ is a $n \times n$ diagonal matrix containing the i-th row of the weight matrix W. Similarly, $\tilde{W}^{(j)} = \text{diag}(W_{\cdot,j})$ is a $d \times d$ diagonal matrix containing the j-th column of W.

As in Algorithm 1 line 6-9, P and Q are computed iteratively, i.e., in a iteration each $P_{i,\cdot}(i = 1, \cdots, d)$ is calculated based on Q, then each $Q_{j,\cdot}(j = 1, \cdots, n)$ is calculated based on P. This can be computed efficiently since: (1) all $P_{i,\cdot}$ share the same $Q^\top Q$; similarly all $Q_{j,\cdot}$ share the same $P^\top P$; (2) X is very sparse. More details can be found in (Steck, 2010).

Adapting WTMF to binary coding is straightforward. Following LSA, we use the matrix P to linearly project tweets into low-dimensional vectors, and then apply the sign function. The k-th binarization function uses the k-th column of the P matrix ($P_{\cdot,k}$) as follows

$$f_k(\boldsymbol{x}) = \text{sgn}\left(P_{\cdot,k} \bar{\boldsymbol{x}}\right) = \begin{cases} 1, & \text{if } P_{\cdot,k} \bar{\boldsymbol{x}} > 0, \\ -1, & \text{otherwise.} \end{cases} \tag{4}$$

4 Removing Redundant Information

It is worth noting that there are two explanations of the $d \times r$ matrix P. The rows of P, denoted by $P_{i,\cdot}$, may be viewed as the collection of r-dimensional latent profiles of words, which we observe frequently

Algorithm 1: OrMF

Procedure P = `OrMF` $(X, W, \lambda, n\_itr, \alpha)$
$n\_words, n\_docs \leftarrow$ size(X);
randomly initialize P, Q;
$itr \leftarrow 1$;
while $itr < n\_itr$ **do**
  for $j \leftarrow 1$ **to** n_docs **do**
    $Q_{j,\cdot} = \left(P^\top \tilde{W}^{(j)} P + \lambda I\right)^{-1} P^\top \tilde{W}^{(j)} X_{\cdot,j}$
  for $i \leftarrow 1$ **to** n_words **do**
    $P_{i,\cdot} = \left(Q^\top \tilde{W}^{(i)} Q + \lambda I\right)^{-1} Q^\top \tilde{W}^{(i)} X_{i,\cdot}^\top$
  $c =$ mean(diag$(P^\top P)$);
  $P \leftarrow P - \alpha P(P^\top P - cI)$;
  $itr \leftarrow itr + 1$;

in the WTMF model. Meanwhile, columns of P are projection vectors, denoted by $P_{\cdot,k}$, which are similar to eigenvectors U obtained by LSA. The projection vector $P_{\cdot,k}$ is employed to multiply to a zero centered data vector $\bar{\boldsymbol{x}}$ to generate a binary string: sgn$({P_{\cdot,k}}^\top \bar{\boldsymbol{x}})$. In this section, we focus on the property of the P matrix columns.

As in equation 3, each row in matrices P and Q is iteratively optimized to approximate the data: ${P_{i,\cdot}}^\top Q_{j,\cdot} \approx X_{ij}$. While it does a good job at preserving the existence/relevance of each word in a short text, it might encode repetitive information by means of the dimensionality reduction or the projection vectors $P_{\cdot,k}$ (the columns of P). For example, the first dimension $P_{\cdot,1}$ may be 90% about the $politics$ topic and 10% about the $economics$ topic, and the second dimension $P_{\cdot,2}$ is 95% on $economics$ and 5% on $technology$ topics, respectively.

Ideally we would like the dimensions to be uncorrelated, so that more distinct topics of data could be captured. One way to ensure the uncorrelatedness is to force P to be orthogonal, i.e., $P^\top P = I$. It implies ${P_{\cdot,j}}^\top P_{\cdot,k} = 0$ if $k \neq j$.

4.1 Implementation of Orthogonal Projections

To produce nearly orthogonal projections in the current framework, we could add a regularizer $\beta(P^\top P - I)^2$ with the weight β in the objective function of the WTMF model (equation 6). However, in practice this method does not lead to the convergence of P. This is mainly caused by the phenomenon that any word profile $P_{i,\cdot}$ becomes dependent of all other word profiles after an iteration.

Therefore, we adopt a simpler method, gradient descent, in which P is updated by taking a small step in the direction of the negative gradient of $(P^\top P - I)^2$. It is also worth noting that $(P^\top P - I)^2$ requires each projection $P_{\cdot,k}$ to be a unit vector because of ${P_{\cdot,k}}^\top P_{\cdot,k} = 1$, which is infeasible when the nonzero values in X are large. Therefore, we multiply the matrix I by a coefficient c, which is calculated from the mean of the diagonal of $P^\top P$ in the current iteration. The following two lines are added at the end of an iteration:

$$\begin{aligned} c &\leftarrow \text{mean}(\text{diag}(P^\top P)), \\ P &\leftarrow P - \alpha P(P^\top P - cI). \end{aligned} \tag{5}$$

This procedure is presented in Algorithm 1. Accordingly, the magnitude of P is not affected. The step size α is fixed to 0.0001. We refer to this model as Orthogonal Matrix Factorization (OrMF).

5 Exploiting Nearest Neighbors for Tweets

We observe that tweets triggered by the same event do not have very high cosine similarity scores among them. This is caused by the inherent short length of tweets such that usually a tweet only describes one

aspect of an event (Agarwal et al., 2012; Guo et al., 2013). Our objective is to find the relevant tweets given a tweet, and then learn a model that assigns similar binary bits to these relevant tweets.

5.1 Modeling Neighboring Tweets

Given a tweet, we treat its nearest neighbors in a temporal window as its most relevant tweets. We assume that the other aspects of an event can be found in its nearest neighbors. Accordingly, we extract t neighbors for a tweet from 10,000 most chronologically close tweets. In this current implementation, we set $t = 5$.

Under the weighted matrix factorization framework, we extend each tweet by its t nearest neighbors. Specifically, for each tweet, we incorporate additional words from its neighboring tweets. The values of the new words are averaged. Moreover, these new words are treated differently by assigning a new weight w_n to them, since we believe that the new words are not as informative as the original words in the tweet.

We present an illustrative example of how to use neighbors to extend the tweets. Let $\boldsymbol{x}_1$ be a tweet with the following words (the numbers after the colon are TF-IDF values):

$\boldsymbol{x}_1$ = {obama:5.5, medicare:8.3, website:3.8}

which has two nearest neighbors:

$\boldsymbol{x}_{27}$ = {obama:5.5, medicare:8.3, website:3.8, down:5.4}

$\boldsymbol{x}_{356}$ = {obama:5.5, medicare:8.3, website:3.8, problem:7.0}

Then there are two additional words added in $\boldsymbol{x}_1$ whose values are averaged. The new data vector $\boldsymbol{x}'_1$ is:

$\boldsymbol{x}'_1$ = {obama:5.5, medicare:8.3, website:3.8, down:2.7, problem:3.5}

Therefore, the algorithm is run on the new neighbor-augmented data matrix, denoted by X', and the weight matrix W becomes

$$W_{i,j} = \begin{cases} 1, & \text{if } X'_{ij} \neq 0 \ \&j \text{ is an original word,} \\ w_n, & \text{if } X'_{ij} \neq 0, \ \&j \text{ is from neighbor tweets,} \\ w_m, & \text{if } X'_{ij} = 0. \end{cases} \tag{6}$$

This model is referred to as Orthogonal Matrix Factorization with Neighbors (OrMFN).

5.2 Binary coding without Neighbors

It is important to point out that the data used by OrMFN, X', could be a very small subset of the whole dataset. Therefore we only need to find neighbors for a small portion of the data. After the P matrix is learned, the neighborhood information is implicitly encoded in the matrix P, and we still apply the same binarization function $\text{sgn}({P_{\cdot,k}}^\top \bar{\boldsymbol{x}})$ on the whole dataset (in large scale) **without** neighborhood information. We randomly sample 200,000 tweets for OrMFN to learn P; neighbors are extracted only for these 200,000 tweets (note that the neighbors are from the 200,000 tweets as well), and then we use the learned P to generate binary codes for the whole dataset 1.35 million tweets **without** searching for their nearest neighbors.[3]

Our scheme has a clear advantage: the binary coding remains very efficient. During binarization for any data, there is no need to compare 10,000 most recent tweets to find nearest neighbors, which could be time-consuming. An opposite example is the method presented in (Guo et al., 2013), where t most nearest neighbor tweets were extracted, and a tweet profile $Q_{j,\cdot}$ was explicitly forced to be similar to its neighbors' profiles. However, for each new data, the approach proposed in (Guo et al., 2013) requires computing its nearest neighbors.

6 Experiments

6.1 Tweet Data

We crawled English tweets spanning three months from October 5th 2013 to January 5th 2014 using the Twitter API.[4] We cleaned the data such that each hashtag appears at least 100 times in the corpus, and

[3] When generating the binary codes for the 200,000 tweets, these tweets are not augmented with neighbor words.

[4] https://dev.twitter.com

each word appears at least 10 times. This data collection consists of 1,350,159 tweets, 15 million word tokens, 30,608 unique words, and 3,214 unique hashtags.

One of main reasons to use hashtags is to enhance accessing topically similar tweets (Efron, 2010). In a large-scale data setting, it is impossible to manually identify relevant tweets. Therefore, we use Twitter hashtags to create groundtruth labels, which means that tweets marked by the same hashtag as the query tweet are considered relevant. Accordingly, in our experiments all hashtags are removed from the original data corpus. We chose a subset of hashtags from the most frequent hashtags to create groundtruth labels: we manually removed some tags from the subset that are not topic-related (e.g., *#truth*, *#lol*) or are ambiguous; we also removed all the tags that are referring to TV series (the relevant tweets can be trivially obtained by named entity matching). The resulting subset contains 18 hashtags.[5]

100 tweets are randomly selected as queries (test data) for each of the 18 hashtags. The median number of relevant tweets per query is 5,621. The small size of gold standard makes the task relatively challenging. We need to identify 5,621 (0.42% of the whole dataset) tweets out of 1.35 million tweets.

200,000 tweets are randomly selected (not including the 1,800 queries) as training data for the data dependent models to learn binarization functions.[6] The functions are subsequently applied on all the 1.35 million tweets, including the 1,800 query tweets.

6.2 Evaluation

We evaluate a model by the search quality: given a tweet as query, we would like to rank the relevant tweets as high as possible. Following previous work (Weiss et al., 2008; Liu et al., 2011), we use mean precision among top 1000 returned list (MP@1000) to measure the ranking quality. Let pre@k be the precision among top k return data, then MP@1000 is the average value of pre@1, pre@2...pre@1000. Obviously MP gives more reward on the systems that can rank relevant data in the top places, e.g., if the highest ranked tweet is a relevant tweet, then all the precision values (pre@2, pre@3, pre@4...) are increased. We also calculate the precision and recall curve at varying values of top k returned list.

6.3 Methods

We evaluate the proposed unsupervised binary coding models OrMF and OrMFN, whose performance is compared against 5 other unsupervised methods, LSH, SH, LSA, ITQ, and WTMF. All the binary coding functions except LSH are learned on the 200,000 tweet set. All the methods have the same form of binary coding functions: $\text{sgn}({P_{\cdot,k}}^\top \bar{\boldsymbol{x}})$, where they differ only in the projection vector $P_{\cdot,k}$. The retrieved tweets are ranked according to their Hamming distance to the query, where Hamming distance is the number of different bit positions between the binary codes of a tweet and the query.

For ITQ and SH, we use the code provided by the authors. Note that the dense matrix $\bar{X}\bar{X}^\top$ is impossible to compute due the large vocabulary, therefore we replace it by sparse matrix $XX^\top$. For the three matrix factorization based methods (WTMF, OrMF, OrMFN) we run 10 iterations. The regularizer λ in equation 6 is fixed at 20 as in (Guo and Diab, 2012). A small set of 500 tweets is selected from the training set as tuning set to choose the missing word weight w_m in the baseline WTMF, and then its value is fixed for OrMF and OrMFN. The same 500 tweets tuning set is used to choose the neighbor word weight w_n. In fact these models are very stable, consistently outperforming the baselines regardless of different values of w_m and w_n, as later shown in Figure 4 and 5.

We also present the results of cosine similarity on the original word space (COSINE) as an upper bound of the binary coding methods. We implemented an efficient algorithm for COSINE, which is the algorithm 1 in (Petrovic et al., 2010). It firstly normalizes each data to a unit vector, then cosine similarity is calculated by traversing only once the tweets via inverted word index.

6.4 Results

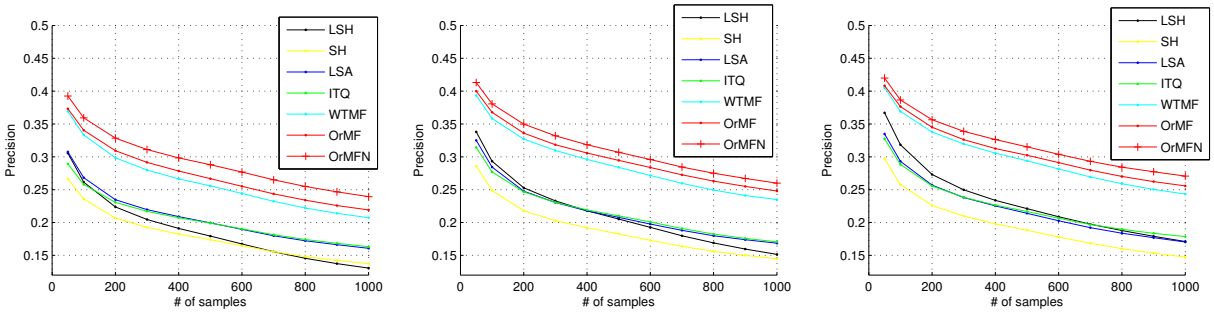
Table 2 summarizes the ranking performance measured by MP@1000 (the mean precision at top 1000 returned list). Figures 2 and 3 illustrate the corresponding precision and recall curve for the Hamming

[5]The tweet dataset and their associated list of hashtags will be available upon request.

[6]Although we use the word "training", the hashtags are never seen by the models. The training data is used for the models to learn the word co-occurrence, and construct binary coding functions.

Models	Parameters	r=64	r=96	r=128
LSH	–	19.21%	21.84%	23.75%
SH	–	18.29%	19.32%	19.95%
LSA	–	21.04%	22.07%	22.67%
ITQ	–	20.8%	22.06%	22.86%
WTMF	$w_m = 0.1$	26.64%	29.39%	30.38%
OrMF	$w_m = 0.1$	27.7%	30.48%	31.26%
OrMFN	$w_m = 0.1, w_n = 0.5$	**29.73%**	**31.73%**	**32.55%**
COSINE	–	33.68%		

Table 2: Mean precision among top 1000 returned list



(a) $r = 64$ (b) $r = 96$ (c) $r = 128$

Figure 2: Hamming ranking: precision curve under top 1000 returned list

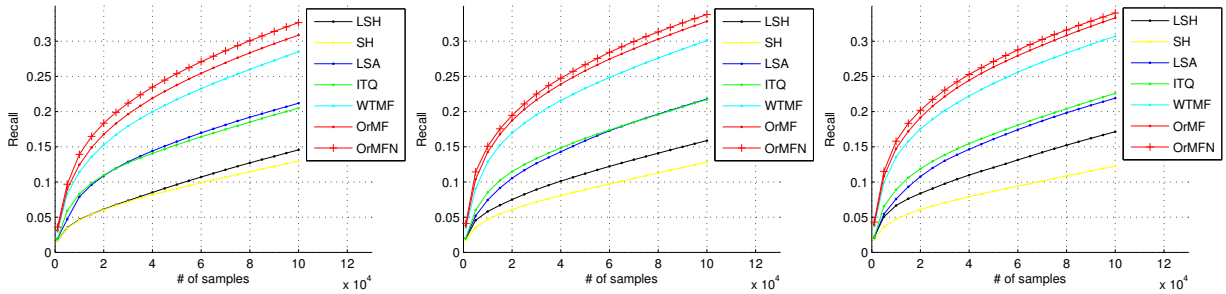
distance ranking. The number of r binary coding functions corresponds to the number of dimensions in the 6 data-dependent models LSA, SH, ITQ, WTMF, OrMF and OrMFN. The missing words weight w_m is fixed as 0.1 based on the tuning set in the three weighted matrix factorization based models WTMF, OrMF and OrMFN. The neighbor word weight w_n is chosen as 0.5 for OrMFN. Later in Section 6.4.1 we show that the performance is robust using varying values of w_m and w_n.

As the number of bits increases, all binary coding models yield better results. This is understandable since the binary bits really record very tiny bits of information from each tweet, and more bits, the more they are able to capture more semantic information.

SH has the worst MP@1000 performance. The reason might be it is designed for vision data where the data vector is relatively dense. ITQ yields comparable results to LSA in terms of MP@1000, yet the recall curve in Figure 3b,c clearly shows the superiority of ITQ over LSA.

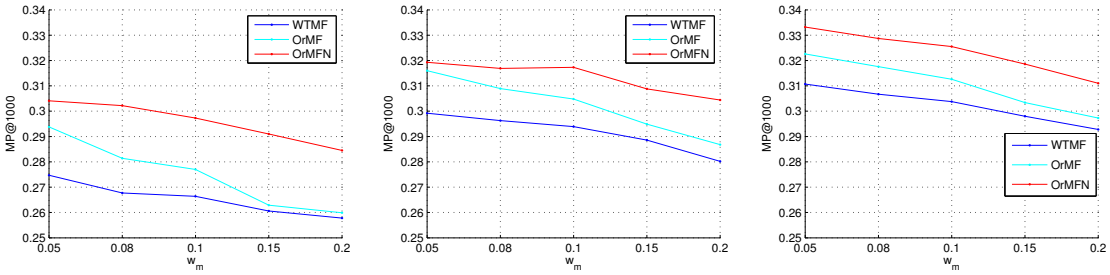
WTMF outperforms LSA by a large margin (around 5% to 7%) through properly modeling missing words, which is also observed in (Guo and Diab, 2012). Although WTMF already reaches a very high MP@1000 performance level, OrMF can still achieve around 1% improvement over WTMF, which can be attributed to orthogonal projections that captures more distinct topics. At last, leveraging neighborhood information, OrMFN is the best performing model (around 1% improvement over OrMF). The trend holds consistently across all conditions. The precision and recall curves in Figures 2 and 3 confirm the trend observed in Table 2 as well.

All the binary coding models yield worse performance than COSINE baseline. This is expected, as the binary bits are employed to gain efficiency at the cost of accuracy: the 128 bits significantly compress the data losing a lot of nuanced information, whereas in the high dimensional word space 128 bits can be only used to record two words (32 bits for two word indices and 32 bits for two TF-IDF values). We manually examined the ranking list. We found in the binary coding models, there exist a lot of ties (128 bits only result in 128 possible Hamming distance values), whereas the COSINE baseline can correctly rank them by detecting the subtle difference signaled by the real-valued TF-IDF values.



(a) $r = 64$ (b) $r = 96$ (c) $r = 128$

Figure 3: Hamming ranking: recall curve under top 100,000 returned list



(a) $r = 64$ (b) $r = 96$ (c) $r = 128$

Figure 4: Weighted matrix factorization based models: MP@1000 vs. missing word weight w_m

6.4.1 Analysis

We are interested in whether other values of w_m and w_n can generate good results – in other words, whether the performance is robust to the two parameter values. Accordingly, we present their impact on MP@1000 in Figure 4 and 5. In Figure 4, the missing word weight w_m is chosen from $\{0.05, 0.08, 0.1, 0.15, 0.2\}$, where in OrMFN the neighbor weight w_n is fixed as 0.5. The figure indicates we can achieve even better MP@1000 around 33.2% when selecting the optimal $w_m = 0.05$. In general, the curves for all the code length are very smooth; the chosen value of w_m does not have a negative impact, e.g., the gain from OrMF over WTMF is always positive.

Figure 5 demonstrates the impact of varying the values of neighbor word weight w_n from $\{0, 0.25, 0.5, 0.75, 1\}$ on OrMFN tested in different r conditions. Note that when $w_n = 0$ indicating that no neighbor information is exploited, the OrMFN model is simply reduced to the OrMF model. Based on the Figure illustration we can conclude that integrating neighboring word information always yields a positive effect, since any value of $w_n > 0$ yields a performance gain over $w_n = 0$ which is OrMF.

6.5 Computation Cost

The data-dependent models involve 2 steps: 1) learning coding functions from a small dataset, and 2) binary coding for the large scale whole dataset.[7] In real-time scenarios, the time is only spent on the 2nd step that involves no matrix factorization. The computation cost of binary coding for all models (LSH, ITQ, LSA, WTMF, OrMF and OrMFN) are roughly the same: $\text{sgn}({P_{\cdot,k}}^\top \bar{\boldsymbol{x}})$. Note that ${P_{\cdot,k}}^\top \bar{\boldsymbol{x}} = {P_{\cdot,k}}^\top \boldsymbol{x} - {P_{\cdot,k}}^\top \mu$ where $\boldsymbol{x}$ is a very sparse vector (with 11 non-zeros values on average) and ${P_{\cdot,k}}^\top \mu$ can be precomputed. On the other hand, calculating Hamming distance on binary codes is also very fast using the logic operations.

[7] Learning the binarization functions can be always done on a small dataset, for example in this paper all the data dependent models are run on the 200,000 tweets, hence it performs very fast. In addition, in the OrMFN model, there is no need to find nearest neighbors for the whole dataset in the 2nd step (the binary coding step).

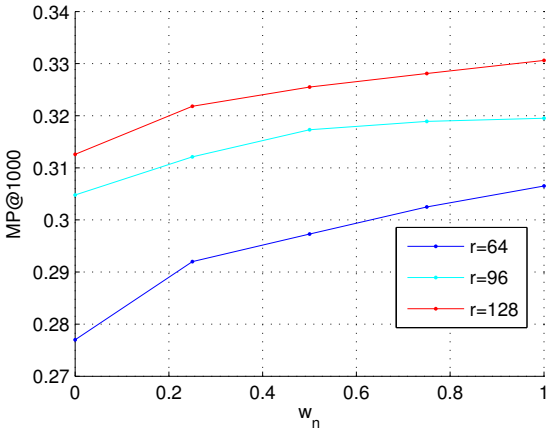


Figure 5: OrMFN model: MP@1000 vs. neighbor word weight w_n

7 Conclusion

In this paper, we proposed a novel unsupervised binary coding model which provides efficient similarity search in massive tweet data. The proposed model, OrMFN, improves an existing matrix factorization model through learning nearly orthogonal projection directions and leveraging the neighborhood information hidden in tweet data. We collected a dataset whose groundtruth labels are created from Twitter hashtags. Our experiments conducted on this dataset showed significant performance gains of OrMFN over the competing methods.

Acknowledgements

We thank Boyi Xie and three anonymous reviewers for their valuable comments. This project is supported by the DARPA DEFT Program.

References

Puneet Agarwal, Rajgopal Vaithiyanathan, Saurabh Sharma, and Gautam Shroff. 2012. Catching the long-tail: Extracting local news events from twitter. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *First Joint Conference on Lexical and Computational Semantics (*SEM)*.

Edward Benson, Aria Haghighi, and Regina Barzilay. 2011. Event discovery in social media feeds. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Rahul Bhagat and Deepak Ravichandran. 2008. Large scale acquisition of paraphrases for learning surface patterns. In *Proceedings of ACL-08: HLT*.

Andrei Z Broder, Moses Charikar, Alan M Frieze, and Michael Mitzenmacher. 1998. Min-wise independent permutations. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*.

Deepayan Chakrabarti and Kunal Punera. 2011. Event summarization using tweets. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*.

Moses S. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*.

Miles Efron. 2010. Information search and retrieval in microblogs. In *Journal of the American Society for Information Science and Technology*.

Yunchao Gong and Svetlana Lazebnik. 2011. Iterative quantization: A procrustean approach to learning binary codes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Weiwei Guo and Mona Diab. 2012. Modeling sentences in the latent space. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*.

Weiwei Guo, Hao Li, Heng Ji, and Mona Diab. 2013. Linking tweets to news: A framework to enrich online short text data in social media. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*.

Brian Kulis and Kristen Grauman. 2012. Kernelized locality-sensitive hashing. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 34(6):1092–1104.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. In *Psychological review*.

Yuhua Li, David McLean, Zuhair A. Bandar, James D. O'Shea, and Keeley Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transaction on Knowledge and Data Engineering*, 18.

Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. 2011. Hashing with graphs. In *Proceedings of the 28th International Conference on Machine Learning*.

Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. 2012a. Supervised hashing with kernels. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Wei Liu, Jun Wang, Yadong Mu, Sanjiv Kumar, and Shih-Fu Chang. 2012b. Compact hyperplane hashing with bilinear functions. In *Proceedings of the 29th International Conference on Machine Learning*.

Mohammad Norouzi and David J. Fleet. 2011. Minimal loss hashing for compact binary codes. In *Proceedings of the 28th International Conference on Machine Learning*.

Sasa Petrovic, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Nathan Srebro and Tommi Jaakkola. 2003. Weighted low-rank approximations. In *Proceedings of the Twentieth International Conference on Machine Learning*.

Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Julien Subercaze, Christophe Gravier, and Frederique Laforest. 2013. Towards an expressive and scalable twitter's users profiles. In *IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies*.

Yair Weiss, Antonio Torralba, and Rob Fergus. 2008. Spectral hashing. In *Advances in Neural Information Processing Systems*.

Learning Sense-specific Word Embeddings By Exploiting Bilingual Resources

Jiang Guo[†], **Wanxiang Che**[†], **Haifeng Wang**[‡], **Ting Liu**[†*]
[†]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
[‡]Baidu Inc., Beijing, China
{jguo, car, tliu}@ir.hit.edu.cn
wanghaifeng@baidu.com

Abstract

Recent work has shown success in learning word embeddings with neural network language models (NNLM). However, the majority of previous NNLMs represent each word with a single embedding, which fails to capture polysemy. In this paper, we address this problem by representing words with multiple and sense-specific embeddings, which are learned from bilingual parallel data. We evaluate our embeddings using the word similarity measurement and show that our approach is significantly better in capturing the sense-level word similarities. We further feed our embeddings as features in Chinese named entity recognition and obtain noticeable improvements against single embeddings.

1 Introduction

Word embeddings are conventionally defined as compact, real-valued, and low-dimensional vector representations for words. Each dimension of word embedding represents a latent feature of the word, hopefully capturing useful syntactic and semantic characteristics. Word embeddings can be used straightforwardly for computing word similarities, which benefits many practical applications (Socher et al., 2011; Mikolov et al., 2013a). They are also shown to be effective as input to NLP systems (Collobert et al., 2011) or as features in various NLP tasks (Turian et al., 2010; Yu et al., 2013).

In recent years, neural network language models (NNLMs) have become popular architectures for learning word embeddings (Bengio et al., 2003; Mnih and Hinton, 2008; Mikolov et al., 2013b). Most of the previous NNLMs represent each word with a single embedding, which ignores polysemy. In an attempt to better capture the multiple senses or usages of a word, several multi-prototype models have been proposed (Reisinger and Mooney, 2010; Huang et al., 2012). These multi-prototype models simply induce K prototypes (embeddings) for every word in the vocabulary, where K is predefined as a fixed value. These models still may not capture the real senses of words, because different words may have different number of senses.

We present a novel and simple method of learning sense-specific word embeddings by using bilingual parallel data. In this method, word sense induction (WSI) is performed prior to the training of NNLMs. We exploit bilingual parallel data for WSI, which is motivated by the intuition that the same word in the source language with different senses is supposed to have different translations in the foreign language.[1] For instance, 制服 can be translated as *investment* / *overpower* / *subdue* / *subjugate* / *uniform*, etc. Among all of these translations, *subdue* / *overpower* / *subjugate* express the same sense of 制服, whereas *uniform* / *investment* express a different sense. Therefore, we could effectively obtain the senses of one word by clustering its translation words, exhibiting different senses in different clusters.

The created clusters are then projected back into the words in the source language texts, forming a sense-labeled training data. The sense-labeled data are then trained with recurrent neural network langugae model (RNNLM) (Mikolov, 2012), a kind of NNLM, to obtain sense-specific word embeddings. As a concrete example, Figure 1 illustrates the process of learning sense-specific embeddings.

*Email correspondence.

[1] In this paper, *source language* refers to Chinese, whereas *foreign language* refers to English.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 497–507, Dublin, Ireland, August 23-29 2014.

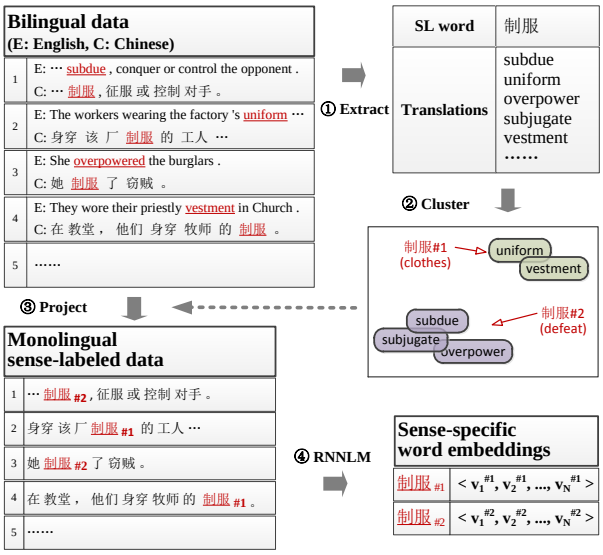


Figure 1: An illustration of the proposed method. SL stands for *source language*.

To evaluate the sense-specific word embeddings we have learned, we manually construct a Chinese polysemous word similarity dataset that contains 401 pairs of words with human-judged similarities. The performance of our method on this dataset shows that sense-specific embeddings are significantly better in capturing the sense-level similarities for polysemous words.

We also evaluate our embeddings by feeding them as features to the task of Chinese named entity recognition (NER), which is a simple semi-supervised learning mechanism (Turian et al., 2010). In order to use sense-specific embeddings as features, we should discriminate the word senses for the NER data first. Therefore, we further develop a novel monolingual word sense disambiguation (WSD) algorithm based on the RNNLM we have already trained previously. NER results show that sense-specific embeddings provide noticeable improvements over traditional single embeddings.

Our contribution in this paper is twofold:

- We propose a novel approach of learning sense-specific word embeddings by utilizing bilingual parallel data (Section 3). Evaluation on a manually constructed polysemous word similarity dataset shows that our approach better captures word similarities (Section 5.2).
- To use the sense-specific embeddings in practical applications, we develop a novel WSD algorithm for monolingual data based on RNNLM (Section 4). Using the algorithm, we feed the sense-specific embeddings as additional features to NER and achieve significant improvement (Section 5.3).

2 Background: Word Embedding and RNNLM

There has been a line of research on learning word embeddings via NNLMs (Bengio et al., 2003; Mnih and Hinton, 2008; Mikolov et al., 2013b). NNLMs are language models that exploit neural networks to make probabilistic predictions of the next word given preceding words. By training NNLMs, we obtain both high performance language models and word embeddings.

Following Mikolov et al. (2013b), we use the recurrent neural network as the basic framework for training NNLMs. RNNLM has achieved the state-of-the-art performance in language modeling (Mikolov, 2012) and learned effective word embeddings for several tasks (Mikolov et al., 2013b). The architecture of RNNLM is shown in Figure 2.

The input layer of RNNLM consists of two components: $\mathbf{w}(t)$ and $\mathbf{h}(t-1)$. $\mathbf{w}(t)$ is the *one-hot* representation of the word at time step t,[2] $\mathbf{h}(t-1)$ is the output of hidden layer at the last time step. Therefore, the input encodes all previous history when predicting the next word at time step t. Compared

[2] A feature vector of the same size of the vocabulary, and only one dimension is on.

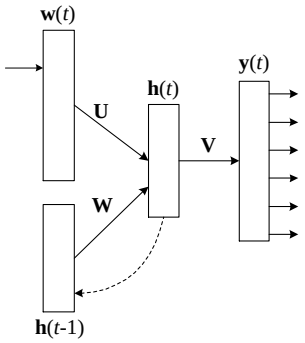


Figure 2: The basic architecture of RNNLM.

with other feed-forward NNLMs, the RNNLM can theoretically represent longer context patterns. The output $\mathbf{y}(t)$ represents the probability distribution of the next word $p(\mathbf{w}(t+1)|\mathbf{w}(t), \mathbf{h}(t-1))$. The output values are computed as follows:

$$\mathbf{h}(t) = f(\mathbf{U}\mathbf{w}(t) + \mathbf{W}\mathbf{h}(t-1)) \quad (1)$$

$$\mathbf{y}(t) = g(\mathbf{V}\mathbf{h}(t)) \quad (2)$$

where f is a sigmoid function and g is a softmax function.

The RNNLM is trained by maximizing the log-likelihood of the training data using stochastic gradient descent (SGD), in which back propagation through time (BPTT) is used to efficiently compute the gradients. In the RNNLM, $\mathbf{U}$ is the embedding matrix, where each column vector represents a word.

As discussed in Section 1, the RNNLM and even most NNLMs ignore the polysemy phenomenon in natural languages and induce a single embedding for each word. We address this issue and introduce an effective approach for capturing polysemy in the next section.

3 Sense-specific Word Embedding Learning

In our approach, WSI is performed prior to the training of word embeddings. Inspired by Gale et al. (1992) and Chan and Ng (2005), who used bilingual data for automatically generating training examples of WSD, we present a bilingual approach for unsupervised WSI, as shown in Figure 1. First, we extract the translations of the source language words from bilingual data (①). Since there may be multiple translations for the same sense of a source language word, it is straightforward to cluster the translation words, exhibiting different senses in different clusters (②).

Once word senses are effectively induced for each word, we are able to form the sense-labeled training data of RNNLMs by tagging each word occurrence in the source language text with its associated sense cluster (③). Finally, the sense-tagged corpus is used to train the sense-specific word embeddings in a standard manner (④).

3.1 Translation Words Extraction

Given bilingual data after word alignment, we present a way of extracting translation words for source language words by exploiting the translation probability produced by word alignment models (Brown et al., 1993; Och and Ney, 2003; Liang et al., 2006).

More formally, we notate the Chinese sentence as $c = (c_1, ..., c_I)$ and English sentence as $e = (e_1, ..., e_J)$. The alignment models can be generally factored as:

$$p(c|e) = \sum_a p(a, c|e) \quad (3)$$

$$p(a, c|e) = \prod_{j=1}^{J} p_d(a_j|a_{j-}, j) p_t(c_j|e_{a_j}) \quad (4)$$

where a is the alignment specifying the position of an English word aligned to each Chinese word, $p_d(a_j|a_{j-}, j)$ is the distortion probability, and $p_t(c_j|e_{a_j})$ is the translation probability which we use.

SL Word	Translation Words	Translation Word Clusters	Nearest Neighbours
制服	investment, overpower, subdue, subjugate, uniform	investment, **uniform**	穿着$_{dress}$, 警服$_{policeman\ uniform}$
		subdue, subjugate, overpower	打败$_{defeat}$, 击败$_{beat}$, 征服$_{conquer}$
花	blossom, cost, flower, spend, take, took	**flower**, blossom	菜$_{greens}$, 叶$_{leaf}$, 果实$_{fruit}$
		take, cost, **spend**	花费$_{cost}$, 节省$_{save}$, 剩下$_{rest}$
法	act, code, France, French, law, method	**France**, French	德$_{Germany}$, 俄$_{Russia}$, 英$_{Britain}$
		law, act, code	法令$_{ordinance}$, 法案$_{bill}$, 法规$_{rule}$
		method	概念$_{concept}$, 方案$_{scheme}$, 办法$_{way}$
领导	lead, leader, leadership	**leader**, leadership	主管$_{chief}$, 上司$_{boss}$, 主席$_{chairman}$
		lead	监督$_{supervise}$, 决策$_{decision}$, 工作$_{work}$

Table 1: Results of our approach on a sample of polysemous words. The second column lists the extracted translation words of the source language word (Section 3.1). The third column lists the clustering results using affinity propagation (Section 3.2). The last column lists the nearest neighbour words computed using the learned sense-specific word embeddings (Section 5.2.2).

In this paper, we use the alignment model proposed by Liang et al. (2006). We utilize the bidirectional translation probabilities for the extraction of translations, where a foreign language word w_e is determined as a translation of source language word w_c only if both translation probabilities $p_t(w_c|w_e)$ and $p_t(w_e|w_c)$ exceed some threshold $0 < \delta < 1$.

The second column of Table 1 presents the extraction results on a sample of source language words with the corresponding translation words.

3.2 Clustering of Translation Words

For each source language word, its translation words are then clustered so as to separate different senses. At the clustering time, we first represent each translation word with a feature vector (point), so that we can measure the similarities between points. Then we perform clustering on these feature vectors, representing different senses in different clusters.

Different from Apidianaki (2008) who represents all occurrences of the translation words with their contexts in the foreign language for clustering, we adopt the embeddings of the translation words as the representations and directly perform clustering on the translation words,[3] rather than the contexts of occurrences. The embedding representation is chosen for two reasons: (1) Word embeddings encode rich lexical semantics. They can be directly used to measure word similarities. (2) Embedding representation of the translation words leads to extremely high-efficiency clustering, because the number of translation words is orders of magnitude less than their occurrences.

Moreover, since the number of senses of different source language words is varied, the commonly-used k-means algorithm becomes inappropriate for this situation. Instead, we employ affinity propagation (AP) algorithm (Frey and Dueck, 2007) for clustering. In AP, each cluster is represented by one of the samples of it, which we call an *exemplar*. AP finds the *exemplars* iteratively based on the concept of "message passing". AP has the major advantage that the number of the resulting clusters is dynamic, which mainly depends on the distribution of the data. Compared with other possible clustering approaches, such as hierarchical agglomerative clustering (Kartsaklis et al., 2013), AP determines the number of resulting clusters automatically without using any partition criterions.

The third column of Table 1 lists the resulting clusters of the translation words for the sampled polysemous words. We can see that the resulting clusters are meaningful: senses are well represented by clusters of translation words.

3.3 Cross-lingual Word Sense Projection

The produced clusters are then projected back into the source language to identify word senses.

[3]The publicly available word embeddings proposed by Collobert et al. (2011) are used.

For each occurrence w^o of the word w in the source language corpora, we first select the aligned word with the highest marginal edge posterior (Liang et al., 2006) as its translation. We then identify the sense of w^o by computing the similarities of its translation word with each *exemplar* of the clusters, and select the one with the maximum similarity. When w^o is aligned with *NULL*, we heuristically identify its sense as the most frequent sense of w that appears in the bilingual dataset.

After projecting the word senses into the source language, we obtain a sense-labeled corpus, which is used to train the sense-specific word embeddings with RNNLM. The training process is exactly the same as single embeddings, except that the words in our training corpus has been labeled with senses.

4 Application of Sense-specific Word Embeddings

One of the attractive characteristic of word embeddings is that they can be directly used as word features in various NLP applications, including NER, chunking, etc. Despite of the usefulness of word embeddings on these applications, previous work seldom concerns that words may have multiple senses, which cannot be effectively represented with single embeddings. In this section, we address this problem by utilizing sense-specific word embeddings.

We take the task of Chinese NER as a case study. Intuitively, word senses are important in NER. For instance, 美 is likely to be an NE of Location when it refers to *America*. However, when it expresses the sense of *beautiful*, it should not be an NE.

Using sense-specific word embedding features for NER is not as straightforward as using single embeddings. For each word in the NER data, we first need to determine the correct word sense of it, which is a typical WSD problem. Then we use the embedding which corresponds to that sense as features. Here we treat WSD as a sequence labeling problem, and solve it with a very natural algorithm based on RNNLM we have already trained (Section 3).

4.1 RNNLM-based Word Sense Disambiguation

Given the automatically induced word sense inventories and the RNNLM which has already been trained on the sense-labeled data of source language, we first develop a greedy decoding algorithm for the sequential WSD, which works deterministically. Then we improve it using beam search.

Greedy. For word $\mathbf{w}$, we denote the sense-labeled $\mathbf{w}$ as $\mathbf{w}_{s^k}$, where s^k represents the k^{th} sense of $\mathbf{w}$. In each step, a single decision is made and the sense of next word ($\mathbf{w}(t+1)$) which has the maximum RNNLM output is chosen, given the current (sense-labeled) word $\mathbf{w}(t)_{s^*}$ and the hidden layer $\mathbf{h}(t-1)$ at the last time step as input. We simply need to compute a shortlist of $\mathbf{y}(t)$ associated with $\mathbf{w}(t+1)$, that is, $\mathbf{y}(t)|_{\mathbf{w}(t+1)}$ at each step. This process is illustrated in Figure 3.

Beam search. The greedy procedure described above can be improved using a left-to-right beam search decoding for obtaining a better sequence. The beam-search decoding algorithm keeps B different sequences of decisions in the agenda, and the sequence with the best overall score is chosen as the final sense sequence.

Note that the dynamic programming decoding (e.g. viterbi) is not applicable here, because of the recurrent characteristic of RNNLM. At each step, decisions made by RNNLM depends on all previous decisions instead of the previous state only, hence markov assumption is not satisfied.

5 Experiments

5.1 Experimental Settings

The Chinese-English parallel datasets we use include *LDC03E24*, *LDC04E12* (1998), the *IWSLT* 2008 evaluation campaign dataset and the *PKU 863* parallel dataset. All corpora are sentence-aligned. After cleaning and filtering the corpus,[4] we obtain 918,681 pairs of sentences (21.7M words).

In this paper, we use *BerkeleyAligner* to produce word alignments over the parallel dataset.[5] *BerkeleyAligner* also gives translation probabilities and marginal edge posterior probabilities. We adopt the

[4]Sentences that are too long (more than 40 words) or too short (less than 10 words) are discarded.

[5]code.google.com/p/berkeleyaligner/

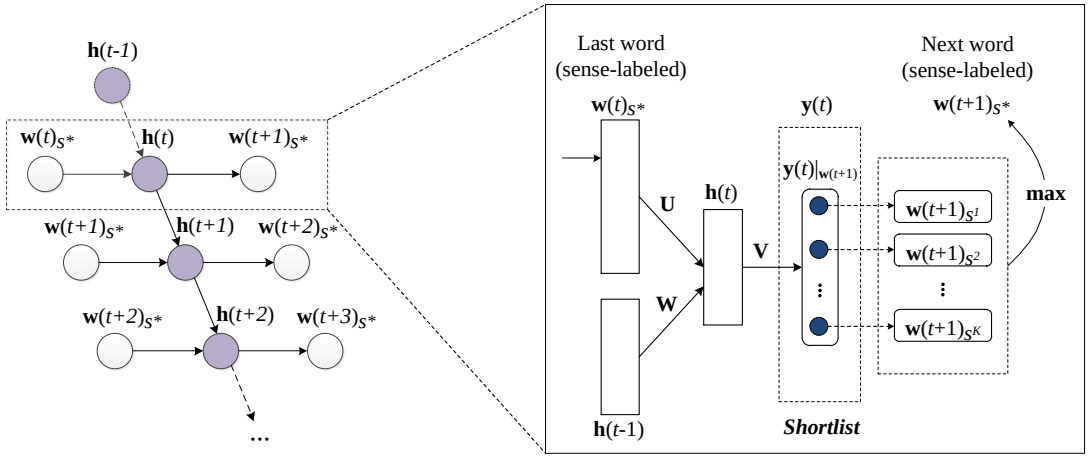


Figure 3: Using RNNLM for WSD by sequential labeling (left). Decision at each step of the RNNLM-based WSD algorithm (right).

scikit-learn tool (Pedregosa et al., 2011) to implement the AP clustering algorithm.[6] The AP algorithm is not fully automatic in deciding the cluster number. There is a tunable parameter calls *preference*. A *preference* with a larger value encourages more clusters to be produced. We set the *preference* at the median value of the input similarity matrix to obtain a moderate number of clusters. The *rnnlm* toolkit developed by Mikolov et al. (2011) is used to train RNNLM and obtain word embeddings.[7] We induce both single and sense-specific embeddings with 50 dimensions. Finally, We obtain embeddings of a vocabulary of $217K$ words, with a proportion of 8.4% having multiple sense clusters.

5.2 Evaluation on Word Similarity

Word embeddings can be directly used for computing similarities between words, which benefits many practical applications. Therefore, we first evaluate our embeddings using a similarity measurement.

Word similarities are calculated using the *MaxSim* and *AvgSim* metric (Reisinger and Mooney, 2010):

$$MaxSim(u, v) = \max_{1 \le i \le k_u, 1 \le j \le k_v} s(u^i, v^j) \quad (5)$$

$$AvgSim(u, v) = \frac{1}{k_u \times k_v} \sum_{i=1}^{k_u} \sum_{j=1}^{k_v} s(u^i, v^j) \quad (6)$$

where k_u and k_v are the number of the induced senses for words u and v, respectively. $s(\cdot, \cdot)$ can be any standard similarity measure. In this study, we use the *cosine* similarity.

Previous works used the WordSim-353 dataset (Finkelstein et al., 2002) or the Chinese version (Jin and Wu, 2012) for the evaluation of general word similarity. These datasets rarely contain polysemous words, and thus is unsuitable for our evaluation. To the best of our knowledge, no datasets for polysemous word similarity evaluation have been published yet, either in English or Chinese. In order to fill this gap in the research community, we manually construct a Chinese polysemous word similarity dataset.

5.2.1 Chinese Polysemous Word Similarity Dataset Construction

We adopt the HowNet database (Dong and Dong, 2006) in constructing the dataset. HowNet is a Chinese knowledge database that maintains comprehensive semantic definitions for each word in Chinese. The process of the dataset construction includes three steps: (1) Commonly used polysemous words are extracted according to their sense definitions in HowNet. (2) For each polysemous word, we select several other words to form word pairs with it. (3) Each word pair is manually annotated with similarity.

In step (1), we mainly took advantage of HowNet for the selection of polysemous words. However, the synsets defined in HowNet are often too fine-grained and many of them are difficult to distinguish,

[6] scikit-learn.org

[7] www.fit.vutbr.cz/~imikolov/rnnlm/

particularly for non-experts. Therefore, we manually discard those words with senses that are hard to distinguish.

In step (2), for each polysemous word w selected in step 1, we sample several other words to form word pairs with w. The sampled words can be roughly divided into two categories: *related* and *unrelated*. The *related* words are sampled manually. They can be the *hypernym*, *hyponym*, *sibling*, *(near-)synonym*, *antonym*, or *topically related* to one sense of w. The *unrelated* words are sampled randomly.

In step (3), we ask six graduate students who majored in computational linguistics to assign each word pair a similarity score. Following the setting of WordSim-353, we restrict the similarity score in the range (0.0, 10.0). To address the inconsistency of the annotations, we discard those word pairs with a standard deviation greater than 1.0. We end up with 401 word pairs annotated with acceptable consistency. Unlike the WordSim-353, in which most of the words are nouns, the words in our dataset are more diverse in terms of part-of-speech tags.

Table 2 lists a sample of word pairs with annotated similarities from the dataset. The whole evaluation dataset will be publicly available for the research community.[8]

Word	Paired word	Category	Mean.Sim	Std.Dev
制服	征服$_{conquer}$	synonym	8.60	0.29
	重点$_{key\ point}$	unrelated	0.12	0.19
出	进$_{enter}$	autonym	7.90	0.97
	发表$_{publish}$	near-synonym	7.86	0.76
花	茎$_{plant\ stem}$	sibling	7.80	0.12
	费用$_{cost}$	topic-related	5.86	0.90
面	食物$_{food}$	hypernym	6.50	0.71

Table 2: Sample word pairs of our dataset. The unrelated words are randomly sampled. $Mean.Sim$ represents the mean similarity of the annotations, $Std.Dev$ represents the standard deviation.

5.2.2 Evaluation Results

Following Zou et al. (2013), we use Spearman's ρ correlation and Kendall's τ correlation for evaluation. The results are shown in Table 3. By utilizing sense-specific embeddings, our approach significantly outperforms the single-version using either *MaxSim* or *AvgSim* measurement.

For comparison with multi-prototype methods, we borrow the context-clustering idea from Huang et al. (2012), which was first presented by Schütze (1998). The occurrences of a word are represented by the average embeddings of its context words. Following Huang et al.'s settings, we use a context window of size 10 and all occurrences of a word are clustered using the spherical k-means algorithm, where k is tuned with a development set and finally set to 2.

System	**MaxSim**		**AvgSim**	
	$\rho \times 100$	$\tau \times 100$	$\rho \times 100$	$\tau \times 100$
Ours	**55.4**	**40.9**	49.3	35.2
SingleEmb	42.8	30.6	42.8	30.6
Multi-prototype	40.7	29.1	38.3	27.4

Table 3: Spearman's ρ correlation and Kendall's τ correlation evaluated on the polysemous dataset.

Surprisingly, the *multi-prototype* method performs even slightly worse than the single-version, which suggests that learning a fixed number of embeddings for every word may even harm the embedding. Additionally, the clustering process of the multi-prototype approach suffers from high memory and time cost, especially for the high-frequency words.

[8] ir.hit.edu.cn/~jguo

To obtain intuitive insight into the superior performance of sense-specific embeddings, we list in the last column of Table 1 the nearest neighborhoods of the sampled words in the evaluation dataset. The list shows that we are able to find the different meanings of a word by using sense-specific embeddings.

5.3 Application on Chinese NER

We further apply the sense-specific embeddings as features to Chinese NER. We first perform WSD on the NER data using the algorithm introduced in Section 4. For beam search decoding, the beam size B is tuned on a development set and is finally set to 16.

We conduct our experiments on data from *People's Daily* (Jan. and Jun. 1998).[9] The original corpus contains seven NE types.[10] In this study, we select the three most common NE types: Person, Location, Organization. The data from January are chosen as the training set (37,426 sentences). The first 2,000 sentences from June are chosen as the development set and the next 8,000 sentences as the test set.

CRF models are used in our NER system and are optimized by L2-regularized SGD. We use the CRFSuite (Okazaki, 2007) because it accepts feature vectors with numerical values. The state-of-the-art features (Che et al., 2013) are used in our baseline system. For both single and sense-specific embedding features, we use a window size of 4 (two words before and two words after).

5.3.1 Results

Table 4 demonstrates the performance of NER on the test set. As desired, the single embedding features improve the performance of our baseline, which were also shown in (Turian et al., 2010). Furthermore, the sense-specific embeddings outperform the single word embeddings by nearly 1% F-score (88.56 vs. 87.58), which is statistically significant (p-value < 0.01 using one-tail t-test).

System	P	R	F
Baseline	93.27	81.46	86.97
+SingleEmb	93.55	82.32	87.58
+SenseEmb (greedy)	93.38	83.56	88.20
+SenseEmb (beam search)	**93.59**	**84.05**	**88.56**

Table 4: Performance of NER on test data.

According to our hypothesis, the sense-specific embeddings should bring considerable improvements to the NER of polysemous words. To verify this, we evaluate the per-token accuracy of the polysemous words in the NER test data. We again adopt HowNet to determine the polysemy. Words that are defined with multiple senses are selected as test set. Figure 4 shows that the sense-specific embeddings indeed improve the NE recognition of the polysemous words, whereas the single embeddings even decrease the accuracy slightly. We also obtain improvements on the NE recognition of the monosemous words, which provide evidences that more accurate prediction of polysemous words is beneficial for the prediction of the monosemous words through contextual influence.

6 Related Work

Previous studies have explored the NNLMs, which predict the next word given some history or future words as context within a neural network architecture. Schwenk and Gauvain (2002), Bengio et al. (2003), Mnih and Hinton (2007), and Collobert et al. (2011) proposed language models based on feed-forward neural networks. Mikolov et al. (2010) studied language models based on RNN, which managed to represent longer history information for word-predicting and demonstrated outstanding performance.

Besides, researchers have also explored the word embeddings learned by NNLMs. Collobert et al. (2011) used word embeddings as the input of various NLP tasks, including part-of-speech tagging, chunking, NER, and semantic role labeling. Turian et al. (2010) made a comprehensive comparison of various types of word embeddings as features for NER and chunking. In addition, word embeddings

[9] www.icl.pku.edu.cn/icl_groups/corpus/dwldform1.asp

[10] Person, Location, Organization, Date, Time, Number and Miscellany

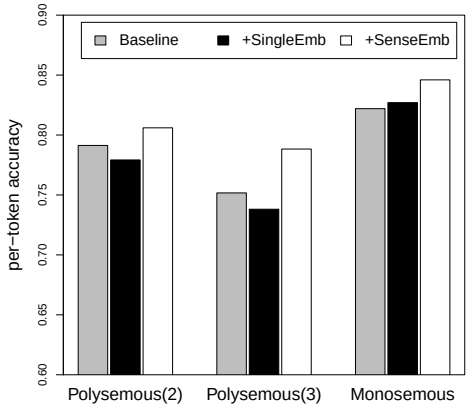


Figure 4: Per-token accuracy on the polysemous and monosemous words in the NER test data. Polysemous(k) represents the set of words that have more than or equal to k senses defined in HowNet.

are shown to capture many relational similarities, which can be recovered by vector arithmetic in the embedding space (Mikolov et al., 2013b; Fu et al., 2014). Klementiev et al. (2012) and Zou et al. (2013) learned cross-lingual word embeddings by utilizing MT word alignments in bilingual parallel data to constrain translational equivalence.

Most previous NNLMs induce single embedding for each word, ignoring the polysemous property of languages. In an attempt to capture the different senses or usage of a word, Reisinger and Mooney (2010) and Huang et al. (2012) proposed multi-prototype models for inducing multiple embeddings for each word. They did this by clustering the contexts of words. These multi-prototype models simply induced a fixed number of embeddings for every word, regardless of the real sense capacity of the specific word.

There has been a lot of work on using bilingual resources for word sense disambiguation (Gale et al., 1992; Chan and Ng, 2005). By using aligned bilingual data along with word sense inventories such as WordNet, training examples for WSD can be automatically gathered. We employ this idea for word sense induction in our study, which is free of any pre-defined word sense thesaurus.

The most similar work to our sense induction method is Apidianaki (2008). They presented a method of sense induction by clustering all occurrences of each word's translation words. In their approach, occurrences are represented with their contexts. We suggest that clustering contexts suffer from high memory and time cost, as well as data sparsity. In our method, by clustering the embeddings of translation words, we induce word senses much more efficiently.

To evaluate word similarity models, researchers often apply a dataset with human-judged similarities on word pairs, such as WordSim-353 (Finkelstein et al., 2002), MC (Miller and Charles, 1991), RG (Rubenstein and Goodenough, 1965) and Jin and Wu (2012). For context-based multi-prototype models, (Huang et al., 2012) constructs a dataset with context-dependent word similarity. To the best of our knowledge, there is no publicly available datasets for context-unaware polysemous word similarity evaluation yet. This paper fills this gap.

7 Conclusion

This paper presents a novel and effective approach of producing sense-specific word embeddings by exploiting bilingual parallel data. The proposed embeddings are expected to capture the multiple senses of polysemous words. Evaluation on a manually annotated Chinese polysemous word similarity dataset shows that the sense-specific embeddings significantly outperforms the single embeddings and the multi-prototype approach.

Another contribution of this study is the development of a beam-search decoding algorithm based on RNNLM for monolingual WSD. This algorithm bridges the proposed sense-specific embeddings and practical applications, where no bilingual information is provided. Experiments on Chinese NER show that the sense-specific embeddings indeed improve the performance, especially for the recognition of the polysemous words.

Acknowledgments

We are grateful to Dr. Zhenghua Li, Yue Zhang, Shiqi Zhao, Meishan Zhang and the anonymous reviewers for their insightful comments and suggestions. This work was supported by the National Key Basic Research Program of China via grant 2014CB340503 and 2014CB340505, the National Natural Science Foundation of China (NSFC) via grant 61370164.

References

Marianna Apidianaki. 2008. Translation-oriented word sense induction based on parallel corpora. In *In Proceedings of the 6th Conference on Language Resources and Evaluation (LREC)*, Marrakech, Morocco.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155.

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

Yee Seng Chan and Hwee Tou Ng. 2005. Scaling up word sense disambiguation via parallel texts. In *AAAI*, volume 5, pages 1037–1042.

Wanxiang Che, Mengqiu Wang, Christopher D. Manning, and Ting Liu. 2013. Named entity recognition with bilingual constraints. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 52–62, Atlanta, Georgia, June.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Zhendong Dong and Qiang Dong. 2006. *HowNet and the Computation of Meaning*. World Scientific.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science*, 315(5814):972–976.

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, Baltimore MD, USA.

William A Gale, Kenneth W Church, and David Yarowsky. 1992. Using bilingual materials to develop word sense disambiguation methods. In *Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 101–112.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882.

Peng Jin and Yunfang Wu. 2012. Semeval-2012 task 4: evaluating chinese word similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 374–377.

Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Stephen Pulman. 2013. Separating disambiguation from composition in distributional semantics. *CoNLL-2013*, pages 114–123.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words. In *Proceedings of COLING 2012*, pages 1459–1474, Mumbai, India, December.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 104–111.

Tomas Mikolov, Martin Karafiát, Lukáš Burget, Jan Cernocky, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of Interspeech*, pages 1045–1048.

Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and J Černockỳ. 2011. Rnnlm-recurrent neural network language modeling toolkit. In *Proc. of the 2011 ASRU Workshop*, pages 196–201.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, pages 746–751.

Tomas Mikolov. 2012. *Statistical Language Models Based on Neural Networks*. Ph.D. thesis, Ph. D. thesis, Brno University of Technology.

George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648.

Andriy Mnih and Geoffrey E Hinton. 2008. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088.

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

Naoaki Okazaki. 2007. Crfsuite: a fast implementation of conditional random fields (crfs). *URL http://www. chokkan. org/software/crfsuite*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123.

Holger Schwenk and Jean-Luc Gauvain. 2002. Connectionist language modeling for large vocabulary continuous speech recognition. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 1, pages 765–768.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394.

Mo Yu, Tiejun Zhao, Daxiang Dong, Hao Tian, and Dianhai Yu. 2013. Compound embedding features for semi-supervised learning. In *Proceedings of NAACL-HLT*, pages 563–568.

Will Y. Zou, Richard Socher, Daniel Cer, and Christopher D. Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398, Seattle, Washington, USA, October.

Using unmarked contexts in nominal lexical semantic classification

Lauren Romeo[*], Sara Mendes[*,†], Núria Bel[*]
*Universitat Pompeu Fabra, Roc Boronat, 138, Barcelona, Spain
†Centro de Linguística da Universidade de Lisboa, Av. Prof. Gama Pinto, 2, Lisboa, Portugal
{lauren.romeo,sara.mendes,nuria.bel}@upf.edu

Abstract

The work presented here addresses the use of unmarked contexts in pattern-based nominal lexical semantic classification. We define unmarked contexts to be the counterposition of the class-indicatory, or marked, contexts. Its aim is to evaluate how unmarked contexts can be used to improve the accuracy and reliability of lexical semantic classifiers. Results demonstrate that the combined use of both types of distributional information (marked and unmarked) is crucial to improve classification. This result was replicated using two different corpora, demonstrating the robustness of the method proposed.

1 Introduction

Lexical resources annotated with lexical semantic classes have been successfully incorporated into a wide range of NLP applications, such as grammar induction (Agirre *et al.*, 2011) and the building and extending of semantic ontologies (Abbès *et al.*, 2011). However, lexical semantic tagging in large lexica is mostly done by hand, implying high costs with regard to maintenance and domain tuning. As the use of an inadequate lexicon is one of the causes of poor performance of NLP applications, current research to improve the automatic production of rich language resources, and of class-annotated lexica, in particular, is critical.

One way to approach this task is through supervised cue-based lexical semantic classification. Based on the distributional hypothesis (Harris, 1954), according to which words occurring in the same contexts can be said to belong to the same class, cue-based lexical semantic classification uses particular linguistic contexts where nouns occur as cues that represent distinctive distributional traits of a lexical class. Yet, training a classifier with information about word occurrences in a corpus within a selected number of contexts can present a challenge, mainly because specific words might be observed in a number of class-indicative contexts but not always are.

This type of marked, or class-indicative, context (e.g. co-occurrence with specific prepositions, predicate selectional restrictions, and grammatical information, such as indirect objects) are sparse in any corpus as, being so specific, they do not occur often with each target noun. Using only exclusive class-indicative contexts as features in nominal lexical semantic classification has been shown to not always provide sufficient information to make a decision regarding class membership of a noun (Bel *et al.*, 2012), especially when the data does not contain relevant co-occurrences or when those co-occurrences are too disperse to be correlated.

Recent work on the use of distributional models for nominal classification tasks (Romeo *et al.*, 2014) discusses potential bottlenecks of models using data extracted with lexico-syntactic patterns as features, identifying data sparsity as one of the major issues affecting the performance of these systems. In fact, the selection of class-indicative information, in an attempt to provide relevant information to classifiers and thus reduce noise, naturally limits the amount of data available to the system, often resulting in sparse vectors.

Resulting from the necessity of selecting the information provided to classifiers, in an attempt to improve the accuracy of classification decisions, the sparse data problem in nominal lexical semantic classification is one of the crucial issues to be addressed to improve the performance of these systems. We propose to approach this issue by utilizing a larger fraction of the distributional information available in a corpus, by incorporating information typically considered non-indicatory of semantic class membership, which we will designate as *unmarked contexts* (see Section 3 for a definition).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 508–519, Dublin, Ireland, August 23-29 2014.

Our hypothesis is that the distributional behavior of nouns of a particular class in this type of generally occurring contexts can show certain characteristics in common that may be explored in the context of lexical semantic classification. Our goal in the work presented here is to evaluate to which extent this information, in combination with the widely explored class-indicative lexico-syntactic contexts, can be used to improve results in classification tasks, by providing more information to classifiers. To do this, we experiment with English nouns of the following lexical semantic classes: INFORMATION (INF), ORGANIZATION (ORG), LOCATION (LOC), EVENT (EVT) and HUMAN (HUM).

The work presented in this paper is structured as follows: Section 2 describes theoretical claims used in approaches to nominal lexical semantic classification, as well as related work; Section 3 elaborates upon the concept of unmarked contexts; Section 4 describes the methodology followed; Section 5 presents the results obtained; Section 6 discusses their implications; and Section 7 concludes with final remarks and future work.

2 Motivation and Related Work

In semantic classification approaches grounded in usage-based theories of grammar (Goldberg, 2006), a lexical class is seen as a generalization of the systematic co-distribution of a number of words and contexts. Construction-based grammar hypotheses allow us to predict there are sets of word occurrences that, together, constitute a class mark, indicating a particular semantic class, in line with the structuralist notion of markedness (Jakobson, 1971).

The identification of relevant cues for machine learning classification is problematic as low frequency evidence is typically disregarded by automatic systems. To overcome this problem, Bel (2010) applied smoothing methods, demonstrating increases in accuracy, though low frequency words remained problematic for classifiers when evidence was scarce, and thus not considered as a positive cue for the class, although it was indicative.

In Bel *et al*. (2012) we built on this hypothesis, assuming only frequently occurring contexts would be efficient for classification tasks, and considering only frequent predicates, prepositions, affixes, etc., as well as negative cases (i.e. marked cues for other classes), as indicators for a class membership decision. Thus, we ignored all information that, although frequent, co-occurred with nouns from all classes and was deemed not distinctive of a particular class, as well as all information that, though distinctive, was not frequent enough to be used by the classifier.

Table 1 provides examples of contexts considered in that work, which did not rely on unique, exclusive hints, but a number of them that, when correlated, could identify members of a given class. However, our results were inconclusive, a fact which we attributed to the high impact of sparse data.

Class	Examples of lexico-syntactic patterns
ORG	x-NN (found|establish|organize)-VBD
LOC	(inside|outside)-IN (the|a|an)-(DT|Z) x-NN
INF	(submit|publish|report)-V* (the|a|an)-(DT|Z) x-NN
EVT	during-IN (the|a|an)-(DT|Z) x-NN
HUM	x-(-er|–or| –man)-NN

Table 1: Examples of lexico-syntactic patterns indicative of 5 different lexico-semantic classes, which we refer to as *marked contexts*.

According to Bybee (2010), general contexts, not exclusive to a particular class (i.e. unmarked contexts, as defined in Section 3), are more frequent than contexts marked toward a particular class, as they occur with nouns of all classes. In view of this, it becomes apparent that a large part of available distributional data is not taken into consideration when these very general co-occurrences observed with nouns of all classes (e.g. co-occurrence with an article) are treated as stop words or contexts in lexical semantic classification tasks.

The basic claim leading most authors to neglect this kind of context is that, due to its assumed undifferentiated distribution, this information presents a challenge for classifiers to accurately use it in class membership decisions, which is bound to negatively affect results (see Cooke and Gillam (2008), Turney and Pantel (2010), Bullinaria and Levy (2012), among many others). In contrast with this mainstream position, Rumshisky *et al*. (2007) argued there is an asymmetry in the way certain word

senses are used in language, preferably or rarely occurring in certain very general contexts (e.g. subject position, occurrence with an adjectival modifier, etc.).

This type of asymmetry is essentially referring to a difference in how general, semantically neutral distributional contexts are more or less frequent in data, depending on the sense in which a word is used. We hypothesize that these tendencies can also be observed when considering different lexical semantic classes.

In this paper, we propose a way to include this type of asymmetry in the information provided to classifiers to verify its impact on their overall performance. Considering such distributional evidence will increase the amount of information made available to classifiers. Our main claim is that devising a strategy to informatively include this type of distributional information in classification tasks can allow us to take advantage of a bigger portion of the data available in corpora and improve the accuracy of classifiers in this way.

3 Unmarked contexts

In contrast with mainstream approaches to cue-based lexical-semantic classification, we argue for the inclusion of a type of distributional information typically not considered to be indicatory of class membership, and thus not informative to automatic classification systems. These are very general contexts of occurrence typically disregarded as semantically-empty and thought to be too general to contribute any relevant information. At the same time, they correspond to a large amount of corpus data that is a priori not considered due to the assumption that it does not provide any information.

Examples of such distributional information regard whether nouns occur preceeded by an article, in singular or plural form, whether they are a head or a complement in NPs containing the preposition *of*, if they occur as the subject of the verb to *be*, etc. We will henceforth designate these contexts as ***unmarked contexts***, the counterposition of the class-indicatory contexts, i.e. ***marked contexts***, used as class marks in cue-based lexical-semantic classification systems.

Following the conclusions of Rumshisky *et al.* (2007) regarding asymmetries in the distribution of word senses in general contexts, our hypothesis is that the distribution of members of a class with respect to their occurrence in particular unmarked contexts is consistent and thus can be captured and used to inform classifiers and improve results, when considered along with other indicatory, or marked, contexts.

We also hypothesize that unmarked contexts will alleviate problems caused by data sparsity in classification tasks by providing additional information to classifiers. To assess to which extent this information can be used in classification tasks, we had to identify such contexts and verify whether our hypothesis was confirmed, i.e. if they showed significant variations in terms of distribution that might be explored to augment the amount of information made available to classifiers[1]. Additionally, and given the specific properties of this type of distributional information, we also had to define a strategy to informatively provide it to classifiers (see Section 3.2 for details).

3.1 Identifying unmarked contexts

Considering the characteristics of the contexts discussed above, we identified 32 unmarked contexts under a frequency criterion (see Table 2 for a description of the different contexts identified), hypothesizing that more frequent contexts combine with more nouns in the corpus and thus should not be marked for any restricted set. However, although they are not considered to be class marks, we expect these contexts to be asymmetrically distributed between lexical semantic classes, in an analogous way to what was observed by Rumshisky *et al*. (2007) with regard to the distributional behavior of different word senses in language use.

We studied the distribution of these contexts in a web-crawled corpus (see Section 4), comparing the distribution of each context over all the nouns in the corpus and over nouns defined as part of a specific lexical semantic class, according to a gold standard (see Section 4.2). To do this, we calculated the mean of occurrence of nouns pertaining to a particular class in a specific unmarked context,

[1] The approach detailed in this paper contrasts with a 'bag of words' approach to classification as, even in the case of what we call unmarked contexts, we rely on cue information to populate our vectors. Thus, the information provided to classifiers takes into consideration linguistic information, such as syntactic order or dependencies. Moreover, our use of linguistically-motivated features, from the inherently distinctive to the more generic, reduces the amount of data needed to obtain a desired level of performance.

as well as the mean of occurrence of all the nouns in the corpus in that same context; we then determined if there was a statistically significant difference[2] between the behavior of nouns from specific classes and the behavior of nouns in general with regard to the contexts identified as unmarked.

Feature Type	Description	Examples
article	target noun preceded by a(n) (in)definite article	*(a|an)-(DT|Z) x-NN* or *(the)-(DT|Z) x-NN*
number	target noun in plural/singular form	*x-NNS* or *x-NN*
copula	target noun as subject/object of verb *to be*	*x-NN* be-*VBZ* or be-*VBZ x-NN*
modifiers	adjective or nominal modifier preceding target noun	*x-JJ x-NN* or *x-NN x-NN*
preposition *of*	target noun preceding/following the preposition *of*	*x-NN of-IN* or *of-IN x-NN*
subject of V	target noun as subject of each of the 20 most frequent verbs in the corpus	*x-NN (have|get|make|see|do|take|go|use|find|help|read|know|provide|give|keep|come|say|create|visit) -VB(Z|D)*

Table 2: Description of unmarked contexts identified and used in our experiments.

The results showed there were statistically significant differences ($p<0.05$) in the behavior of nouns in particular classes with regard to certain unmarked contexts. For instance, the occurrence of INF, ORG, LOC, and HUM nouns with a definite article (*the-DT*) showed to be significantly different from its average occurrence with all the nouns in the corpus. The occurrence with an indefinite article (*a|an-DT*), on the other hand, showed to be significantly different for LOC nouns, while the co-occurrence with an adjective (*x-JJ*) was significantly different for INF nouns.

3.2 A strategy to encode unmarked context information in feature vectors

The preliminary study mentioned in Section 3.1 provided evidence confirming that there are, in fact, differences in the behavior of particular lexical semantic classes with regard to their occurrence in unmarked contexts. Thus, the next step consisted in determining the best way to make this information available to classifiers.

Aiming to check the validity of our hypothesis in general, the results obtained in the aforementioned study were not used directly to narrow down the information to include in the vectors used by the classifiers to avoid the risk of over-fitting. Moreover, what was at stake, considering our theoretical hypothesis, was to devise a strategy to account for specific differences between the behavior of each noun considered for classification and the average behavior of all nouns in the corpus with regard to each context considered. Thus, information regarding all 32 unmarked contexts was provided to the classifiers for all lexical classes considered.

To mirror the specificity of the distribution of each noun with regard to each context considered, we subtracted the mean of occurrence of nouns in each context from the actual occurrences of the target noun represented by the vector in that same context to obtain each feature *f*, as defined in Equation 1, where c_i represents a given context, *t* a target noun, *n* any noun belonging to *N*, the set of all nouns in the corpus, and *freq* frequency of occurrence (e.g. $freq(t \mid c_i)$ = frequency of occurrence of the target noun *t* in context c_i).

Equation 1: $$f = \frac{freq(t \mid ci)}{freq(t)} - \frac{1}{|N|} \sum_{n \in N} \left[\frac{freq(n \mid ci)}{freq(n)} \right]$$

Using the difference between the number of occurrences of a given noun and the average occurrence of all nouns in a specific context, we encode the deviation of the behavior of that noun with re-

[2] In this work, statistical significance was calculated using Student's t-test (cf. Krenn and Samuelsson, 1997).

gard to the general behavior of all nouns in the corpus, under the hypothesis that nouns of the same class display similar tendencies in terms of deviant behavior in the contexts considered, providing relevant information to the classifier. We apply our method to two different corpora making apparent its robustness.

4 Experimental design and setup

In order to evaluate the impact of using distributional information on unmarked contexts for lexical-semantic classification tasks, first, we had to extract distributional information regarding the unmarked contexts identified (see Section 3.1), as well as distributional information regarding class-indicative marked contexts. In our experiments, we used the marked contexts identified and described in Bel *et al*. (2012) (see Table 1 for examples).

Once the distributional information was extracted, we incorporated it in feature vectors, using the different aforementioned strategies for encoding distributional information regarding marked and unmarked contexts, respectively, as detailed further below in this section. Once all of the information was compiled, the vectors were provided to classifiers.

As previously mentioned, our experiments covered English nouns of the classes: INF, ORG, HUM, EVT and LOC (see Section 4.2). For the purpose of the work presented here, we experimented with two corpora to determine the transferability and robustness of our method, independently of specific corpus data.

We first used a general web-crawled corpus (Pecina *et al.*, 2011) consisting of 30 million PoS-tagged English tokens (henceforth Corpus A) to identify unmarked contexts (see Section 3.1) as well as to train our classifiers.

We also employed an excerpt of the web-crawled UkWaC corpus (Baroni *et al.*, 2009), consisting of 60 million PoS-tagged English tokens (henceforth Corpus B) to test our approach on unknown data, in this way ensuring that our approach and classifiers are not over-fitted to any specific corpus, instead confirming that the method we propose can be generalized, and the results obtained are replicable given any dataset.

Regular expressions over both corpora were used to identify occurrences of nouns in marked and unmarked contexts. For marked contexts, the relative frequency of each pattern seen with a particular noun was stored in an *n*-dimensional vector.[3] The occurrences of a noun in unmarked contexts were encoded in the same vectors following the strategy outlined in Section 3.2 (see Equation 1).

4.1 Classification

For classification, we used the Logistic Model Trees (LMT) (Landwehr *et al.*, 2005) Decision Tree (DT) classifier in the WEKA (Witten and Frank, 2005) implementation in a 10-fold cross-validation setting. We conducted binary classifications, one for each semantic class considered. We measure the success of our approach in regards to the joint performance of individual classifiers in accurately distinguishing members of each individual class from any other noun. This method was used in the classification experiments over both corpora described above.

4.2 Gold Standard Description

In regards to the gold standard lists used for training and evaluation, we automatically extracted from WordNet (Miller *et al.*, 1990) all of the nouns encoded in this repository of lexical information that contained a sense corresponding to a class considered in our experiments (e.g. *people* in the case of HUM).

The gold standards were not contrasted with the actual occurrences of the nouns in the corpora. They were, however, balanced with respect to class members and elements not belonging to the class, resulting in the dataset described in Table 3. Each noun appears *x* times in any corpus considered. The elements not belonging to a class were randomly selected from the set of nouns that do not contain a sense in WordNet that corresponded to the target class being classified.

For a fair comparison, the baseline classification model was obtained using the context patterns described in Bel *et al*. (2012) with the LMT classifier, using the previously described gold standard lists over Corpus A. This baseline allows us to assess the impact of unmarked contexts in nominal lexical

[3] In this work, *n* is equal to the amount of marked contexts plus unmarked contexts considered for each class.

semantic classification, since the classifiers proposed here that are provided with information on the distributional behavior of nouns in unmarked contexts also use Bel *et al.* (2012)'s context patterns to extract class-indicative, or marked, distributional information regarding the nouns to classify.

Class	**ORG**	**LOC**	**EVT**	**INF**	**HUM**
Class members	138	157	260	262	246
Elements not belonging to the class	135	156	260	259	246

Table 3: Number of nouns included in gold standards per class.

5 Results

Tables 4 and 5 show results obtained in our experiments in terms of Precision (P), Recall (R) and F-Measure (F). The overall accuracy of all classifiers for each experiment is also provided. The baseline classifiers achieve an average accuracy of 70.84%. By including unmarked contexts in the vectors provided to the classifiers, the average accuracy of the classifiers rises to 75.16%, representing an error reduction of 4.32 points. We tested the statistical significance ($p<0.1$) of this increase in the accuracy of classification and, for all classes except for HUM, the increase in accuracy between the baseline results and those obtained when including unmarked contexts is significant. These results are discussed in detail in Section 6.

Knowing a potential downside of using unmarked contexts in classification tasks is an increase in noise (see Section 6.1 for a detailed discussion regarding this concept), we conducted an error analysis of the results obtained, which made apparent that most of the noise was due to imprecise information extracted with our regular expressions, leading us to revise them. As these revisions resulted from the observation that a portion of the errors in the baseline results was due to imprecise regular expressions, they did not consist in the definition of new marked contexts, rather in a revision of how to extract marked contexts already considered in this work from corpora data. Thus, these revisions resulted in more accurate and better defined regular expressions.

As indicated by the results, these revisions in combination with the unmarked contexts further raised the average accuracy of the classifiers to 76.35% (see Table 4), representing an error reduction of 5.51 points with regard to the baseline. Having obtained these promising results over the data in the corpus used to develop our approach (Corpus A), it was crucial to verify the replicability of our method using a different and completely independent corpus, as described in Section 4. Moreover, replicating the original experiments over a different corpus was also crucial to assure that the revisions made to the regular expressions did not result in any over-fitting between the extraction of distributional information and the corpus being used. The results obtained for the experiments conducted over Corpus B are presented in Table 5.

Class	**baseline**			**baseline + unmarked contexts**			**marked contexts**			**marked + unmarked contexts**		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
ORG	0.64	0.62	0.60	0.70	0.68	0.68	0.76	0.74	0.74	0.75	0.74	0.74
LOC	0.72	0.70	0.70	0.73	0.73	0.73	0.70	0.70	0.70	0.77	0.79	0.77
EVT	0.70	0.68	0.67	0.74	0.73	0.72	0.73	0.72	0.64	0.73	0.72	0.69
INF	0.67	0.66	0.65	0.74	0.73	0.73	0.71	0.70	0.69	0.71	0.71	0.71
HUM	0.86	0.84	0.86	0.87	0.86	0.86	0.87	0.87	0.87	0.85	0.84	0.84
Acc	70.84%			75.16%			75.05%			76.35%		

Table 4: Precision (P), Recall (R), and F-Measure (F) of classifiers over Corpus A.

The classifiers that include unmarked contexts yielded an average accuracy of 76.03% over Corpus B, representing an error reduction of 3.34 points with regard to the classifier including only marked contexts (using the revised version of Bel *et al.* (2012)'s cues), which is a statistically significant improvement ($p<0.05$). Moreover, these results represent an improvement of accuracy by 5.19 points with regard to the baseline. This demonstrates, on the one hand, that the definition of relevant contexts based on Corpus A data did not result in an over-fitted approach; and, on the other hand, that the

method presented here is robust, as we used our classifiers over a completely different corpus (cf. Section 3) and still yielded comparable results. Due to space limitations, below we detail only the results obtained on Corpus B data, as these are independent of all the preliminary studies conducted and thus demonstrate the potential applicability of our approach to any corpus.

Class	marked contexts			marked + unmarked contexts		
	P	R	F	P	R	F
ORG	0.72	0.69	0.69	0.76	0.76	0.76
LOC	0.74	0.71	0.71	0.75	0.75	0.75
EVT	0.68	0.67	0.67	0.73	0.73	0.73
INF	0.69	0.69	0.68	0.70	0.70	0.70
HUM	0.86	0.86	0.86	0.84	0.84	0.84
Acc	72.69%			76.03%		

Table 5: Precision (P), Recall (R), and F-Measure (F) of classifiers over Corpus B.

Class	marked contexts				marked + unmarked context			
	members		non-members		members		non-members	
	P	R	P	R	P	R	P	R
ORG	0.79	0.52	0.65	0.86	0.78	0.72	0.75	0.80
LOC	0.82	0.55	0.66	0.73	0.78	0.70	0.73	0.80
EVT	0.73	0.57	0.63	0.78	0.74	0.72	0.72	0.73
INF	0.72	0.62	0.66	0.75	0.72	0.65	0.68	0.74
HUM	0.87	0.84	0.84	0.87	0.86	0.82	0.82	0.86

Table 6: Precision (P) and Recall (R) of classification of members and non-members of different lexical classes over Corpus B

Table 6 presents the precision and the recall of each individual classifier over Corpus B both with regard to the members of a given class, and those nouns that are not members of that class. This table allows us to identify more precisely how the unmarked contexts contribute to the error reduction in classification.

According to the results, unmarked contexts allow us to gain an average of 10.2 points in recall for class members, demonstrating that they provide useful information to classifiers, which allows them to cover cases which they were not able to before, most likely due to phenomena such as data sparsity. However, the impact on precision varies between classes, as the inclusion of very frequent information in the vectors representing target nouns may provide additional noise to the classifier (see Section 6.1).

The precision of classification of class members decreases slightly with the inclusion of unmarked contexts, although the differences are not statistically significant ($p<0.1$). However, the precision of the classification of nouns not belonging to the classes considered significantly increases ($p<0.1$) with the inclusion of unmarked contexts in all cases except for HUM. This shows that although unmarked contexts do not contribute to a better definition of the characteristics of individual classes (see Table 6), they allow for a cleaner discrimination of members and non-members of a class, contributing to a better partition of the classification space.

Class	marked contexts		marked + unmarked contexts	
	FN (%)	FP (%)	FN (%)	FP (%)
ORG	23.32	6.71	13.43	9.98
LOC	22.30	5.75	14.74	9.71
EVT	21.91	10.42	13.82	12.97
INF	18.94	12.00	17.26	12.63
HUM	7.79	6.01	8.90	6.45

Table 7: Percentage of False Negatives (FN) and False Positives (FP) in classifiers over Corpus B with and without unmarked contexts.

Table 7 presents the percentage of False Positives (FP), i.e. nouns incorrectly marked as members of the class, and False Negatives (FN), i.e. nouns incorrectly marked as not belonging to a class, in the results of each classifier both with and without the inclusion of unmarked contexts. Again, for each of the classes, except HUM, the inclusion of unmarked contexts decreases the percentage of FN, mirroring a reduction in silence. Yet, there was an increase of FP across all classes, signifying an increase of the noise provided to the classifier. These results are discussed in detail in Section 6.

6 Discussion

In Section 5, we presented the results obtained in our experiments using distributional information regarding both marked and unmarked contexts for the classification of English nouns. Overall, our results show that unmarked contexts either improve accuracy or do not affect classification results. Specifically, the improvements in accuracy are particularly significant for those classes for which there were difficulties to find enough occurrences in marked contexts in previous experiments, i.e. those classes with a higher level of FN when classified without using unmarked contexts. This way, the results confirm our general hypothesis that the distribution of words in unmarked contexts, when considered along with contexts marked towards a lexical semantic class, provides information to improve classifiers, particularly when not enough class-specific information is available. In this section we analyze the results obtained, making apparent the main advantages of our proposal.

6.1 A trade-off between silence and noise

An important result of our experiments is the overall reduction in the negative effect of silence in our classifiers, which decreased by an average of 5.21% (see the difference in terms of FN in Table 7), resulting in an increase in accuracy (see Table 5): as more information is supplied to the classifier, the additional information permits more accurate membership decisions. To illustrate this, we consider examples from the INF, ORG and EVT classes, for which there was not enough information for classification when unmarked contexts were not considered. The inclusion of unmarked contexts provided information resulting in correct classifications.

The INF noun *theorem* illustrates this case: *theorem* occurs 118 times in the corpus, though only 8 times in marked contexts, which was not enough to accurately classify it as a member of the INF class. As this noun occurs in class-marked contexts, but not enough times for the classifier to make an accurate prediction regarding its class membership, we can consider that the lack of enough information provided to the classifier is responsible for its misclassification. However, after the inclusion of information regarding the behavior of this noun in unmarked contexts, the classifier was able to accurately decide for its inclusion as a member of the INF class. This was also observed in the case of the ORG noun *secretariat* and the EVT noun *impulse*, which occur 190 and 154 times, respectively, in the corpus, yet only 8 and 12 times in marked contexts, which was not enough for an accurate classification. Again, the inclusion of information regarding the distribution of these nouns in unmarked contexts provided the classifier with sufficient information to allow for correct classification.

One of the main concerns regarding the use of unmarked distributional information was the introduction of extra noise as a side effect and the way this affects classification results. For the purpose of the work presented in this paper, we define noise as contradictory distributional information, particularly the occurrence of nouns that are not members of a particular class in prototypical contexts of that particular class, which provides misleading information to classifiers. The impact of this misleading information is made apparent by the amount of FP observed in classification results. In contrast, silence has to do with the well known problem of data sparsity, which can be caused by the particular distribution of lexical, and thus strict, though informative, contexts used in cue-based classification tasks, which are often rare in any corpus of any size due to their specificity.

In our experiment, we did identify some cases of nouns correctly ruled out as members of a class when using only marked cues, which were incorrectly classified as class members after the inclusion of unmarked contexts. The slight increase of FP in our results (see Table 7) shows our method does introduce some extra noise into the classifier, although, in the overall results, this is compensated by the larger amount of nouns that were correctly classified after the inclusion of unmarked distributional information (see Tables 4 and 5).

Analyzing the additional FP observed, we identify two different cases: (i) nouns correctly classified using only marked contexts as not belonging to a class based on a borderline probability, which were

incorrectly classified as members of that class when unmarked contexts were also considered, again based on a borderline probability; and (ii) nouns correctly classified as not belonging to a class as they hardly or never occurred in class-marked contexts, but whose behavior in unmarked contexts was similar to that of members of the class being classified, thus providing contradictory information to the classifier and resulting in incorrect classification.

The first case is illustrated by a noun like *biography*, which was predicted not to be a member of the LOC class with a borderline probability score (0.47). The inclusion of unmarked contexts provided information to the classifier, which slightly changed this probability (0.56), and resulted in an incorrect classification. The noun *megalopolis* illustrates the other case. Occurring only 3 times in class-marked contexts of the INF class, this LOC noun had been correctly classified as not belonging to the INF class. Yet, its behavior in unmarked contexts showed more similarities with members of the INF class than with non-members, resulting in its incorrect classification. Illustrating two paradigmatic cases of noise in the results of the classifiers, these examples make apparent how unmarked contexts are sometimes responsible for incorrect class membership decisions, and how further improving their use in classification tasks, particularly in the case of "borderline" classification decisions, remains a promising line of research to explore in the future (see Section 7).

6.2 More robust classification decisions

Besides the reduction of the impact of silence in the results of the classifiers, with the consequent improvements in accuracy, as discussed in the previous section, we also noticed that the introduction of unmarked contexts provided additional information regarding the distribution of nouns that were classified by chance (i.e. correctly classified nouns, with a borderline probability score), resulting in more robust classification decisions. We saw this with the EVT noun *consolidation* and the LOC noun *coalfield*. Each of these nouns was correctly classified using only marked contexts, yet with borderline probability scores: 0.52 and 0.53, respectively. Upon providing information on unmarked contexts to the classifier, these nouns continued to be correctly classified but with much higher probability scores, and thus more reliable: 0.75 and 0.76, respectively.

These examples are considerably different from those discussed in Section 6.1, as these are far from being cases of silence. In fact, the EVT noun *consolidation* occurs 312 times in the corpus and 317 times in marked contexts while the LOC noun *coalfield* occurs 52 times in the corpus and 53 times in marked contexts[4]. In both cases, almost all of the occurrences in marked contexts were found to be in only one cue, which was therefore not strongly valued by the classifier, as few correlations between the evidence available could be made, hence the low probability scores observed. The inclusion of unmarked distributional information provides "bridging information", allowing for more reliable classifications, which is crucial to consider especially when the ultimate goal of improving and tuning classification systems is to employ classification results for the automatic production of language resources (see Section 1).

6.3 Classification results unevenly affected by unmarked contexts

As made apparent by the results, the contribution of unmarked contexts to the classification of different semantic classes is not always the same. For example, we observed that classes whose members demonstrated a more heterogeneous linguistic behavior, such as the ORG, LOC or EVT classes, improve more with the inclusion of unmarked distributional information than classes with a more homogeneous distributional behavior. To make our statement clearer, we claim that some nominal classes are composed of nouns that tend to occur in a wider range of contexts, thus displaying a more heterogeneous and disperse distributional behavior. This heterogeneity is made apparent by an analysis of the overall distribution of the marked cues between the members of each lexical semantic class. In contrast with heterogeneous noun classes, other classes are composed of members that display a more homogeneous collective behavior that is more easily captured by distributional approaches[5].

[4] Note that a single occurrence in corpus data can activate more than one cue considered in our experiments (for instance, in the case of a target noun that has a marked suffix and simultaneously occurs in a marked syntactic construction), hence the higher amount of occurrences in cues than overall occurrences in the corpus discussed in the examples introduced in this paragraph.

[5] Our analysis of the data showed that the dispersion of distributional behavior is independent of frequency.

Analyzing the distribution of cues between class members in Corpus B, we identified, in each class, a set of cues that occurred with the majority of nouns of the class, and which we will consider to represent the core linguistic behavior of each specific class. We also observed the amount of cues included in this set differed considerably from class to class (see Figure 1). Thus, the larger the amount of marked contexts shared by the majority of the members of a class, the more homogeneous we can claim their behavior to be.

In the specific case of the classes considered in this paper, 30.7% of the cues for the HUM class are shared by the majority of HUM nouns, while 26.6%, 13.3%, 9.5% and 9.1% of the cues for the INF, ORG, EVT and LOC classes, respectively, are shared by the majority of the nouns of these classes, as represented in Figure 1. An effect of a class collectively having a more heterogeneous linguistic behavior is that the evidence regarding each of its marks will typically be more disperse and, as a result, often not strong enough to be considered by classifiers, which explains the improvement introduced by unmarked contexts. In contrast, classes like HUM are composed of nouns that generally occur in a common set of prototypical contexts of that class. Thus, on the one hand, identifying contexts that mirror the prototypical behavior of that class is more straightforward and, on the other, class members almost always show enough occurrences in such contexts to be accurately classified.

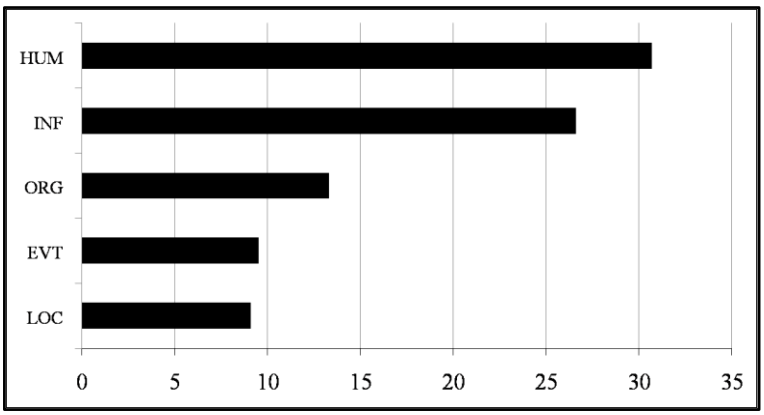


Figure 1: Percentage of cues shared by the majority of class members, per class

Additionally, there are also strong marks based on suffixes and degree of grammaticalization for the HUM class (as demonstrated in Bel *et al.* (2012)), which can be more readily captured by these more available marked contexts. For instance, on the one hand, suffixes, such as: *"-er"* and *"-or"* are indicative of many HUM type nouns (e.g. "doctor", "painter", "officer", etc.) while the preposition *"during"*, when preceding a nominal phrase, is very indicative of occurrences of EVT nouns. These examples are both instances of features that can be easily identified for inclusion in a feature vector, readily providing a large amount of class-indicative information. On the other hand, there are other types of features that although indicative, result in a much sparser vector because of their reliance of occurrence within corpus data. For instance the occurrence as the subject of an agentive verb, which is considered an indicative feature for the ORG class, does not necessarily occur readily with all members of the class, thus making marked contexts that provide a homogeneous representation of the class more difficult to capture.

In this way, the inclusion of extra contexts (e.g. unmarked contexts) are rendered ineffective when class membership decisions are already accurately made to a great extent (in our case 86.19% of the times) based on the information provided by marked contexts. This is consistent with the stability of the results reported for the HUM class in the different experiments performed, which did not demonstrate any significant changes with the inclusion of unmarked contexts.

7 Conclusions and Final Remarks

Our main goal in this paper was to evaluate how unmarked contexts can be used to improve the accuracy of nominal lexical semantic classification tasks. Departing from the hypothesis that these contexts can provide additional information to classifiers when there is not enough distinctive co-occurrence information available, the results reported demonstrate the use of unmarked contexts, which are typically discarded as non-discriminatory, can significantly improve the results of lexical semantic classification when considered along with marked contexts. Our results also show that using both types of

distributional information (marked and unmarked) is crucial to reduce the sparse data problem, thus improving classification (see increase in classification accuracy in Tables 4 and 5). Moreover, in our experiments, we apply this method to two independent corpora obtaining comparable results and thus demonstrating the robustness and transferability of our approach to any dataset.

The higher accuracy and error reduction achieved with the inclusion of unmarked contexts constitute a significant improvement with respect to the state of the art (Bel, 2010; Bel *et al.*, 2012; Romeo *et al.*, 2014), contributing particularly to the increase of accuracy and reliability of classifiers for classes that exhibit more disperse linguistic behavior. Moreover, the approach depicted here leaves room for further improvements and future work, particularly with regard to designing strategies to minimize the introduction of borderline false positives in classification.

One promising line of research to explore is the optimization of the inclusion of unmarked contexts in classification decisions. As detailed in the discussion, for the experiments depicted in this paper, we did not expect particular marked or unmarked features to be more useful than others, as we relied on the correlation of all the distributional information considered for each specific class to be indicative of class membership.

Another aspect to be further explored consists of determining the most effective amount of unmarked contexts to be provided to automatic systems. Building on the demonstration of the positive contribution of unmarked contexts in classification tasks, as indicated by the results obtained in the work depicted in this paper (see Section 5), we will start by determining the specific contribution to classification of each unmarked feature used. In this way, we would check whether there is a context, within our set, that is not contributing to the classification, in order to establish a threshold to systematically identify the information that is not relevant or whether we need to widen/relax our frequency criterion to include more unmarked contexts with the goal of elaborating a set of information to be as robust as possible, thus resulting in more accurate and more reliable classification decisions.

Finally, we believe the results obtained make a clear contribution towards the automatic production of high-quality language resources, which will benefit any NLP system that requires information on lexical semantic classes as an input.

Acknowledgements

This work was funded with the support of the SUR of the DEC of the Generalitat de Catalunya and the European Social Fund, by SKATER TIN2012-38584-C06-05, and by Fundação para a Ciência e a Tecnologia (FCT) post-doctoral fellowship SFRH/BPD/79900/2011.

References

Abbès, S. B., Zargayouna, H. and Nazarenko, A. 2011. Evaluating Semantic Classes Used for Ontology Building and Learning from Texts. In *Proceedings in the International Conference on Knowledge Engineering and Ontology Development*. Paris, France.

Agirre, E., Bengoetxea, K., Gojenola, K. and Nivre, J. 2011. Improving Dependency Parsing with Semantic Classes. In *Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics, (ACL-HLT 2011).* Portland, Oregon.

Baroni, M., Bernardini, S., Ferraresi, A. and Zanchetta, E. 2009. The WaCky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation, 43*(3): 209-226.

Bel, N. 2010. Handling of Missing Values in Lexical Acquisition, In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010),* Valletta, Malta.

Bel, N., Romeo, L. and Padró, M. 2012. Automatic Lexical Semantic Classification of Nouns. In *Proceedings of the 8th Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey.

Bullinaria, J. A. and Levy, J. 2012. Extracting semantic representations from word co-occurrence statistics: Stoplists,stemming and svd. *Behavior Research Methods*,44:890-907.

Bybee, J. 2010. *Language, Usage and Cognition*. Cambridge University Press, Cambridge.

Cooke, N. and Gillam, L. 2008. Distributional lexical semantics for stop lists. In *Proceedings of the 2008 BCS-IRSG conference on Corpus Profiling* (IRSG'08), Anne De Roeck, Dawei Song, and Udo Kruschwitz (Eds.). British Computer Society, Swinton, UK.

Goldberg, A. E. 2006. *Constructions at Work. The Nature of Generalization in Language*. Oxford University Press: Oxford.

Harris, Z. 1954. Distributional Structure. *Word*, 10(23): 146-162.

Jakobson, R. 1971. *Selected Writings II: Word & Language*. Mouton, The Hague.

Krenn, B. and Samuelsson, C. 1997. *The Linguist's Guide to Statistics – Don't Panic*. http://nlp.stanford.edu/fsnlp/dontpanic.pdf

Landwehr, N., Hall, M. and Frank, E. 2005. Logistic Model Trees. *Machine Learning*, 95(1-2): 161-205.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D. and Miller, K. J. 1990. Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4): 235-244.

Pecina, P., Toral, A., Way, A., Papavassiliou, V., Prokopidis, P. and Giagkou, M. 2011. Towards Using Web-Crawled Data for Domain Adaptation in Statistical Machine Translation. In *Proceedings of the 15th Conference of the European Association for Machine Translation (EAMT 2011)*. Leuven, Belgium.

Pustejovsky, J. 1995. *Generative Lexicon*. The MIT Press, Cambridge.

Quinlan, R. J. 1993. C4.5: *Programs for Machine Learning. Series in Machine Learning*. Morgan Kaufman: San Mateo.

Romeo, L., Lebani G. E., Bel, N. and Lenci, A. 2014 Choosing which to use? A study of distributional models for nominal lexical semantic classification. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*. Reykjavik, Iceland.: 4366-4373.

Rumshisky, A., Grinberg, V. and Pustejovsky, J. 2007. Detecting Selectional Behavior of Complex Types in Text. In *Proceedings of the 4th International Workshop on Generative Lexicon*. Paris, France.

Turney, P. D. and Pantel, P. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141-188.

Witten, I. H. and Frank, E. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. 2nd Edition, Morgan Kaufmann: San Francisco.

Skill Inference with Personal and Skill Connections

Zhongqing Wang[†], Shoushan Li[†*], Hanxiao Shi[‡], and Guodong Zhou[†]
[†] Natural Language Processing Lab, School of Computer Science and Technology,
Soochow University, China
[‡] School of Computer Science and Information Engineering,
Zhejiang Gongshang University, China
{wangzq.antony, shoushan.li}@gmail.com
hxshi@mail.zjgsu.edu.cn, gdzhou@suda.edu.cn

Abstract

Personal skill information on social media is at the core of many interesting applications. In this paper, we propose a factor graph based approach to automatically infer skills from personal profile incorporated with both personal and skill connections. We first extract personal connections with similar academic and business background (e.g. co-major, co-university, and co-corporation). We then extract skill connections between skills from the same person. To well integrate various kinds of connections, we propose a joint prediction factor graph (JPFG) model to collectively infer personal skills with help of personal connection factor, skill connection factor, besides the normal textual attributes. Evaluation on a large-scale dataset from LinkedIn.com validates the effectiveness of our approach.

1 Introduction

With the large amount of user-generated content (UGC) published online every day in the context of social networks (Tan et al., 2011; Luo et al., 2013), such online social networks (e.g., Twitter, Facebook, and LinkedIn) have significantly enlarged our social circles and much affected our everyday life. One popular and important type of UGC is the personal profile, where people post their detailed information, such as education, experience and other personal information, on online portals. Social websites like Facebook.com and LinkedIn.com have created a viable business as profile portals, with the popularity and success largely attributed to their comprehensive personal profiles.

Obviously, online personal profiles can help people connect with others of similar backgrounds and provide valuable resources for businesses, especially for personnel resource managers to find talents (Yang et al., 2011a; Guy et al., 2010). In the profiles, the personal skill information is the most important aspect to reflect the expertise of a person. However, few social platforms allow users to manually attach such personal skill information into their personal profiles. For example, in our collected dataset, 91.8% skills appear less than 10 times. Even the distribution of the top 10 frequently occurring skills is asymmetric, and only 43.1% people attach skills on their profiles. For this regard, it is highly desirable to develop reliable methods to automatically infer personal skills for personal profiles.

Although it is straightforward to recast skill inference as a standard text classification problem, i.e., predicting the skills with the profile text alone, personal profiles usually are poorly organized, even with critical information missing. Thus, it is challenging to infer skills given the limited information from the profile texts. We propose two assumptions to address above challenges by incorporating additional connection information between persons and skills:

- People are always connected to others with similar academic and business backgrounds (e.g. co-major, co-corporation). For example if there is co-major, co-university, or co-corporation relationship between two persons, it is very likely that they may share similar skills. Therefore, it is reasonable to resort to personal connection information to improve the performance of skill inference.

*corresponding author

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 520–529, Dublin, Ireland, August 23-29 2014.

- One person tends to have some related skills. For example, it is very likely that C++, C, and Python programming languages may co-occur in the one's profile, i.e., if a person has skill C++, it is highly possible that he would have the skills such as C or Python. Thus, it is useful to integrate skill connection information when inferring personal skills.

Based on these assumptions, we propose a Joint Prediction Factor Graph (JPFG) model, which collectively predicts personal skills with help of both personal and skill connections. In particular, the JPFG model provides a general framework to integrate three kinds of knowledge, i.e. local textual attribute functions of an individual person, personal connection factors between persons, and skill connection factors between skills, in collectively inferring personal skills. Specially, we extract personal connections with similar academic and business background (e.g. co-major, co-corporation). We then extract skill connections between skills from same person. Evaluation on a large-scale data set from LinkedIn.com indicates that our JPFG model can significantly improve the performance of personal skill inference.

The remainder of this paper is structured as follows. We review the related work in Section 2. In Section 3, we introduce the data collection. In Section 4, we give the problem definition and some analysis on the task of personal skill reference. In Section 5, we propose the JPFG model and corresponding algorithms for parameter estimation and prediction. In Section 6, we present our experimental results. In Section 7, we summarize our work and discuss future directions.

2 Related Works

In this section, we briefly review related studies in expert finding, social tag suggestion and factor graph model.

2.1 Expert Finding

Expert finding aims to find right persons with appropriate skills or knowledge, i.e. "Who are the experts on topic X?" TREC-2005 and TREC-2006 have provided a common platform for researchers to empirically evaluate methods and techniques on expert finding (Soboroff et al, 2006; Zhang et al., 2007a).

In the literature, expert finding tends to consider each skill individually and seeks the most authority experts for each skill. Thus, expert finding is always considered as a ranking process, i.e., ranking the experts from the candidates who are most suitable for the skill (Balog and Rijke, 2007). For example, Campbell et al. (2003) investigated the issue of expert finding in an email network. They utilized the link between email authors and receivers to improve the expert finding performance.

Besides that link structure-based algorithms, such as PageRank and HITS, are employed to analyze the relationship of the link-relationship graph, social networks are utilized to improve the performance of expert finding. Zhang et al. (2007a) proposed a unified propagation-based approach to address the issue of expert finding in a social network, considering both personal local and network information (e.g. the relationship between persons).

Expert finding is in nature different from skill inference. Our study predicts various skills attachable to a person collectively with both personal and skill connections among people. One distinguishing characteristics of our study is that several skills from a person are simultaneously modeled and the relationship among these skills is fully leveraged in the inference.

2.2 Social Tag Suggestion

Social tag suggestion aims to extract proper tags from social media and can thus help people organize their information in an unconstrained manner (Ohkura et al., 2006; Si et al., 2010). Ohkura et al. (2006) created a multi-tagger to determine whether a particular tag from a candidate tag list should be attached to a weblog. Lappas et al. (2011) proposed a social endorsement-based approach to generate social tags from Twitter.com and Flickr.com where various kinds of information in recommendations and comments are used. Liu et al. (2012) propose a probabilistic model to connect the semantic relations between words and tags of microblog, and takes the social network structure as regularization. Li et al., (2012) propose to model context-aware relations of tags for suggestion by regarding resource content as context of tags.

Different from above researches, our study is forced on skill inference instead of traditional tag suggestion. Basically, the social connections in skill inference are much different from those in social tagging. In our study, we use co-major, co-title and other academic and business relationships to build the social connections. Meanwhile, there are also few researches concern to propose a joint model to leverage both personal and skill connections.

2.3 Factor Graph Model

Among various approaches investigated in social networks in the last several years (Leskovec et al., 2010; Lu et al., 2010; Lampos et al., 2013; Guo et al., 2013), Factor Graph Model (FGM) becomes an effective way to represent and optimize the relationship in social networks (Dong et al., 2012; Yang et al., 2012b) via a graph structure. Tang et al. (2011a) and Zhuang et al. (2012) formalized the problem of social relationship learning as a semi-supervised framework, and proposed Partially-labeled Pairwise Factor Graph Model (PLP-FGM) for inferring the types of social ties. Tang et al. (2013) further proposed a factor graph based distributed learning method to construct a conformity influence model and formalize the effects of social conformity in a probabilistic way.

Different from previous studies, this paper proposes a pairwise factor graph model to collectively infer personal skills with both social connection factor and skill connection factor.

3 Data Construction

We collect our data set from LinkedIn.com. It contains a large number of personal profiles generated by users, containing various kinds of information, such as personal Summary, Experience, Education, and Skills & Expertise. We do not collect personal names in public profiles to protect people's privacy.

The dataset contains 7,381 personal profiles, among which only 3,182 profiles (43.1% of all the profiles) show the Skills & Expertise field. In this study, we adopt only these profiles in all our experiments. As a result, we get 6,863 skills in total, among which 6,299 skills (91.8% of them) appear less than 10 times. Among the remaining 564 skills, we select top 10 frequently occurring skills as the candidate personal skills in this study (Since the remaining 554 skills only appear less than 250 times in total, it is difficult to build an effective classifier for them). Table 1 illustrates the statistics.

Skill	Number	Ratio
Semiconductors	948	0.298
IC	369	0.116
Thin Films	328	0.103
Characterization	326	0.102
CMOS	311	0.098
Matlab	287	0.090
Microsoft Office	283	0.089
Manufacturing	278	0.087
Design of Experiments	262	0.082
Semiconductor Industry	250	0.079

Table 1: The distribution of the candidate personal skills

From Table 1, we can see that the skill distribution in the personal profiles is asymmetric. For example, the Semiconductor skill occurs about 1,000 times, taking 29.8%, while the Semiconductor Industry skill occurs 250 times only, taking 7.9%.

4 Problem Definition and Analysis

Before presenting our approach for skill inference, we first give the definition of the problem, and convey a series of discoveries we observed from the data.

4.1 Problem Definition

We first introduce some necessary definitions and then formulate of the problem.

Definition 1: *Skill inference*. In principle, we cast skill inference as a skill prediction problem. Since one person might have several skills, we build several vectors for a person and each vector is designed to determine whether the corresponding skill is appropriate for the person or not ("Positive" means that the person has the target skill, whereas "Negative" stands for the opposite). Note that the number of vectors for a person is equal to the number of candidate skills. For example, suppose we have m persons and n candidate skills in the dataset, we totally build vectors to represent if these skills are attached in these persons' profiles.

Definition 2: *Textual information*. We use texts of Summary and Experience as the textual information for our research. Texts of Summary and Experience are unstructured information, while texts of Skills & Expertise are structured information. However, some skills in the Skill & Expertise fields may not be mentioned in the Summary and Experience fields.

Definition 3: *Personal connections*. We can explicitly extract four kinds of personal relationships between two persons from the Education and Experience fields, as follows:

- *co_major*, which denotes that two persons have the same major at school
- *co_univ*, which denotes that two persons graduated from the same university
- *co_title*, which denotes that two persons have the same title in a corporation.
- *co_corp*, which denotes that two persons work in the same corporation.

Definition 4: *Skill connections*. We extract skill connections from same person. That is, if two vectors are from the same person with different skills, we consider these two vectors share skill connections (e.g. John has IC and Thin Films skills).

Learn task: Given the textual information of each profile, the personal connections between profiles, and skill connections of skill from same persons, the goal is to infer the skill through the above information.

To learn the skill inference model, there are several requirements. First, the skills of persons are related to multiple factors, e.g., network structure, personal connections, and skill connections, it is important to find a unified model which is able to incorporate all the information together. Second, the algorithm to learn the inference model should be efficient. In practice, the scale of the social network might be very large.

4.2 Statistics and Observations

In the following, we give some statistics and observations on personal and skill connections.

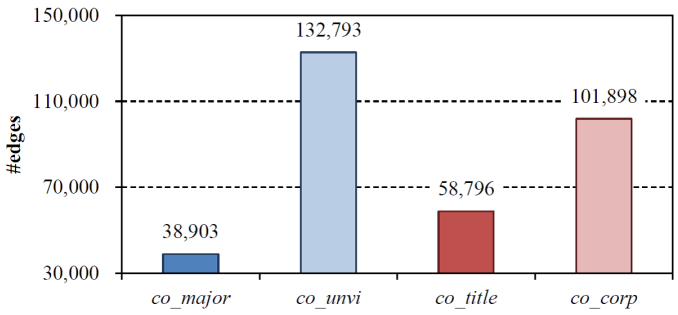


Figure 1: The statistic of personal connection edges in our dataset

Statistics of personal connections: Figure 1 gives the statistics of personal connection edges. It shows that with 3,182 profiles, there exist 332,390 personal connection edges. Besides, among all the

four relations, co_major, co_unvi, co_title, and co_corp occupy 11.7%, 40.0%, 17.7% and 30.6% respectively.

Observations of skills connections: To validate the tendency of a person sharing similar skills, we use PMI (Point-wise Mutual Information) to measure the co-occurrence between two skills. As a popular way to measure the co-occurrence between a pair (Turney, 2002), PMI is calculated as follows:

$$PMI(i,j) = \log\left(N\frac{P(i\&j)}{P(i)P(j)}\right) \tag{1}$$

N is the number of profiles, $P(i\&j)$ denotes the probability of the skills (i.e., i and j) co-occurrence in a person's profile, while $P(i)$ denotes the probability of the skill i appearing in a person's profile.

Skill i	**Skill** j	**PMI**
C	COMS	1.711
Thin Films	Characterization	1.624
Thin Films	Design of Experiments	1.543
Semiconductor Industry	IC	1.345
Semiconductor Industry	Design of Experiments	1.345
IC	Microsoft Office	-2.390
CMOS	Microsoft Office	-2.627
Semiconductor Industry	Matlab	-3.112
Average PMI score		0.190

Table 2: The top-5 and bottom-3 co-occurred skill pairs with their PMI scores

Table 2 lists the top-5 and bottom-3 co-occurred skill pairs with their PMI scores, together with the average PMI score. From this table, we can see that if two skills are related, e.g., "IC" and "CMOS", these two skills tend to co-occur in one person's profile, vice versa.

5 Joint Prediction Factor Graph Model

In this section, we propose a Joint Prediction Factor Graph (JPFG) model for learning and predicting the skills with personal and skill connection information besides local textual information.

5.1 Model

We formalize the problem of skill prediction using a pairwise factor graph model, and our basic idea of defining the correlations is to use different types of factor functions (i.e., personal connection factor, and skill connection factor). Here, the objective function $P_\theta(Y|X,G)$ is defined based on the joint probability of the factor functions, and the problem of collective skill inference model learning is cast as learning model parameters θ that maximizes the joint probability of skills based on the input continuous dynamic network.

Since directly maximizing the conditional probability $P_\theta(Y|X,G)$ is often intractable, we factorize the "global" probability as a product of "local" factor functions, each of which depends on a subset of the variables in the graph (Tang et al., 2013). In particular, we use three kinds of functions to represent the local textual information of the vector (local textual attribute function), personal connection information between vectors (personal connection factor) and skill connection information between skills (skill connection factor), respectively. We now briefly introduce the ways to define the above three functions.

Local textual attribute functions $f(x_{ij},y_i)_j$ **:** It denotes the attribute value associated with each person i. Here, we define the local textual attribute as a feature (Lafferty et al., 2001) and accumulate all the attribute functions to obtain local entropy for a person:

$$\frac{1}{Z_1}\exp\left(\sum_i\sum_k \alpha_k f_k(x_{ik},y_i)\right) \tag{2}$$

Where α_k is the function weight, representing the influence degree of the attribute k. For simplicity, we use word unigrams of a text as the basic textual attributes.

Personal connection factor function $g(y_i, y_j)$ **:** For the personal correlation factor function, we define it through the pairwise network structure. That is, if a person i and another person j have a personal relationship, we define a personal connection factor function as follows:

$$g(y_i, y_j) = \exp\left\{\beta_{ij}(y_i - y_j)^2\right\} \tag{3}$$

The personal connections are defined Section 4, i.e., co_major, co_univ, co_title, and co_corp. We define that if two persons have at least one personal connection edge, they have a personal relationship. In addition, β_{ij} is the weight of the function, representing the influence degree of i on j.

Skill connection factor function $h(y_i, y_j)$**:** For the skill connection factor function, we define it through the pairwise network structure. That is, if vector i and vector j are from the same person with different skills, we define their skill connection influence factor function as follows:

$$h(y_i, y_j) = \exp\left\{\gamma_{ij}(y_i - y_j)^2\right\} \tag{4}$$

Where γ_{ij} is the function weight, representing the influence degree of i on j.

By the above defined correlations, we can construct the graphical structure in the factor model. According to the Hammersley-Clifford theorem (Hammersley and Clifford, 1971), we integrate all the factor functions and obtain the following log-likelihood objective function:

$$\begin{aligned} L(\theta) &= \log_\theta P(Y|X,G) \\ &= \frac{1}{Z_1}\sum_i\sum_k \alpha_k f_k(x_{ik}, y_i) \\ &+ \frac{1}{Z_2}\sum_i \sum_{j\in NB(i)} \exp\left\{\beta_{ij}(y_i - y_j)^2\right\} \\ &+ \frac{1}{Z_3}\sum_i \sum_{k\in SAME(i)} \exp\left\{\gamma_{ik}(y_i - y_k)^2\right\} - \log Z \end{aligned} \tag{5}$$

Where (i, j) is a pair derived from the input network, $Z = Z_1 Z_2 Z_3$ is a normalization factor and $\theta = (\{\alpha\}, \{\beta\}, \{\gamma\})$ indicates a parameter configuration, $NB(i)$ denotes the set of social relationship neighbors nodes of i (personal connection), and $SAME(i)$ denotes the set of the node with the same person of i (skill connection).

5.2 Learning and Prediction

Model Learning: Learning of the factor model is to find the best configuration for free parameters $\theta = (\{\alpha\}, \{\beta\}, \{\gamma\})$ that maximizes the log likelihood objective function $L(\theta)$.

$$\theta^* = \arg\max L(\theta) \tag{6}$$

As the network structure in a social network can be arbitrary (e.g. possible of containing cycles), we use the Loopy Belief Propagation (LBP) algorithm (Tang et al., 2011a) to approximate the marginal distribution. To explain how we learn the parameters, we can get the gradient of each β_k with regard to the objective function (Eq. 5), taking β (the weight of the personal connection factor function $g(y_i, y_j)$) as an example:

$$\frac{L(\theta)}{\beta_k} = E[g(i,j)] + E_{\beta_k P(Y|X,G)}[g(i,j)] \tag{7}$$

Where $E[g(i,j)]$ is the expectation of factor function $g(i,j)$ given the data distribution in the input network and $E_{\beta_k P(Y|X,G)}[g(i,j)]$ represents the expectation under the distribution learned by the model, i.e., $P(y_i|X,G)$.

With the marginal probabilities, the gradient is obtained by summing up all triads (similar gradients can be derived for parameter α_k and γ_{ij}). It is worth noting that we need to perform the LBP process

twice in each iteration. The first run to estimate the marginal distribution of unknown variables $y_i =?$ and the second one is to estimate the marginal distribution over all pairs. Finally, with the obtained gradient, we update each parameter with a learning rate η.

Skill Prediction: We can see that in the learning process, additional loopy belief propagation is used to infer the label of unknown relationships. After learning, all unknown skills are assigned with labels that maximize the marginal probabilities (Tang et al., 2011b), i.e.,

$$Y^* = \arg\max L(Y|X, G, \theta) \tag{8}$$

6 Experimentation

In this section, we first introduce the experimental setting, and then evaluate the performance of our proposed JPFG model with both personal and skill connection information.

6.1 Experimental Setting

As described in Section 3, the experimental data are collected from LinkedIn.com. With top 10 frequently used skills as candidate skills in all our experiments, we randomly select 2,000 profiles as training data and 1,000 profiles as testing data.

Though positive and negative samples of each skill are imbalanced (In this paper, the number of the negative samples is much larger than that of the positive samples), we select balanced testing and training samples for each skill. Following models are implemented and compared.

- *Keyword*, for each profile, we consider the profile attached with the skill, only if the text of the skill appears on the profile article with textual information.

- *MaxEnt*, which first uses local textual information as features to train a maximum entropy (ME) classification model, and then employs the classification model to predict the skills in the testing data set. The ME algorithm is implemented with the mallet toolkit [1].

- *JPFG*, exactly our proposed model, which jointly predicts personal skills with local textual information, personal connection and skill connection.

For performance evaluation, we adopt Precision (P.), Recall (R.) and F1-Measure (F1.).

6.2 Comparison with Baselines

Our first group of experiments is to investigate whether the JPFG model is able to improve skill inference and whether the personal and skill connections are useful. The experimental results are shown in Table 3. From the table we can find that as some skills may not be mentioned on the Summary and Experience fields directly, the performance of the Keyword approach is far from satisfaction. As incorporating personal and skill connections, the JPFG model yields a much higher F1-measure, which improves the performance with about 6.8% gain than the MaxEnt model.

6.3 Performance of JPFG with Different Training Data Sizes

After we evaluate the effective of the JPFG model with the large-scale training data, we carry out experiments to test the effect of the JPFG model with different training data sizes. Experiment results are shown in Figure 3. It shows that the JPFG model with both personal and skill connections always outperform the two baseline models. Impressively, our JPFG model using 20% training data outperforms MaxEnt using 100% training data.

[1] http://mallet.cs.umass.edu/

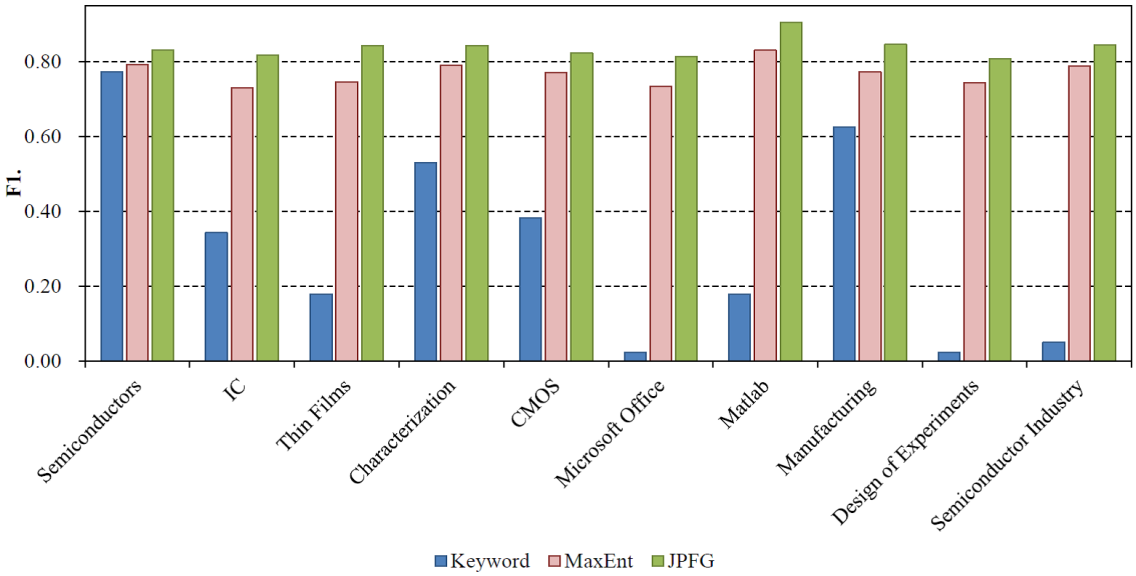


Figure 2: The performance of different methods for skill inference

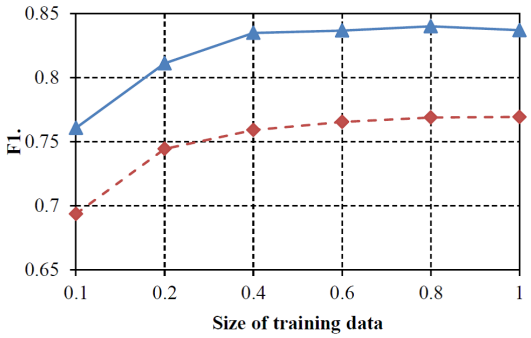


Figure 3: The performance of JPFG with different training data sizes

6.4 Connections Contribution Analysis

Personal connections and skill connections can be also used to build the factor graph models to infer the skills. We therefore want to compare our JPFG model with the factor graph model with only consider the personal connections or skill connections, and analysis the contribution of each kinds of connection. Specifically, MaxEnt-Personal employs the personal connections as additional features incorporated with textual features to build the maximum entropy classification. FGM-Personal is a simplified version of the JPFG model, which only employs textual attribute functions and personal connection factor functions to build the factor graph model. Likewise, FGM-Skill only employs textual attribute functions and skill connection factor functions to build the factor graph model. Table 3 shows the experiment results.

System	P.	R.	F1.
MaxEnt	0.744	0.797	0.769
MaxEnt-Personal	0.758	0.812	0.783
FGM-Personal	0.765	0.817	0.790
FGM-Skill	0.704	**0.967**	0.815
JPFG	**0.780**	0.905	**0.837**

Table 3: The contribution of connections

From Table 3, we can observe that, 1) Both FGM-Personal and FGM-Skill outperform the baseline

MaxEnt approach. It shows that both personal connections and skill connections are helpful for skill inference; 2) MaxEnt-Personal and FGM-Personal outperform the baseline MaxEnt approach, it show that personal connections are helpful for inferring skills, and as considering the global optimization, FGM-Personal is more effective; 3) FGM-Skill built on the skill connections is more effective than MaxEnt-Personal and FGM-Personal, it show that skill connections are more useful than personal connections; 4) JPFG model outperforms both FGM-Personal and FGM-Skill, it suggests that we should incorporate both personal and skill connections to the factor graph model when we infer the skills from profile.

7 Conclusion

In this study, we propose a novel task named personal skill inference, which aims to determine whether a person takes a specific skill or not. To address this task, we propose a joint prediction factor graph model with help of both personal and skill connections besides local textual information. Evaluation on a large-scale dataset shows that our joint model performs much better than several baselines. In particular, it shows that the performance on personal skill inference can be greatly improved by incorporating skill connection information.

The general idea of exploring personal and skill connections to help predict people's skills represents an interesting research direction in social networking, which has many potential applications. Besides, as skill information of a person is normally incomplete and fuzzy, how to better infer personal skills with weakly labeled information is challenging.

Acknowledgements

This research work is supported by the National Natural Science Foundation of China (No. 61273320, No. 61331011, and No. 61375073), National High-tech Research and Development Program of China (No. 2012AA011102), Zhejiang Provincial Natural Science Foundation of China (No. LY13F020007), the Humanity and Social Science on Young Fund of the Ministry of Education (No. 12YJC630170).

We thank Dr. Jie Tang and Honglei Zhuang for providing their software and useful suggestions about PGM. We thank Prof. Deyi Xiong for helpful discussions, and we acknowledge Dr. Xinfang Liu, and Yunxia Xue for corpus construction and insightful comments. We also thank anonymous reviewers for their valuable suggestions and comments.

References

Balog K and M. Rijke. 2007. Determining Expert Profiles (With an Application to Expert Finding). In *Proceedings of IJCAI-07*.

Campbell C, P. Maglio, A. Cozzi, and B. Dom. 2003. Expertise Identification Using Email Communications. In *Proceedings of CIKM-03*.

Dong Y., J. Tang, S. Wu, J. Tian, N. Chawla, J. Rao, and H. Cao. 2012. Link Prediction and Recommendation across Heterogeneous Social Networks. In *Proceedings of ICDM-12*.

Guo W., H. Li, H. Ji, and M. Diab. 2013. Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media. In *Proceedings of ACL-13* .

Guy I., N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel. 2010. Social Media Recommendation based on People and Tags. In *Proceedings of SIGIR-10* .

Hammersley J. and P. Clifford. 1971. Markov Field on Finite Graphs and Lattices, *Unpublished manuscript*.

Helic D. and M. Strohmaier. 2011. Building Directories for Social Tagging Systems. In *Proceedings of CIKM-2011*.

Lafferty J, A. McCallum, and F. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of ICML-01*.

Lampos V., D. Preo?iuc-Pietro, and T. Cohn. 2013. A User-centric Model of Voting Intention from Social Media. In *Proceedings of ACL-13*.

Lappas T., K. Punera, and T. Sarlos. 2011. Mining Tags Using Social Endorsement Networks. In *Proceedings of SIGIR-11*.

Li H., Z. Liu, and M. Sun. 2012. Random Walks on Context-Aware Relation Graphs for Ranking Social Tags. In *Proceedings of COLING-12*.

Liu Z., X. Chen, and M. Sun. 2011. A Simple Word Trigger Method for Social Tag Suggestion. In *Proceedings of EMNLP-2011*.

Liu Z., C. Tu, and M. Sun. 2012. Tag Dispatch Model with Social Network Regularization for Microblog User Tag Suggestion. In *Proceedings of COLING-12*.

Lu Y., and P. Tsaparas, A. 2010. Ntoulas and L. Polanyi. 2010. Exploiting Social Context for Review Quality Prediction. In *Proceedings of WWW-10*.

Luo T., J. Tang, J. Hopcroft, Z. Fang, and X. Ding. 2013. Learning to Predict Reciprocity and Triadic Closure in Social Networks. *ACM Transactions on Knowledge Discovery from Data*. vol.7(2), Article No. 5.

Murphy K., Y. Weiss, and M. Jordan. 1999. Loopy Belief Propagation for Approximate Inference: An Empirical Study. In *Proceedings of UAI-99* .

Ohkura T., Y. Kiyota and H. Nakagawa. 2006. Browsing System for Weblog Articles based on Automated Folksonomy. In *Proceedings of WWW-06*.

Si X., Z. Liu, and M. Sun. 2010. Explore the Structure of Social Tags by Subsumption Relations. In *Proceedings of COLING-10*.

Soboroff I., A. Vries and N. Craswell. 2006. Overview of the TREC 2006 Enterprise Track In *Proceedings of TREC-06*.

Turney P. 2002. Thumbs up or Thumbs down? Semantic Orientation Applied to Unsupervised Classification of reviews. In *Proceedings of ACL-02*.

Tan C., L. Lee, J. Tang, L. Jiang, M. Zhou, and P. Li. 2011. User-Level Sentiment Analysis Incorporating Social Networks. In *Proceedings of KDD-11*.

Tang W., H. Zhuang, and J. Tang. 2011a. Learning to Infer Social Ties in Large Networks. In *Proceedings of ECML/PKDD-11*.

Tang J., Y. Zhang, J. Sun, J. Rao, W. Yu, Y. Chen, and A. Fong. 2011b. Quantitative Study of Individual Emotional States in Social Networks. *IEEE Transactions on Affective Computing*. vol.3(2), Pages 132-144.

Tang J., S. Wu, J. Sun, and H. Su. 2012. Cross-domain Collaboration Recommendation. In *Proceedings of KDD-12*.

Tang J., S. Wu, and J. Sun. 2013. Confluence: Conformity Influence in Large Social Networks. In *Proceedings of KDD-13*.

Xing E, M. Jordan, and S. Russell. 2003. A Generalized Mean Field Algorithm for Variational Inference in Exponential Families. In *Proceedings of UAI-03*.

Yang S., B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha. 2011a. Like like alike - Joint Friendship and Interest Propagation in Social Networks. In *Proceedings of WWW-11*.

Yang Z., K. Cai, J. Tang, L. Zhang, Z. Su, and J. Li. 2011b. Social Context Summarization. In *Proceedings of SIGIR-11*.

Zhang J., J. Tang, and J. Li. 2007a. Expert Finding in A Social Network. In *Proceedings of the Twelfth Database Systems for Advanced Applications (DASFAA-2007)*.

Zhang J., M. Ackerman, and L. Adamic. 2007b. Expertise Networks in Online Communities: Structure and Algorithms. In *Proceedings of TREC-07*.

Zhuang H, J. Tang, W. Tang, T. Lou, A. Chin, and X. Wang. 2012. Actively Learning to Infer Social Ties. In *Proceedings of Data Mining and Knowledge Discovery (DMKD-12)*, vol.25 (2), pages 270-297.

Jointly or Separately: Which is Better for Parsing Heterogeneous Dependencies?

Meishan Zhang[†], **Wanxiang Che**[†], **Yanqiu Shao**[‡] , **Ting Liu**[†*]
[†]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{mszhang, car, tliu}@ir.hit.edu.cn
[‡]Beijing Language and Culture University
yqshao163@163.com

Abstract

For languages such as English, several constituent-to-dependency conversion schemes are proposed to construct corpora for dependency parsing. It is hard to determine which scheme is better because they reflect different views of dependency analysis. We usually obtain dependency parsers of different schemes by training with the specific corpus separately. It neglects the correlations between these schemes, which can potentially benefit the parsers. In this paper, we study how these correlations influence final dependency parsing performances, by proposing a joint model which can make full use of the correlations between heterogeneous dependencies, and finally we can answer the following question: parsing heterogeneous dependencies jointly or separately, which is better? We conduct experiments with two different schemes on the Penn Treebank and the Chinese Penn Treebank respectively, arriving at the same conclusion that jointly parsing heterogeneous dependencies can give improved performances for both schemes over the individual models.

1 Introduction

Dependency parsing has been intensively studied in recent years (McDonald et al., 2005; Nivre, 2008; Zhang and Clark, 2008; Huang et al., 2009; Koo and Collins, 2010; Zhang and Nivre, 2011; Sartorio et al., 2013; Choi and McCallum, 2013; Martins et al., 2013). Widely-used corpus for training a dependency parser is usually constructed according to a specific constituent-to-dependency conversion scheme. Several conversion schemes for certain languages have been available. For example, the English language has at least four schemes based on the Penn Treebank (PTB), including the `Yamada` scheme (Yamada and Matsumoto, 2003), the `CoNLL 2007` scheme (Nilsson et al., 2007), the `Stanford` scheme (de Marneffe and Manning, 2008) and the `LTH` scheme (Johansson and Nugues, 2007). There are different conversion schemes for the Chinese Penn Treebank (CTB) as well, including the `Zhang` scheme (Zhang and Clark, 2008) and the `Stanford` scheme (de Marneffe and Manning, 2008). It is hard to judge which scheme is more superior, because each scheme reflects a specific view of dependency analysis, and also there is another fact that different natural language processing (NLP) applications can prefer different conversion schemes (Elming et al., 2013).

Traditionally, we get dependency parsers of different schemes by training with the specific corpus separately. The method neglects the correlations between these schemes, which can potentially help different dependency parsers. On the one hand, there are many consistent dependencies across heterogeneous dependency trees. Some dependency structures remain constant in different conversion schemes. Taking the `Yamada` and the `Stanford` schemes as an example, overall 70.27% of the dependencies are identical (ignoring the dependency labels), according to our experimental analysis. We show a concrete example for the two heterogeneous dependency trees in Figure 1, where six of the twelve dependencies are consistent in the two dependency trees (shown by the solid arcs).

On the other hand, differences between heterogeneous dependencies can possibly boost the evidences of the consistent dependencies. For example in Figure 1, the dependencies "do$\overset{\text{VC}}{\frown}$think"

*Corresponding author.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 530–540, Dublin, Ireland, August 23-29 2014.

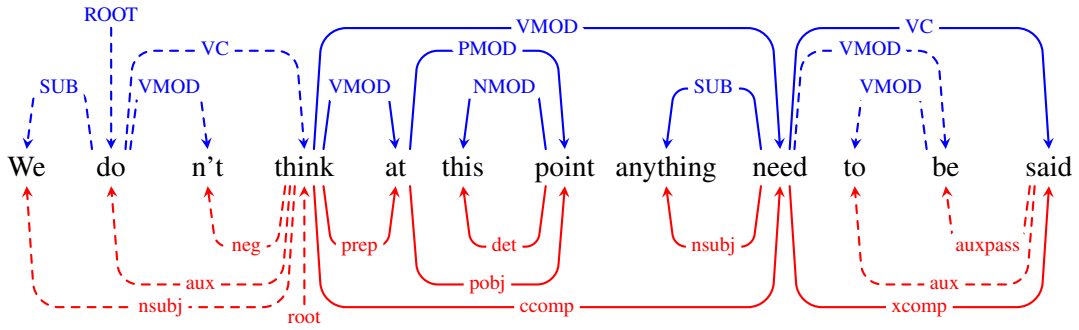


Figure 1: An example to show the differences and similarities of two dependency schemes. The above dependency tree is based on the `Yamada` scheme, while the below dependency tree is based on the `Stanford` scheme. The solid arcs show the consistent dependencies between the two dependency trees, while the dashed arcs show the differences between the two trees.

and "We$\overset{\text{nsubj}}{\curvearrowright}$think" from the two trees can both be potential evidences to support the dependency "think$\curvearrowright$at". Another example, the label "PMOD" from the `Yamada` scheme and the label "pobj" from the `Stanford` scheme on a same dependency "at$\curvearrowright$point" can make it more reliable than one alone.

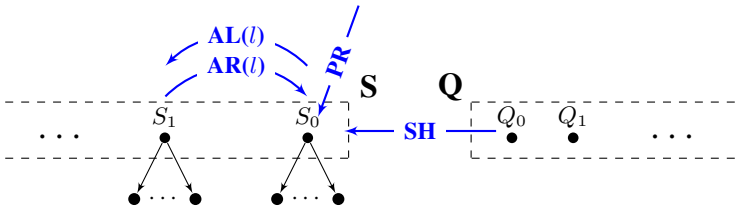
In this paper, we investigate the influences of the correlations between different dependency schemes on parsing performances. We propose a joint model to parse heterogeneous dependencies from two schemes simultaneously, so that the correlations can be fully used by their interactions in a single model. Joint models have been widely studied to enhance multiple tasks in NLP community, including joint word segmentation and POS-tagging (Jiang et al., 2008; Kruengkrai et al., 2009; Zhang and Clark, 2010), joint POS-tagging and dependency parsing (Li et al., 2011; Hatori et al., 2011), and the joint word segmentation, POS-tagging and dependency parsing (Hatori et al., 2012). These models are proposed over pipelined tasks. We apply the joint model into parallel tasks, and parse heterogeneous dependencies together. To our knowledge, we are the first work to investigate joint models on parallel tasks.

We exploit a transition-based framework with global learning and beam-search decoding to implement the joint model (Zhang and Clark, 2011). The joint model is extended from a state-of-the-art transition-based dependency parsing model. We conduct experiments on PTB with the `Yamada` and the `Stanford` schemes, and also on CTB 5.1 with the `Zhang` and the `Stanford` schemes. The results show that our joint model gives improved performances over the individual baseline models for both schemes on both English and Chinese languages, demonstrating positive effects of the correlations between the two schemes. We make the source code freely available at `http://sourceforge.net/projects/zpar/,version0.7`.

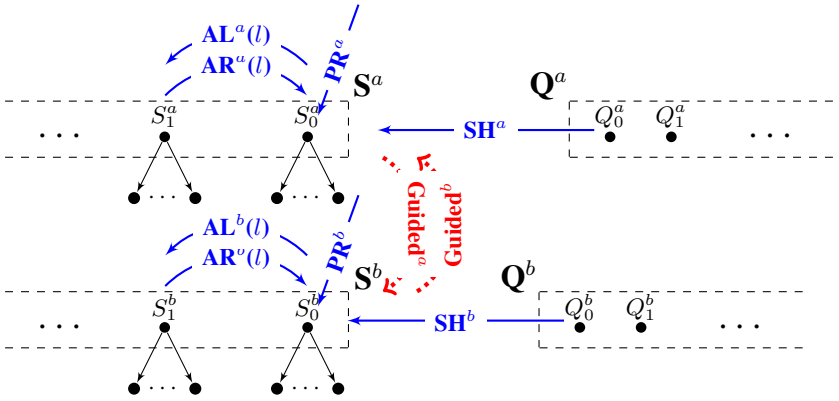
2 Baseline

Traditionally, the dependency parsers of different schemes are trained with their corpus separately, using a state-of-the-art dependency parsing algorithm (Zhang and Clark, 2008; Huang et al., 2009; Koo and Collins, 2010; Zhang and McDonald, 2012; Choi and McCallum, 2013). In this work, we exploit a transition-based arc-standard dependency parsing model combined with global learning and beam-search decoding as the baseline. which is initially proposed by Huang et al. (2009). In the following, we give a detailed description of the model.

In a typical transition-based system for dependency parsing, we define a transition state, which consists of a stack to save partial-parsed trees and a queue to save unprocessed words. The parsing is performed incrementally via a set of transition actions. The transition actions are used to change contents of the stack and the queue in a transition state. Initially, a start state has an empty stack and all words of a sentence in its queue. Then transition actions are applied to the start state, and change states step by step. Finally, we arrive at an end state with only one parsed tree on the stack and no words in the queue. We score each state by its features generated from the historical actions.



(a) Arc-standard dependency parsing model for a single dependency tree



(b) The joint model based on arc-standard dependency parsing for two dependency trees

Figure 2: Illustrations for the baseline dependency parsing model and our proposed joint model.

In the baseline arc-standard transition system, we define four kinds of actions, as shown in Figure 2(a). They are *shift* (SH), *arc-left with dependency label* l (AL(l)), *arc-right with dependency label* l (AR(l)) and *pop-root* (PR), respectively. The *shift* action shifts the first element Q_0 of the queue onto the stack; the action *arc-left with dependency label* l builds a left arc between the top element S_0 and the second top element S_1 on the stack, with the dependency label being specified by l; the action *arc-right with dependency label* l builds a right arc between the top element S_0 and the second top element S_1 on the stack, with the dependency label being specified by l; and the *pop-root* action defines the root node of a dependency tree when there is only one element on the stack and no element in the queue.

During decoding, each state may have several actions. We employ a fixed beam to reduce the search space. The low-score states are pruned from the beam when it is full. The feature templates in our baseline are shown by Table 1, referring to *baseline feature templates*. We learn the feature weights by the averaged percepron algorithm with early-update (Collins and Roark, 2004; Zhang and Clark, 2011).

3 The Proposed Joint Model

The aforementioned baseline model can only handle a single dependency tree. In order to parse multiple dependency trees for a sentence, we usually use individual dependency parsers. This method is not able to exploit the correlations across different dependency schemes. The joint model to parse multiple dependency trees with a single model is an elegant way to exploit these correlations fully. Inspired by this, we make a novel extension to the baseline arc-standard transition system, arriving at a joint model to parse two heterogeneous dependency trees for a sentence simultaneously.

In the new transition system, we double the original transition state of one stack and one queue into two stacks and two queues, as shown by Figure 2(b). We use stacks S^a and S^b and queues Q^a and Q^b to save partial-parsed dependency trees and unprocessed words for two schemes a and b, respectively. Similarly, the transition actions are doubled as well. We have eight transition actions, where four of them are aimed for scheme a, and the other four are aimed for scheme b. The concrete action definitions are similar to the original actions, except an additional constraint that actions should be operated over the corresponding stack and queue of scheme a or b.

We assume that the actions to build a specific tree of scheme a are $A_1^a A_2^a \cdots A_n^a$, and the actions to

Baseline feature templates
Unigram features
S_0w S_0t S_0wt S_1w S_1t S_1wt N_0w N_0t N_0wt N_1w N_1t N_1wt
Bigram features
$S_0w{\cdot}S_1w$ $S_0w{\cdot}S_1t$ $S_0t{\cdot}S_1w$ $S_0t{\cdot}S_1t$ $S_0w{\cdot}N_0w$ $S_0w{\cdot}N_0t$ $S_0t{\cdot}N_0w$ $S_0t{\cdot}N_0t$
Second-order features
$S_{0l}w$ $S_{0r}w$ $S_{0l}t$ $S_{0r}t$ $S_{0l}l$ $S_{0r}l$ $S_{1l}w$ $S_{1r}w$ $S_{1l}t$ $S_{1r}t$ $S_{1l}l$ $S_{1r}l$ $S_{0l2}w$ $S_{0r2}w$ $S_{0l2}t$ $S_{0r2}t$ $S_{0l2}l2$ $S_{0r2}l2$ $S_{1l2}w$ $S_{1r2}w$ $S_{1l2}t$ $S_{1r2}t$ $S_{1l2}l2$ $S_{1r2}l2$
Third-order features
$S_0t{\cdot}S_{0l}t{\cdot}S_{0l2}t$ $S_0t{\cdot}S_{0r}t{\cdot}S_{0r2}t$ $S_1t{\cdot}S_{1l}t{\cdot}S_{1l2}t$ $S_1t{\cdot}S_{1r}t{\cdot}S_{1r2}t$ $S_0t{\cdot}S_1t{\cdot}S_{0l}t$ $S_0t{\cdot}S_1t{\cdot}S_{0l2}t$ $S_0t{\cdot}S_1t{\cdot}S_{0r}t$ $S_0t{\cdot}S_1t{\cdot}S_{0r2}t$ $S_0t{\cdot}S_1t{\cdot}S_{1l}t$ $S_0t{\cdot}S_1t{\cdot}S_{1l2}t$ $S_0t{\cdot}S_1t{\cdot}S_{1r}t$ $S_0t{\cdot}S_1t{\cdot}S_{1r2}t$
Valancy features
S_0wv_l S_0tv_l S_0wv_r S_0tv_r S_1wv_l S_1tv_l S_1wv_r S_1tv_r
Label set features
S_0ws_r S_0ts_r S_0ws_l S_0ts_l S_1ws_l S_1ts_l
Proposed new feature templates for the joint model
Guided head features
$S_0w{\cdot}h_{guide}$ $S_0t{\cdot}h_{guide}$ $S_0wt{\cdot}h_{guide}$ $S_1w{\cdot}h_{guide}$ $S_1t{\cdot}h_{guide}$ h_{guide}
Guided label features
$S_0w{\cdot}S_0l_{guide}$ $S_0t{\cdot}S_0l_{guide}$ $S_0wt{\cdot}S_0l_{guide}$ $S_1w{\cdot}S_0l_{guide}$ $S_1t{\cdot}S_0l_{guide}$ S_0l_{guide} $S_0w{\cdot}S_1l_{guide}$ $S_0t{\cdot}S_1l_{guide}$ $S_0wt{\cdot}S_1l_{guide}$ $S_1w{\cdot}S_1l_{guide}$ $S_1t{\cdot}S_1l_{guide}$ S_1l_{guide}

Table 1: Feature templates for the baseline and joint models, where w denotes the word; t denotes the POS tag; v_l and v_r denote the left and right valencies; l denotes the dependency label; s_l and s_r denotes the label sets of the left and right children; the subscripts l and r denote the left-most and the right-most children, respectively; the subscripts $l2$ and $r2$ denote the second left-most and the second right-most children, respectively; h_{guide} denotes the head direction of the top two elements on the processing stack in the other tree; l_{guide} denotes the label of the same word in the other tree.

build a specific tree of scheme b for the same sentence are $A_1^bA_2^b\cdots A_n^b$. We use $\mathrm{ST}_0^a\mathrm{ST}_1^a\cdots\mathrm{ST}_n^a$ and $\mathrm{ST}_0^b\mathrm{ST}_1^b\cdots\mathrm{ST}_n^b$ to denote the historical states for the two action sequences, respectively. A sequence of actions should consist of $A_1^aA_2^a\cdots A_n^a$ and $A_1^bA_2^b\cdots A_n^b$ in a joint model. However, one question that needs to be answered is that, for a joint state $(\mathrm{ST}_i^a, \mathrm{ST}_j^b)$, which action should be chosen as the next step to merge the two action sequences into one sequence, A_{i+1}^a or A_{j+1}^b? To resolve the problem, we employ a parameter t to limit the next action in the joint model. When t is above zero, an action for scheme b can be applied only if the last action of scheme a is t steps in advance. For example, the action sequence is $A_1^aA_1^bA_2^aA_2^b\cdots A_n^aA_n^b$ when $t=1$. t can be negative as well, denoting the reverse constraints.

In the joint model, we extract features separately for the two dependency schemes. When the next action is aimed for scheme a, we will extract features from S^a and Q^a, according to *baseline feature templates* in Table 1. In order to make use of the correlations between the two dependency parsing trees, we introduce several new feature templates, shown in Table 1 referring to *proposed new feature templates for the joint model*. The new features are based on two kinds of atomic features: the guided head h_{guide} and the guided dependency label l_{guide}. Assuming that the currently processing scheme is a, when the top two elements (S_0^a and S_1^a) have both found their heads in Guided^b (the partial-parsed trees of scheme b), we can fire the atomic feature h_{guide}, which denotes the arc direction between S_0 and S_1 in Guide^b ($S_0^{\curvearrowleft}S_1$, $S_0^{\curvearrowright}S_1$ or other). When S_0^a or S_1^a has its dependency label in Guided^b, we can fire the atomic feature l_{guide}, which denotes the dependency label of S_0^a or S_1^a in Guided^b. Similarly we can extract the h_{guide} and l_{guide} from Guide^a when we are processing scheme b. When t is infinite, we always have

the two atomic features, because the other tree is already parsed. Thus the proposed new features can be the most effective when $t = \infty$ and $t = -\infty$. In other conditions, the other tree may not be ready for the new feature extracting. Similar to the baseline model, we use the beam-search decoding strategy to reduce the search space, and use the averaged perceptron with early-update to learn the feature weights.

We are especially interested in two cases of the joint models when t is infinite ($t = \infty$ and $t = -\infty$), where the tree of one specified scheme is always processed after the other tree is finished, because the new features can be most effectively exploited according to the above analysis. We assume that the first and second processing schemes are s_1 and s_2 respectively, to facilitate the below descriptions. We can see that the joint model behaves similarly to a pipeline reranking model, in optimizing scheme s_1's parsing performances. First we get K-best (K equals the beam size of the joint model) candidates for scheme s_1, and then employ additional evidences from scheme s_2's result, to rerank the K-best candidates, obtaining a better result. The joint model also behaves similarly to a pipeline feature-based stacking model (Li et al., 2012), in optimizing scheme s_2's parsing performances. After acquiring the best result of scheme s_1, we can use it to generate guided features to parse dependencies of scheme s_2. Thus additional information from scheme s_1 can be imported into the parsing model of scheme s_2. Different with the pipeline reranking and the feature-based stacking models, we employ a single model to achieve the two goals, making the interactions between the two schemes be better performed.

4 Experiments

4.1 Experimental Settings

In order to evaluate the baseline and joint models, we conduct experiments on English and Chinese data. For English, we obtain heterogeneous dependencies by the `Yamada` and the `Stanford` schemes, respectively. We transform the bracket constituent trees of English sentences into the `Yamada` dependencies with the Penn2Malt tool,[1] and into the `Stanford` dependencies with the Stanford parser version 3.3.1.[2] Following the standard splitting of PTB, we use sections 2-21 as the training data set, section 22 as the development data set, and section 23 as the final test data set. For Chinese, we obtain heterogeneous dependencies by the `Zhang` and the `Stanford` schemes, respectively. The `Zhang` dependencies are obtained by the Penn2Malt tool using the head rules from Zhang and Clark (2008), while the `Stanford` dependencies are obtained by the Stanford parser version 3.3.1 similar to English.

We use predicted POS tags in all the experiments. We utilize a linear-CRF POS tagger to obtain automatic POS tags for English and Chinese datasets.[3] We use a beam size of 64 to train dependency parsing models. We train the joint models with the `Yamada` or `Zhang` dependencies being handled on stack S^a and queue Q^a, and the `Stanford` dependencies being handled on stack S^b and queue Q^b, referring to Section 3. We follow the standard measures of dependency parsing to evaluate the baseline and joint models, including unlabeled attachment score (UAS), labeled attachment score (LAS) and complete match (CM). We ignore the punctuation words for all these measures.

4.2 Development Results

4.2.1 Baseline

Table 2 at the subtable "Baseline" shows the baseline results on the development data set. The performances of the `Yamada` scheme are better than those of the `Stanford` scheme. The UAS and LAS of the `Yamada` scheme are 92.83 and 91.73 respectively, while they are 92.85 and 90.49 for the `Stanford` scheme respectively. The results demonstrate that parsing the `Stanford` dependencies is more difficult than parsing the `Yamada` dependencies because of the lower performances of the `Stanford` scheme.

[1] `http://stp.lingfil.uu.se/~nivre/research/Penn2Malt.html`.

[2] The tool is available on `http://nlp.stanford.edu/software/lex-parser.shtml`. We use three options to perform the conversion: "-basic" and "-keepPunct", respectively.

[3] The tagging accuracies are 97.30% on the English test dataset and 93.68% on the Chinese test dataset. We thank Hao Zhang for sharing the data used in Martins et al. (2013) and Zhang et al. (2013a).

Model	Yamada			Stanford		
	UAS	LAS	CM	UAS	LAS	CM
Baseline	92.83	91.73	47.35	92.85	90.49	50.06
The joint models, where the Yamada dependencies are processed with priority						
$t = 1$	92.65	91.55	46.35	93.11	90.75	50.24
$t = 2$	92.65	91.57	46.71	93.15	90.77	50.59
$t = 3$	92.82	91.74	47.12	93.19	90.82	50.76
$t = 4$	92.89	91.78	47.35	93.27	90.93	51.29
$t = \infty$	**93.04**	**92.01**	**48.65**	**93.52**	**91.15**	**52.59**
The joint models, where the Stanford dependencies are processed with priority						
$t = -1$	92.62	91.54	46.71	93.10	90.70	50.76
$t = -2$	92.50	91.41	46.18	93.06	90.74	51.12
$t = -3$	92.57	91.42	47.00	93.10	90.68	51.35
$t = -4$	92.74	91.60	47.41	93.15	90.72	51.29
$t = -\infty$	**93.04**	91.95	47.88	93.19	90.91	50.71

Table 2: The main results on the development data set of the baseline and proposed joint models.

4.2.2 Parameter Tuning

The proposed joint model has one parameter t to adjust. The parameter t is used to control the decoding in a joint model, determining which kind of dependencies should be processed at the next step. In our joint model, if t is larger than zero, scheme a (the Yamada scheme) should be handled t steps in advance, while when t is smaller than zero, scheme b (the Stanford scheme) should be handled in advance. When the value of t is infinite, the dependency tree of one scheme is handled until the dependency tree of the other scheme is finished for a sentence.

As shown by Table 2, we have two major findings. First, the joint models are slightly better when t is above zero, by decoding with the Yamada scheme in advance. The phenomenon demonstrates that the decoding sequence is important in the joint parsing models. Second, no matter when t is above or below zero, the performances arrive at the peak when t is infinite. One benefit of the joint models is that we can use the correlations between different dependency trees, through the new features proposed by us. The new features can be the most effective when t is infinite according to the analysis Section 3. Thus this finding indicates that the new features are crucial in the joint models, since the ineffective utilization would decrease the model performances a lot. Actually, when the absolute value of t is small, the features can sometimes be fired and in some other times are not able to be fired, making the training insufficient and also inconsistent for certain word-pair dependencies when their distances can differ (when $t = 1$ for example, the joint model can fire the new features only if the dependency distance equals 1). This would make the final model deficient, and can even hurt performances of the Yamada scheme.

According to the results on the development data set, we use the $t = \infty$ for the final joint model, which first finishes the Yamada tree and then the Stanford tree for each sentence. Our final model achieves increases of 0.21 on UAS and 0.28 on LAS for the Yamada scheme, and increases 0.67 on UAS and 0.66 on LAS for the Stanford scheme.

4.2.3 Feature Ablation

In order to test the effectiveness of the proposed new features, we conduct a feature ablation experiment. Table 3 shows the results, where the mark "/wo" denotes the model without the new features proposed by us. For the Yamada scheme, losses of 0.15 on UAS and 0.21 on LAS are shown without the new features. While for Stanford scheme, larger decreases are shown by 0.57 on UAS and 0.58 on LAS, respectively. The results demonstrate the new features are effective in the joint model.

Model	Yamada			Stanford		
	UAS	LAS	CM	UAS	LAS	CM
Our joint model	93.04	92.01	48.65	93.52	91.15	52.59
Our joint model/wo	92.89	91.80	48.25	92.95	90.57	50.62
Δ	-0.15	-0.21	-0.40	-0.57	-0.58	-1.97

Table 3: Feature ablation results.

Model	Yamada			Stanford		
	UAS	LAS	CM	UAS	LAS	CM
Baseline	92.71	91.67	47.48	92.72	90.61	47.76
Our joint model	**92.89**	**91.86**	**48.39**	**93.30**‡	**91.19**‡	**50.37**
Zhang and Nivre (2011)	92.9	91.8	48.0	–	–	–
Rush and Petrov (2012)	–	–	–	92.7*	–	–
Martins et al. (2013)	93.07	–	–	92.82*	–	–
Zhang et al. (2013a)	93.50	92.41	–	93.64*	91.28*	–
Zhang and McDonald (2014)	93.57	92.48	–	93.71*/93.01**	91.37*/90.64**	–
Kong and Smith (2014)	–	–	–	92.20**	89.67**	–

Table 4: The final results on the test data set, where the results with mark ‡ demonstrates that the p-value is below 10^{-3} using t-test. Our Stanford dependencies are slightly different with previous works, where the results with mark * show the numbers for the Stanford dependencies from Stanford parser version 2.0.5 and the results with mark ** show the numbers for the Stanford dependencies from Stanford parser version 3.3.0.

4.3 Final Results

Table 4 shows our final results on the English test dataset. The final joint model achieves better performances than the baseline models for both the `Yamada` and the `Stanford` schemes, by increases of 0.18 on UAS and 0.19 on LAS for the `Yamada` scheme, and increases of 0.58 on UAS and 0.58 on LAS for the `Stanford` scheme. The results demonstrate that the interactions between the two dependency schemes are useful, and the joint model is superior to separately trained models in handling heterogeneous dependencies.

We compare our results with some representative previous work of dependency parsing as well. Zhang and Nivre (2011) is a feature-rich transition-based dependency parser using the arc-eager transition system. Rush and Petrov (2012), Zhang et al. (2013a) and Zhang and McDonald (2014) are state-of-the-art graph-based dependency parsers. Martins et al. (2013) and Kong and Smith (2014) report their results with the full TurboParser. TurboParser is also a graph-based dependency parser but its decoding algorithm has major differences with the general MST-style decoding.

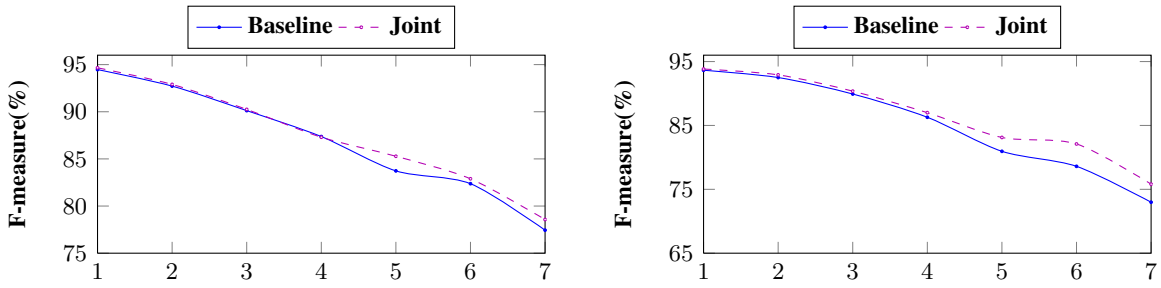
4.4 Analysis

To better understand the joint model, we conduct analysis work on the Chinese development dataset. First, we make a comparison to see whether the consistent dependencies give larger increases by the joint model. As mentioned before, the consistent dependencies can be supported by different evidences from heterogeneous dependencies. We compute the proportion of the consistent dependencies (ignoring the dependency labels) between the `Yamada` and the `Stanford` dependencies, finding that 70.27% of the overall dependencies are consistent. Table 5 shows the comparison results. The joint model shows improvements for the consistent dependencies. However, it does not always show positive effectiveness for the inconsistent dependencies. The results support our initial motivation that consistent dependencies can benefit much in joint models .

We also make a comparison between the baseline and joint models with respect to dependency distance. We use the F-measure value to evaluate the performances. The dependency distances are normal-

	Yamada				Stanford			
	Consistent		Inconsistent		Consistent		Inconsistent	
	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
Baseline	93.43	92.39	91.44	90.17	93.74	91.35	90.75	88.47
Our joint model	93.81	92.85	91.21	90.02	94.58	92.15	91.01	88.78
Δ	**+0.38**	**+0.46**	-0.23	-0.15	**+0.84**	**+0.80**	+0.36	+0.31

Table 5: Performances of the baseline and joint models by whether the dependencies are consistent across the Yamada and the Stanford schemes, where the bold numbers denote the larger increases by comparisons of consistent and inconsistent dependencies for each scheme.



(a) Yamada (b) Stanford

Figure 3: F-measures of the two heterogeneous dependencies with respect to dependency distance.

ized to a max value of 7. Figure 3 shows the comparison results. We find that the joint model can achieve consistent better performances for the dependencies of different dependency distance, demonstrating the robustness of the joint model in improving parsing performances. The joint model performs slightly better for long-distance dependencies, which is more obvious for the Stanford scheme.

4.5 Parsing Heterogeneous Chinese Dependencies

Table 6 shows our final results on the Chinese test data set. For Chinese, the joint model achieves better performances with Stanford dependencies being parsed first. The final joint model achieves better performances than the baseline models for both the Zhang and the Stanford schemes, by increases of 1.13 on UAS and 0.99 on LAS for the Zhang scheme, and increases of 0.30 on UAS and 0.36 on LAS for the Stanford scheme. The results also demonstrate similar conclusions with the experiments on English dataset.

5 Related Work

Our work is mainly inspired by the work of joint models. There are a number of successful studies on joint modeling pipelined tasks where one task is a prerequisite step of another task, for example, the joint model of word segmentation and POS-tagging (Jiang et al., 2008; Kruengkrai et al., 2009; Zhang and Clark, 2010), the joint model of POS-tagging and parsing (Li et al., 2011; Hatori et al., 2011; Bohnet and Nivre, 2012), the joint model of word segmentation, POS-tagging and parsing (Hatori et

Model	Zhang			Stanford		
	UAS	LAS	CM	UAS	LAS	CM
Baseline	79.07	76.08	27.96	80.33	75.29	31.14
Our joint model	**80.20**‡	**77.07**‡	**30.10**	**80.63**	**75.65**	**31.20**

Table 6: The final results on the test data set, where the results with mark ‡ demonstrates that the p-value is below 10^{-3} using t-test.

al., 2012; Zhang et al., 2013b; Zhang et al., 2014), and the joint model of morphological and syntactic analysis tasks (Bohnet et al., 2013). In our work, we propose a joint model on parallel tasks, to parse two heterogeneous dependency trees simultaneously.

There has been a line of work on exploiting multiple treebanks with heterogeneous dependencies to enhance dependency parsing. Li et al. (2012) proposed a feature-based stacking model to enhance a specific target dependency parser with the help of another treebank. Zhou and Zhao (2013) presented a joint inference framework to combine the parsing results based on two different treebanks. All these work are case studies of annotation adaptation from different sources, which have been done for Chinese word segmentation and POS-tagging as well (Jiang et al., 2009; Sun and Wan, 2012). In contrast to their work, we study the heterogeneous annotations derived from the same source. We use a unified model to parsing heterogeneous dependencies together.

Our joint parsing model exploits a transition-based framework with global learning and beam-search decoding (Zhang and Clark, 2011), extended from a arc-standard transition-based parsing model (Huang et al., 2009). The transition-based framework is easily adapted to a number of joint models, including joint word segmentation and POS-tagging (Zhang and Clark, 2010), the joint POS-tagging and parsing (Hatori et al., 2012; Bohnet and Nivre, 2012), and also joint word segmentation, POS-tagging and parsing (Hatori et al., 2012; Zhang et al., 2013b; Zhang et al., 2014).

6 Conclusions

We studied the effectiveness of the correlations between different constituent-to-dependency schemes for dependency parsing, by exploiting these information with a joint model to parse two heterogeneous dependency trees simultaneously. We make a novel extension to a transition-based arc-standard dependency parsing algorithm for the joint model. We evaluate our baseline and joint models on both English and Chinese datasets, based on the `Yamada/Zhang` and the `Stanford` dependency schemes. Final results demonstrate that the joint model which handles two heterogeneous dependencies can give improved performances for dependencies of both schemes. The source code for the joint model is publicly available at `http://sourceforge.net/projects/zpar/,version0.7`.

Acknowledgments

We thank Yue Zhang and the anonymous reviewers for their constructive comments, and gratefully acknowledge the support of the National Basic Research Program (973 Program) of China via Grant 2014CB340503, the National Natural Science Foundation of China (NSFC) via Grant 61133012, 61170144 and 61370164.

References

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the EMNLP-CONLL*, pages 1455–1465, Jeju Island, Korea, July.

Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas Filip Ginter, and Jan Hajic. 2013. Joint morphological and syntactic analysis for richly inflected languages. *TACL*, 1.

Jinho D. Choi and Andrew McCallum. 2013. Transition-based dependency parsing with selectional branching. In *Proceedings of ACL*, pages 1052–1062, August.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the ACL*, pages 111–118, Barcelona, Spain, July.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Manchester, UK, August.

Jakob Elming, Anders Johannsen, Sigrid Klerke, Emanuele Lapponi, Hector Martinez Alonso, and Anders Søgaard. 2013. Down-stream effects of tree-to-dependency conversions. In *Proceedings of the NAACL*, pages 617–626, Atlanta, Georgia, June.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental joint POS tagging and dependency parsing in Chinese. In *Proceedings of 5th IJCNLP*, pages 1216–1224, Chiang Mai, Thailand, November.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2012. Incremental joint approach to word segmentation, POS tagging, and dependency parsing in Chinese. In *Proceedings of the 50th ACL*, pages 1045–1053, Jeju Island, Korea, July.

Liang Huang, Wenbin Jiang, and Qun Liu. 2009. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proceedings of the EMNLP*, pages 1222–1231.

Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lü. 2008. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL-08*, pages 897–904, Columbus, Ohio, June.

Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic adaptation of annotation standards: Chinese word segmentation and POS tagging: a case study. In *Proceedings of the ACL-IJCNLP*, pages 522–530.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for english. In *Proceedings of NODALIDA 2007*, Tartu, Estonia.

Lingpeng Kong and Noah A Smith. 2014. An empirical comparison of parsing methods for stanford dependencies. *arXiv preprint arXiv:1404.4314*.

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the ACL*, pages 1–11.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging. In *Proceedings of the ACL-IJCNLP*, pages 513–521, Suntec, Singapore, August.

Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011. Joint models for Chinese POS tagging and dependency parsing. In *Proceedings of the EMNLP*, pages 1180–1191, Edinburgh, Scotland, UK., July.

Zhenghua Li, Ting Liu, and Wanxiang Che. 2012. Exploiting multiple treebanks for parsing with quasi-synchronous grammars. In *Proceedings of the 50th ACL*, pages 675–684, Jeju Island, Korea, July.

Andre Martins, Miguel Almeida, and Noah A. Smith. 2013. Turning on the turbo: Fast third-order non-projective turbo parsers. In *Proceedings of the 51st ACL*, pages 617–622, Sofia, Bulgaria, August. Association for Computational Linguistics.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*, number June, pages 91–98, Morristown, NJ, USA.

Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 915–932.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

Alexander M Rush and Slav Petrov. 2012. Vine pruning for efficient multi-pass dependency parsing. In *Proceedings of the NAACL*, pages 498–507.

Francesco Sartorio, Giorgio Satta, and Joakim Nivre. 2013. A transition-based dependency parser using a dynamic parsing strategy. In *Proceedings of the 51st ACL*, pages 135–144, Sofia, Bulgaria, August.

Weiwei Sun and Xiaojun Wan. 2012. Reducing approximation and estimation errors for Chinese lexical processing with heterogeneous annotations. In *Proceedings of the 50th ACL*, pages 232–241, Jeju Island, Korea, July.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT*, volume 3.

Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of EMNLP*, pages 562–571, Honolulu, Hawaii, October.

Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *Proceedings of the EMNLP*, pages 843–852, Cambridge, MA, October.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.

Hao Zhang and Ryan McDonald. 2012. Generalized higher-order dependency parsing with cube pruning. In *Proceedings of the EMNLP*, pages 320–331.

Hao Zhang and Ryan McDonald. 2014. Enforcing structural diversity in cube-pruned dependency parsing. In *Proceedings of ACL*. Association for Computational Linguistics.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th ACL*, pages 188–193, Portland, Oregon, USA, June.

Hao Zhang, Liang Huang, Kai Zhao, and Ryan McDonald. 2013a. Online learning for inexact hypergraph search. In *Proceedings of the EMNLP*, pages 908–913, Seattle, Washington, USA, October. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2013b. Chinese parsing exploiting characters. In *Proceedings of the 51st ACL*, pages 125–134, Sofia, Bulgaria, August.

Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2014. Character-level Chinese Dependency Parsing. In *Proceedings of the 52st ACL*.

Guangyou Zhou and Jun Zhao. 2013. Joint inference for heterogeneous dependency parsing. In *Proceedings of the 51st ACL*, pages 104–109, Sofia, Bulgaria, August.

An LR-inspired generalized lexicalized phrase structure parser

Benoit Crabbé
ALPAGE / INRIA - Université Paris Diderot
Place Paul Ricoeur
F-75013 Paris
benoit.crabbe@univ-paris-diderot.fr

Abstract

The paper introduces an LR-based algorithm for efficient phrase structure parsing of morphologically rich languages. The algorithm generalizes lexicalized parsing (Collins, 2003) by allowing a structured representation of the lexical items. Together with a discriminative weighting component (Collins, 2002), we show that this representation allows us to achieve state of the art accurracy results on a morphologically rich language such as French while achieving more efficient parsing times than the state of the art parsers on the French data set. A comparison with English, a lexically poor language, is also provided.

1 Introduction

The paper provides a phrase structure parsing algorithm inspired by LR (Knuth, 1965), GLR (Tomita, 1988) and the recent developments of (Huang and Sagae, 2010) for dependency grammar. The parsing algorithm comes with a discriminative weighting framework inspired by (Collins, 2002). Although discriminative phrase structure parsing has been shown to be challenging when it comes to efficiency issues (Turian and Melamed, 2006; Finkel et al., 2008), we use here several approximations that make the framework not only tractable but also efficient and accurate on a lexically rich language such as French.

Despite the successes of dependency grammar, we are interested in phrase structure grammar since it naturally allows to support compositional semantic representations as recently highlighted by (Socher et al., 2012). It remains that most phrase structure parsers have been designed in priority for modelling lexically poor languages such as English or Chinese (Collins, 2003; Charniak, 2000; Zhu et al., 2013). Although highly accurate multilingual parsers exist (Petrov et al., 2006), they remain relatively both slow for wide coverage purposes and their inner formal structure is not designed to handle naturally morphological information.

We assume that parsing lexically rich languages benefits from taking into account the structured morphological information that can be extracted from lexical forms. Using French as a case study we show that we can reach both parsing efficiency with an approximative inference method and we can get a state of the art accurracy by generalizing lexicalized parsing to handle feature structure-based word representations. Our proposal also differs theoretically from related ones (Sagae and Lavie, 2006; Zhang and Clark, 2011; Zhu et al., 2013) by explicitly using an LR automaton. The explicit introduction of the LR automaton allows us to establish a formal difference between shift reduce phrase structure parsing and shift reduce dependency parsing. It further provides some insights on the nature of the grammar underlying many contemporary parsers.

The paper is organized as follows. First, section 2, we set up a formal framework for describing weighted phrase structure parsing as a 2-LCFG (Nederhof and Satta, 2010). Observing that the tree structures are actually constrained in practice we formulate in section 3 an LR automaton construction method for treebank grammars suitable for encoding these constraints. We then provide in section 4 a description of the algorithm and its components. Section 5 give an extension to 2-LCFG suitable for parsing morphologically rich languages and meeting common practical requirements. The whole

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 541–552, Dublin, Ireland, August 23-29 2014.

framework is then evaluated in section 6 on French and English allowing to better identify its properties with respects to the state of the art.

2 Grammatical representation

The first step we consider is the grammar actually used for parsing and how it is generated. We suppose here a bilexical context free grammar or 2-LCFG (Nederhof and Satta, 2010). A 2-LCFG is a CFG whose rules are of the form given in Figure 1 (left). Symbols of the form x and h denote terminal symbols while symbols of the form $A[h]$ or $A[x]$ denote lexicalized non terminals. A, B, C are non lexicalized non terminals and h denotes a head. A 2-LCFG rule is typically of the form $NP[cat] \rightarrow D[a]\, N[cat]$.

For practical robust parsing, 2-LCFG are grammars with a very large number of rules generated dynamically at runtime (Section 4). Most of the static grammatical preprocessing involved for the generation of an LR automaton only applies to the underlying delexicalized 2-CFG by ignoring lexical annotation symbols.

$$
\begin{aligned}
A[h] &\rightarrow B[h]\, C[x] \\
A[h] &\rightarrow B[x]\, C[h] \\
A[h] &\rightarrow h
\end{aligned}
$$

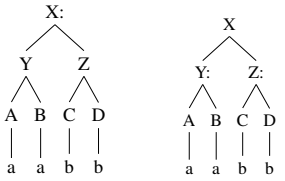


Figure 1: 2-LCFG rule patterns and invalid 2-CFG trees

The first step towards robust parsing thus requires to generate a grammar suitable for this purpose. In our case, the grammar is a treebank grammar and since most treebanks do encode trees with variable branching arities we must transform it to match the 2-LCFG required pattern. The first step amounts to apply an order 0 head markovization (Collins, 2003) which is followed by a reduction of unary rules. Both transformations guarantee that the trees do follow strictly a Chomsky Normal Form (CNF). Trees in CNF have two properties of interest. First, one can show by induction that they can be generated with a constant number of derivation steps η for a sentence of length $n : \eta = 2n-1$. This property is in principle critical for the comparison of weighted parsing hypotheses (Section 5) and explains why we use 2-LCFG as a grammatical representation in the first place. Second, the binarization (markovization) procedure also introduces temporary symbols we consider to be different from other non terminal symbols. These temporary symbols are further constraining the tree structure. Using ':' to denote a temporary symbol in Figure 1 (right), we observe for instance that the root of a tree cannot be temporary and two siblings cannot be temporaries either. By contrast, arc standard dependency parsers such as the one of (Huang and Sagae, 2010) do verify the first property while the second property is irrelevant in that case.

3 LR automaton construction

We use an LR automaton to enforce the parser to generate parse trees satisfying the above mentioned structural constraints. Although, other proposals such as (Sagae and Lavie, 2006) apparently returns a failure when the parser generates invalid trees and (Zhu et al., 2013) apparently handles the problem with local constraints preventing the parser to generate invalid configurations, we use here an $LR(0)$ automaton to ensure that the parser globally enforces these constraints. This seemed to us theoretically justified, easier to generalize (Section 6) and easier to implement.

As such, a traditional $LR(0)$ parser (Knuth, 1965) is not suited for parsing natural language: it aims to statically eliminate ambiguity from the grammar. Here, following (Tomita, 1985) the LR tables are built without trying to resolve conflicts. Instead the conflicts are kept and determinism is brought by a weighting component. The use of $LR(0)$ tables aims to ensure that the parser actually generates valid parse derivations. In this case, the generation of the derivations requires to constrain the underlying 2-CFG grammar with respects to temporary symbols. This being said, building an $LR(0)$ automaton for robust treebank grammars raise two issues. The first is inductive, a grammar read off from a treebank is not guaranteed to be robust and to generalize to other text, since a treebank remains a finite sample of language. The second is practical : traditional $LR(0)$ compilation methods involve the determinisation of

the LR NFA which is exponential in the number n of states of this NFA. In case of very large ambiguous treebank grammars n is very large and the compilation becomes intractable (Briscoe and Carroll, 1993).

These two observations lead us to design this automata by the following construction. First, let Σ be the set of non terminal symbols read off from the treebank, T be the set of temporaries introduced by binarization and N the set of non temporary symbols such that $\Sigma = N \cup T$ and $N \cap T = \emptyset$. Second we note W the set of terminal symbols extracted from the treebank and $A \in N$ the unique axiom of this grammar. We then partition Σ with the following set of equivalence classes: $[a] = \{A\}, [t] = T$ and $[n] = \Sigma - (T \cup A)$. For convenience we also note $[w] = W$. Given these equivalence classes, we define the matrix grammar $G_m = \langle \Sigma_m, [w], [a], R_m \rangle$ (where $\Sigma_m = \{[a], [n], [t]\}$). The rules R_m of G_m are then designed to enforce the above mentioned tree well formedness constraints. Some possible such rules are given in Table 1 using ID/LP notation (Gazdar et al., 1985).In other words, an immediate dominance rule of the form $a \rightarrow b\,,\, c$ is expanded as two rules $a \rightarrow b\,c$ and $a \rightarrow c\,b$. Such a grammar allows

$[a] \rightarrow [n]\,,\, [t]$	$[n] \rightarrow [n]\,,\, [t]$	$[t] \rightarrow [n]\,,\, [t]$
$[a] \rightarrow [n]\,,\, [n]$	$[n] \rightarrow [n]\,,\, [n]$	$[t] \rightarrow [n]\,,\, [n]$
$[a] \rightarrow [w]$	$[n] \rightarrow [w]$	

Table 1: Example of Immediate Dominance rules for G_m

to enforce the above-mentioned constraints, it is also small and it is robust : $L(G_m) = [w]+$. We can then very easily build a deterministic $LR(0)$ automaton $A_m = \langle \Sigma_m \cup \{[w]\}, Q, i, F, E_m \rangle$ with classical methods (Aho et al., 2006). From this automaton we can then efficiently generate an expanded automaton $A_{exp} = \langle \{\Sigma \cup W\}, Q, i, F, E \rangle$ where $E = \{(q, a, q') \mid (q, [x], q') \in E_m, \forall a \in [x]\}$. In order to read off the $LR(0)$ table from A_{exp}, we consider the set of actions $\mathcal{A} \stackrel{def}{=} \{RL(X)|X \in \Sigma\} \cup \{RR(X)|X \in \Sigma\} \cup \{RU(X)|X \in \Sigma\} \cup \{S\}$ first introduced by (Sagae and Lavie, 2006). In short S denotes the shift action, $RU(X)$ denotes an unary reduction by terminal X, $RL(X)$ denotes a binary reduction by terminal X with left symbol marked as head, and $RR(X)$ denotes a binary reduction by terminal X with right symbol marked as head. By contrast with a classical LR action set, we extract the actions $RL(X)$ and $RR(X)$ from a state $q \in Q$ if we have an LR item of the form $\langle X \rightarrow A\,B\bullet \rangle$ without requiring that $\langle X \rightarrow B\,A\bullet \rangle \in q$. This simplification, mirrorring that of (Sagae and Lavie, 2006), reduces the number of actions, eases learning and makes parsing more efficient. This being said, the matrix grammar G_m given in Table 1 is not the only one possible (see also section 6). A valid rule set must enforce tree well formedness constraints by building upon a partition of Σ in equivalence classes. On the other hand the action set $\mathcal{A}$ defined here implies that for every rule $R \in R_m$ of the form $[x] \rightarrow [y]\,[z]$ there is a rule $R' \in R_m$ of the form $[x] \rightarrow [z]\,[y]$. That is why we formulate the rules R_m with ID/LP notation and this also means that we cannot express any word ordering constraint with this grammar. This last property is actually shared by many robust parsers.

4 Discriminative LR-based parsing

The LR tables being built by preserving conflicts, determinism is achieved by a weighting component derived from the global perceptron described by (Collins, 2002). We start by describing the weighted parsing procedure before turning our attention to the weight estimation problem.

We assume that an $LR(0)$ table has been built. The GOTO function of this table GOTO: $(\Sigma \cup W) \times \mathbb{N} \mapsto \mathbb{N}$ sends a couple of symbol and LR state to a new LR state. The ACTION: $(\mathbb{N} \times W) \mapsto 2^{\mathcal{A}}$ function of this table returns a set $\mathbf{a}$ of possible actions given a state and a terminal symbol. The initial LR state of the table is σ_i while σ_e denotes a final state.

The algorithm relies on two data structures: a stack $\mathbf{S}$ and a queue. The stack $\mathbf{S} = \ldots|s_2|s_1|s_0$ has s_0 for topmost element. A node $s_i = \langle \sigma, \tau \rangle$ in the stack is a couple where σ is an LR state number and $\tau = (s_i.c_t[s_i.w_t] \quad s_i.c_l[s_i.w_l] \quad s_i.c_r[s_i.w_r])$ encodes a local tree of depth 1. $s_i.c_t, s_i.c_l, s_i.c_r$ denote the root left child and right child categories of tree and $s_i.w_t, s_i.w_l, s_i.w_r$ denote the root, the left child and right child terminals of this tree such that a node $s_i.c.[s_i.w.]$ denotes a non terminal 2-LCFG symbol at node s_i in the stack. The queue is static and initially filled up with the sequence of tokens to be parsed:

$$\textsc{Item} \quad \langle j, \mathbf{S} \rangle : w$$
$$\textsc{Init} \quad \langle 1, \langle \sigma_i, \epsilon \rangle \rangle : 0$$
$$\textsc{Goal} \quad \langle n+1, \langle \sigma_e, \tau \rangle \rangle : w$$

$$\textsc{Shift} \quad \frac{\langle j, \mathbf{S}_\ominus \mid s_0 = \langle \sigma, \_ \rangle \rangle : w}{\langle j+1, \mathbf{S}_\ominus \mid s_0 \mid \langle \textsc{Goto}(t_j, \sigma), (t_j[t_j] \_\_) \rangle : w + F(S, \langle j, \mathbf{S} \rangle)}$$

$$\text{RL(X)} \quad \frac{\langle j, \mathbf{S}_\ominus \mid s_2 = \langle \sigma_2, \_ \rangle : w_2 \mid s_1 = \langle \sigma_1, (s_1.c_t[s_1.w_t] \_\_) \rangle : w_1 \mid s_0 = \langle \sigma_0, (s_0.c_t[s_0.w_t] \_\_) \rangle \rangle : w_0}{\langle j, \mathbf{S}_\ominus \mid s_2 \mid \langle \textsc{Goto}(X, \sigma_2), (X[s_1.w_t]\ s_1.c_t[s_1.w_t]\ s_0.c_t[s_0.w_t]) \rangle : w_0 + F(RL(X), \langle j, \mathbf{S} \rangle)}$$

$$\text{RR(X)} \quad \frac{\langle j, \mathbf{S}_\ominus \mid s_2 = \langle \sigma_2, \_ \rangle : w_2 \mid s_1 = \langle \sigma_1, (s_1.c_t[s_1.w_t] \_\_) \rangle : w_1 \mid s_0 = \langle \sigma_0, (s_0.c_t[s_0.w_t] \_\_) \rangle \rangle : w_0}{\langle j, \mathbf{S}_\ominus \mid s_2 \mid \langle \textsc{Goto}(X, \sigma_2), (X[s_0.w_t]\ s_1.c_t[s_1.w_t]\ s_0.c_t[s_0.w_t]) \rangle : w_0 + F(RR(X), \langle j, \mathbf{S} \rangle)}$$

$$\text{RU(X)} \quad \frac{\langle j, \mathbf{S}_\ominus \mid s_1 = \langle \sigma_1, (s_1.c_t[s_1.w_t] \_\_) \rangle \mid s_0 = \langle \sigma_0, (s_0.c_t[s_0.w_t] \_\_) \rangle \rangle : w_0}{\langle j, \mathbf{S}_\ominus \mid s_1 \mid \langle \textsc{Goto}(X, \sigma_1), (X[s_0.w_t]\ s_0.c_t[s_0.w_t]) \rangle : w_0 + F(RU(X), \langle j, \mathbf{S} \rangle)}$$

$$\text{GR} \quad \frac{\langle j, \mathbf{S}_\ominus \mid s_1 = \langle \sigma_1, (s_1.c_t[s_1.w_t] \_\_) \rangle \mid s_0 = \langle \sigma_0, (s_0.c_t[s_0.w_t] \_\_) \rangle \rangle : w_0}{\langle j, \mathbf{S}_\ominus \mid s_1 \mid \langle \textsc{Goto}(GR, \sigma_1), (s_0.c_t[s_0.w_t] \_\_) \rangle : w_0 + F(GR, \langle j, \mathbf{S} \rangle)} \qquad \text{(Rule introduced in section 5)}$$

Figure 2: Actions as inference rules in extended deductive notation

$\mathcal{T} = t_1 \dots t_n$. Parsing is performed by generating sequentially configurations $C_i = \langle j, \mathbf{S} \rangle$ where $\mathbf{S}$ is a stack and j the index of the first element of the queue. Given an initial configuration $C_0 = \langle 1, \langle \sigma_i, \epsilon \rangle \rangle$, a derivation step $C_{i-1} \overset{a_{i-1}}{\Rightarrow} C_i$ generates a new configuration $C_i = \langle j', \mathbf{S}' \rangle$ provided a configuration $C_{i-1} = \langle j, \mathbf{S}_\ominus | \langle \sigma, \tau \rangle \rangle$ by applying the action $a_{i-1} \in \textsc{Action}(\sigma, t_j)$. A $k-$step derivation sequence $C_{0 \Rightarrow k}$ is a sequence of derivation steps such that $C_0 \overset{a_0}{\Rightarrow} \dots \overset{a_{k-1}}{\Rightarrow} C_k$. A derivation sequence is finished when the configuration $C_{3n-1} = \langle n+1, \langle \sigma, \tau \rangle \rangle$ is generated[1]. If $\sigma = \sigma_e$ then the derivation is a success, otherwise it is a failure. A derivation is also finished when $\textsc{Action}(\sigma, t_j) = \emptyset$ for a configuration $C_k = \langle \sigma, t_j \rangle$ which is another case of failure. The actions detailed in Figure 2 using extended deductive notation are responsible for modifying the stack and updating LR states. The shift action, SHIFT, thus pushes onto the stack a local tree rooted by the category of the next token in the queue. The reduce left $RL(X)$ and the reduce right $RR(X)$ actions pop the top two elements from the stack and push a new element of category $X[w]$ on top of it. The two actions differ only by the way the head w is assigned: $RL(X)$ chooses w to be X's left child head word while $RR(X)$ sets w to be X's right child head word. $RU(X)$ is an unary reduction action that pops the stack top and pushes a new top element with category X whose head is its unique child head. By design of the automaton we ensure that $RU(X)$ can only be applied after a shift reduction took place.

In order to achieve disambiguation, a derivation sequence $C_{0 \Rightarrow k} = C_0 \overset{a_0}{\Rightarrow} \dots \overset{a_{k-1}}{\Rightarrow} C_k$ is also weighted by a function of the form:

$$W(C_{0 \Rightarrow k}) = \mathbf{w} \cdot \mathbf{\Phi}_g(C_{0 \Rightarrow k}) = \sum_{i=0}^{k-1} \mathbf{w} \cdot \mathbf{\Phi}(a_i, C_i)$$

that is the weight of a derivation sequence is given by an inner product that the parser approximates as a sum of inner products local to each derivation step. $\mathbf{w} \in \mathbb{R}^d$ is a d-dimensional vector of weights and each $\mathbf{\Phi}(a_i, C_i) \in \{0, 1\}^d$ is a d-dimensional vector of feature functions in which every ϕ_i has signature $\phi_i(a, \boldsymbol{\kappa}, j)$. The values a and j denote an action and the current index of the head of the queue in $\mathcal{T}$ while $\boldsymbol{\kappa}$ is a kernel vector similar to the one defined by (Huang and Sagae, 2010). It summarizes information accessible from the stack for the purpose of feature function evaluation. Figure 3 illustrates the actual kernel vector used in this paper: together with j, the index of the first element in the queue, the kernel

[1]For shift reduce parsing with a 2-CFG grammar, the number of steps is the number of reduction steps plus n shifts: $\eta = 2n - 1 + n = 3n - 1$.

vector $\boldsymbol{\kappa}$ is the set of values accessible to feature functions $\phi_i(a, \boldsymbol{\kappa}, j)$ in the stack. As can be seen the stack stores local trees with instanciated 2-LCFG nodes labelled with the notation introduced in section 4. Since the score of a derivation is a sum of independent terms, the (prefix) weight $w = W(C_{0\Rightarrow k})$ of a derivation sequence can be computed at each derivation step. This allows to store the (prefix) weight of this sequence on configurations such that a configuration has the extended form $C_k = \langle j, \mathbf{S}\rangle : w$ in the weighted case. We make explicit the actual prefix weight computation in Figure 2 by using the following abbreviation: $F(a_i, C_i) = \mathbf{w} \cdot \boldsymbol{\Phi}(a_i, C_i)$.

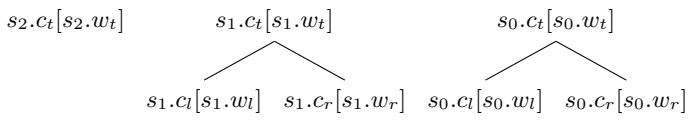


Figure 3: Representation of the kernel vector $\boldsymbol{\kappa}$

For a given input sequence $\mathcal{T}$, the parser is naturally non deterministic. Non determinism is introduced by the ACTION function which returns a set $\mathbf{a} \in 2^{\mathcal{A}}$ of possible actions given the current configuration. In the nondeterministic case, we thus derive from a given derivation sequence $C_{0\Rightarrow k-1}$ a set $\delta(C_{0\Rightarrow k-1})$ of $k-$steps derivation sequences. If we let $\text{GEN}_{k-1}(\mathcal{T})$ be the set of derivation sequences at step $k-1$, the set of derivation sequences at step k is $\text{GEN}_k(\mathcal{T}) = \bigcup_{C_{0\Rightarrow k-1}\in\text{GEN}_{k-1}(\mathcal{T})} \delta(C_{0\Rightarrow k-1})$. In this context, achieving deterministic parsing amounts to solve the following optimization problem:

$$\hat{\mathcal{C}} = \underset{C_{0\Rightarrow 3n-1}\in\text{GEN}_{3n-1}(\mathcal{T})}{\text{argmax}} W(C_{0\Rightarrow 3n-1}) \tag{1}$$

Since in the worst case, the size of $\text{GEN}_k(\mathcal{T})$ is $|\mathcal{A}|^k$, the search space has exponential size. Like (Zhu et al., 2013), we use in this paper a beam search approximation. A beam $\text{GEN}^K_k(\mathcal{T})$ is a subset of size K of $\text{GEN}_k(\mathcal{T})$. Provided a beam $\text{GEN}^K_{k-1}(\mathcal{T})$ we build $\text{GEN}^K_k(\mathcal{T})$ with the following recurrence: $\text{GEN}^K_k(\mathcal{T}) = \text{K-argmax}_{C_{0\Rightarrow k}\in\Delta(\text{GEN}^K_{k-1}(\mathcal{T}))} W(C_{0\Rightarrow k})$ where $\Delta(\text{GEN}^K_{k-1}(\mathcal{T})) = \bigcup_{C_{0\Rightarrow k-1}\in\text{GEN}^K_{k-1}(\mathcal{T})} \delta(C_{0\Rightarrow k-1})$, Using a beam aims to reduce complexity to $\mathcal{O}(K|\mathcal{A}|(3n-1)) \approx \mathcal{O}(n)$ and makes inference computationally tractable in practice. On the other hand it makes inference incomplete (the parser may fail to find a solution even if it exists) and does not guarantee the solution to be optimal. In other words, Equation 1 is replaced by an approximation:

$$\tilde{\mathcal{C}} = \underset{C_{0\Rightarrow 3n-1}\in\text{GEN}^K_{3n-1}(\mathcal{T})}{\text{argmax}} W(C_{0\Rightarrow 3n-1}) \tag{2}$$

The weight estimation procedure is performed by the averaged percetron algorithm (Collins, 2002). As pointed out by (Huang et al., 2012) using a beam introduces an approximation that can also harm the convergence of the learning procedure since we provide at each training iteration the approximative solution given by equation 2 instead of the exact solution to equation 1 expected in theory by the perceptron algorithm. To overcome the problem we perform updates on subderivation sequences. Let $C^{(r)}_{0\Rightarrow k}$ be a subderivation sequence at step k and let $C^{(0)}_{0\Rightarrow k} = \text{argmax}_{C_{0\Rightarrow k}\in\text{GEN}^K_k(\mathcal{T})} W(C_{0\Rightarrow k})$ be the best subderivation in the beam at step k. In this context the perceptron update has the form: $\mathbf{w} \leftarrow \mathbf{w} + \boldsymbol{\Phi}_g(C^{(r)}_{0\Rightarrow k}) - \boldsymbol{\Phi}_g(C^{(0)}_{0\Rightarrow k})$. We tested two methods for choosing k satisfying the weaker convergence criterions established by (Huang et al., 2012) : $C^{(0)}_{0\Rightarrow k} \neq C^{(r)}_{0\Rightarrow k}$ and $W(C^{(0)}_{0\Rightarrow k}) > W(C^{(r)}_{0\Rightarrow k})$. If we let $V = \{k \,|\, C^{(0)}_{0\Rightarrow k} \neq C^{(r)}_{0\Rightarrow k}, W(C^{(0)}_{0\Rightarrow k}) > W(C^{(r)}_{0\Rightarrow k})\}$, then the *early update* method amounts to choose $k = \min_{k\in V} k$ and the *max violation update* method amounts to choose $k = \text{argmax}_{k\in V} W(C^{(0)}_{0\Rightarrow k}) - W(C^{(r)}_{0\Rightarrow k})$.

5 Generalisations

This section introduces two extensions to the algorithm meeting practical motivations: grammar relaxation and extended word representations.

In practical cases, it may be convenient to interface the parser with a morphological tagger. In this case terminal symbols $t_1 \dots t_n$ are part of speech tags. Since grammar transformations introduced in section 2 can potentially modify the tagset and since enforcing a strict Chomsky normal form in this case makes little sense, we allow the trees to have structures such as the one given in Figure 4.

This kind of structure licences the following new patterns of 2-CFG rules: $A \rightarrow B\,t$ and $A \rightarrow t\,B$ where t denotes a terminal symbol (in this case a tag). These new rule patterns modify a property of 2-LCFG on which we relied so far, η is now variable : $n-1 \leq \eta \leq 2n-1$. We observe that longer derivation sequences tend to have an higher weight. Indeed weights increase linearly with the length of the derivation sequence as illustrated in Figure 5 where the weights are averaged out of measurements made over the parses on the French development set described in Section 6.

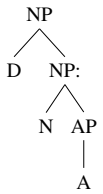


Figure 4: Relaxed tree structure

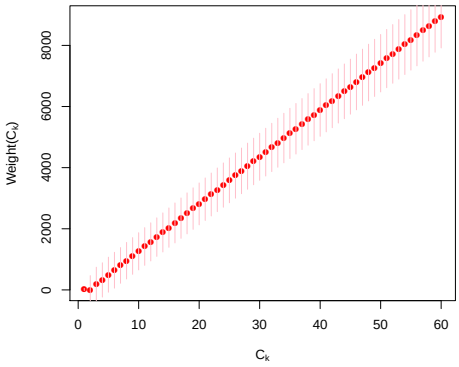


Figure 5: Average derivation weight as a function of the derivation length

Although weights can in principle be positive or negative, this apparently counter-intuitive behaviour is caused by the beam which keeps for further steps only the highest weighted configurations. To further study the behaviour of variable length sequences, we define two variants of the parser: first the 'naive' variant modifies the termination condition. Let $\mathcal{S} = \{C_{0\Rightarrow k} | C_k = \langle n+1, \langle \sigma_e, \tau \rangle \rangle, 2n-1 \leq k \leq 3n-1\}$. In this context, equation 2 is reframed as: $\tilde{\mathcal{C}} = \text{argmax}_{C_{0\Rightarrow k} \in \mathcal{S}}\ W(C_{0\Rightarrow k})$. A second version, called the 'synchronized version' introduces an additional inference rule called the Ghost Reduction and referred as GR in Figure 2. A slight modification of the LR automaton construction, designed to trigger either an unary reduction action or a Ghost reduction after a shift allow us to enforce the property that $\eta = 3n-1$ in this case too. The ghost reduction is designed to both make the parser 'wait' one step during derivation in case it chooses not to perform an unary reduction after shift and also to avoid modifying the content of the stack.

The second extension allows terminals to be not only lexical tokens or part-of-speech tags but arbitrary tuples ω. This allows to encode words with an arbitrary set of additional structured features such as their lemmas, gender, number, case, semantic representation. The exact nature of these additional features depends on the capacity of a parsing preprocessor to actually supply them. In this context the non terminal symbols of the 2-LCFG have thus the form $A[\omega]$. The fields of the tuples are then made accessible to feature functions. This extension is motivated by the hypothesis that parsing morphologically rich languages will benefit significantly from structured word representations, for instance allowing the parser to take advantage of morphology.

We are now in position to describe the feature templates used by the parser (Figure 6). Before the dot s_i and q_i denote respectively the address in the stack and in the queue of the adressed node. t, l, r denote the top, left and right nodes of the local trees in the stack. After the dot w_c, w_f denote a category and a word form, while c is a constituent category. w_m denote the mood of a verb and w_X an refined category dubbed `subcat` in the French Treebank (Abeillé et al., 2003): these subcategories refine crude tags by encoding information such as the definiteness of a determiner, subtypes of adjectives etc. gen, num, agr

$s_0t.w_c \,\&\, s_0t.c$	$s_0t.w_f \,\&\, s_1t.w_f$	$s_0t.c \,\&\, s_1t.c \,\&\, s_2t.c$	$s_0t.c \,\&\, q_2.w_c \,\&\, q_3.w_c$	**Agreement**
$s_0t.w_f \,\&\, s_0t.c$	$s_0t.w_f \,\&\, s_1t.c$	$s_0t.w_f \,\&\, s_1t.c \,\&\, s_2t.c$	$s_0t.c \,\&\, q_2.w_f \,\&\, q_3.w_c$	$s_0tc \,\&\, e(s_0t.agr, s_1t.agr) \,\&\, s_1t.c$
$s_1t.w_c \,\&\, s_1t.c$	$s_0t.c \,\&\, s_1t.w_f$	$s_0t.c \,\&\, s_1t.w_f \,\&\, q_0.w_c$	$s_0t.c \,\&\, q_2.w_c \,\&\, q_3.w_f$	$s_0tc \,\&\, e(s_0t.num, s_1t.num) \,\&\, s_1t.c$
$s_1t.w_f \,\&\, s_1t.c$	$s_0t.c \,\&\, s_1t.c$	$s_0t.c \,\&\, s_1t.c \,\&\, s_2t.w_f$	$s_0t.c \,\&\, s_0r.c \,\&\, s_1t.c$	$s_0tc \,\&\, e(s_0t.gen, s_1t.gen) \,\&\, s_1t.c$
$s_2t.w_c \,\&\, s_2t.c$	$s_0t.w_f \,\&\, q_0.w_f$	$s_0t.c \,\&\, s_1t.c \,\&\, q_0.w_c$	$s_0t.c \,\&\, s_0r.c \,\&\, s_1t.w_f$	$s_0tc \,\&\, e(s_0t.agr, q_0.agr) \,\&\, q_1.w_c$
$s_2t.w_c \,\&\, s_2t.c$	$s_0t.c \,\&\, q_0.w_f$	$s_0t.w_f \,\&\, s_1t.c \,\&\, q_0.w_c$	$s_0t.w \,\&\, s_0r.c \,\&\, s_1t.w_f$	$s_0tc \,\&\, e(s_0t.gen, q_0.gen) \,\&\, q_1.w_c$
$q_0.w_c \,\&\, q_0.w_f$	$s_0t.c \,\&\, q_0.w_c$	$s_0t.c \,\&\, s_1t.w_f \,\&\, q_0.w_c$	$s_0t.c \,\&\, s_0l.w_f \,\&\, s_1t.c$	$s_0tc \,\&\, e(s_0t.num, q_0.num) \,\&\, q_1.w_c$
$q_1.w_c \,\&\, q_1.w_f$	$q_0.w_f \,\&\, q_1.w_f$	$s_0t.c \,\&\, s_1t.c \,\&\, q_0.w_f$	$s_0t.c \,\&\, s_0l.c \,\&\, s_1t.w_f$	$s_0tc \,\&\, e(s_0t.agr, q_1.agr) \,\&\, q_1.w_c$
$q_2.w_c \,\&\, q_2.w_f$	$q_0.w_f \,\&\, q_1.w_c$	$s_0t.c \,\&\, q_0.w_c \,\&\, q_1.w_c$	$s_0t.c \,\&\, s_0l.c \,\&\, s_1t.c$	$s_0tc \,\&\, e(s_0t.num, q_0.num) \,\&\, q_1.w_c$
$q_3.w_c \,\&\, q_3.w_f$	$q_0.w_c \,\&\, q_1.w_c$	$s_0t.c \,\&\, q_0.w_f \,\&\, q_1.w_c$	**Mood**	$s_0tc \,\&\, e(s_0t.gen, q_0.gen) \,\&\, q_1.w_c$
$s_0l.w_f \,\&\, s_0l.c$	$s_1t.w_f \,\&\, q_0.w_f$	$s_0t.c \,\&\, q_0.w_c \,\&\, q_1.w_f$	$s_0t.w_m \,\&\, s_1t.w_f$	**Subcat**
$s_0r.w_f \,\&\, s_0r.c$	$s_1t.w_f \,\&\, q_0.w_c$	$s_0t.c \,\&\, q_1.w_c \,\&\, q_2.w_c$	$s_0t.w_f \,\&\, s_1t.w_m$	$s_0t.w_X \,\&\, s_1t.w_f$
$s_1l.w_f \,\&\, s_1l.c$	$s_1t.c \,\&\, q_0.w_f$	$s_0t.c \,\&\, q_1.w_f \,\&\, q_2.w_c$	$s_0t.c \,\&\, s_1t.w_m$	$s_0t.w_f \,\&\, s_1t.w_X$
$s_1r.w_f \,\&\, s_1r.c$	$s_1t.c \,\&\, q_0.w_c$	$s_0t.c \,\&\, q_1.w_c \,\&\, q_2.w_f$	$s_0t.w_m \,\&\, s_1t.c$	$s_0t.c \,\&\, s_1t.w_X$ $\quad s_0tw_X \,\&\, s_1t.c$

Figure 6: Parser templates

denote the gender, the number and their interaction. The notation $e(\cdot, \cdot)$ is an equality function returning true if the values of both its argument are equal.

6 Experiments

The following experiments aim to identify the contribution of the components of the parser both to parsing accurracy and to parsing speed. Experiments are carried mainly on French. A final set of tests is also carried out on English in order to highlight the generality of the framework and to ease comparisons with other proposals.

6.1 Protocol

The experiments use the French SPMRL dataset (Seddah et al., 2013) which is newer and larger than datasets previously used for parsing French (Crabbé and Candito, 2008). It instanciates the full French Treebank described in (Abeillé et al., 2003) and will surely become the new standard data set for parsing French in the next few years. We use this data set as is, with two scenarios: one with gold standard tags and the second with tags predicted by a 97.35% accurate tagger (Seddah et al., 2013). The French data is head annotated with head rules provided by (Arun and Keller, 2005). Additionally, compound word structures are systematically left headed. For English, we use the Penn Treebank with standard split: section 02-21 for training, section 22 for development and section 23 for test. The predicted scenario uses the MELT tagger (Denis and Sagot, 2012) with an accurracy of 97.1%. The head annotations have been inferred by aligning the phrase structure treebank with its dependency conversion described by (de Marneffe et al., 2006).

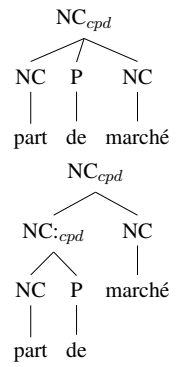
We use a C++ implementation of the algorithm described above for running the experiments. Scores reported for the Berkeley parser (Petrov et al., 2006) use the runs described by (Seddah et al., 2013). F-score is measured with the classical `evalb` and times are measured on the same machine (MacOSX 2.4Ghz) and do not take into account input/output times for both parsers.

Each experiment modifies a single experimental variable by contrast with a default parser configuration. The default parser configuration sets the beam size to $K = 4$ and uses the naive synchronisation procedure (Section 5). The LR automaton uses the grammar $G_m^{(base)}$ (Figure 7) and the update method is *early update* (Section 4). The set of templates is given in Figure 6 except for English where agreement, mood and subcat are ignored since there is no morphology directly available.

Experiment 1 This first experiment tests the impact of the beam size by running the parser with different sizes: $K = 2, K = 4, K = 8, K = 16$.

Experiment 2 The second experiment contrasts the naive synchronisation (*naive*) with the ghost reduction synchronisation (*sync*) described in section 5.

Experiment 3 This third experiment contrasts two different matrix grammars (Figure 7). This experiment aims to test whether we can take advantage of the LR automaton to better account for compound words encoded in the French data. To this end we designed two matrix grammars generating two different automata. In Figure 7, the top left tree is an example of the representation of compound words in the French data set. The corresponding binary structure is given on bottom left. The general grammar $G_m^{(base)}$ encodes a matrix grammar that does not specifically handle compound words and for which equivalence classes are $[a]$ the axiom symbol, $[n]$ non terminal symbols and $[w]$ terminal symbols. Each temporary non terminal $t \in T$ yields its own equivalence class. The grammar $G_m^{(cpd)}$ adds further equivalence classes: $[n]_{cpd}$ gathers non terminals marked as compounds (cpd) and is disjoint from $[n]$, the non compound non terminals. $T_{(cpd)}$ gathers temporary symbols marked as compounds and is disjoint from T. From these two matrix grammar we generate two different LR automata with one of them encoding a specific subgrammar for compounds (*cpd*) while the other is a generic grammar (*base*).



Immediate dominance rules for G_m^{base}		
$[a] \rightarrow [n]\,[n]$	$[n] \rightarrow [n]\,,\,[n]$	$t \rightarrow [n]\,,\,[n] \quad (\forall t \in T)$
$[a] \rightarrow [n]\,,\,[w]$	$[n] \rightarrow [n]\,,\,[w]$	$t \rightarrow [n]\,,\,[w] \quad (\forall t \in T)$
$[a] \rightarrow [w]\,[w]$	$[n] \rightarrow [w]\,,\,[w]$	$t \rightarrow [w]\,,\,[w] \quad (\forall t \in T)$
$[a] \rightarrow [w]$	$[n] \rightarrow [w]$	$t \rightarrow [n]\,,\,t \quad (\forall t \in T)$
$[a] \rightarrow [n]\,,\,t \quad (\forall t \in T)$	$[n] \rightarrow [n]\,,\,t \quad (\forall t \in T)$	$t \rightarrow [w]\,,\,t \quad (\forall t \in T)$
$[a] \rightarrow [w]\,,\,t \quad (\forall t \in T)$	$[n] \rightarrow [w]\,,\,t \quad (\forall t \in T)$	
Additional immediate dominance rules for G_m^{cpd}		
$[a] \rightarrow [n]_{cpd}\,,\,[n]$	$[n] \rightarrow [n]_{cpd}\,,\,[n]$	$t \rightarrow [n]_{cpd}\,,\,[n] \quad (\forall t \in T)$
$[a] \rightarrow [n]_{cpd}\,,\,[n]_{cpd}$	$[n] \rightarrow [n]_{cpd}\,,\,[n]_{cpd}$	$t \rightarrow [n]_{cpd}\,,\,[n]_{cpd} \quad (\forall t \in T)$
$[a] \rightarrow [n]_{cpd}\,,\,[w]$	$[n] \rightarrow [n]_{cpd}\,,\,[w]$	$t \rightarrow [n]_{cpd}\,,\,[w] \quad (\forall t \in T)$
$[a] \rightarrow [n]_{cpd}\,,\,t \quad (\forall t \in T)$	$[n] \rightarrow [n]_{cpd}\,,\,t \quad (\forall t \in T)$	$t \rightarrow [n]_{cpd}\,,\,t \quad (\forall t \in T)$
$[n]_{cpd} \rightarrow [w]\,,\,[w]$	$[n]_{cpd} \rightarrow t_{cpd}\,,\,[w] \quad (\forall t_{cpd} \in T_{cpd})$	$t_{cpd} \rightarrow t_{cpd}\,,\,[w] \quad (\forall t_{cpd} \in T_{cpd})$
$[n]_{cpd} \rightarrow [w]$	$t_{cpd} \rightarrow [w]\,,\,[w] \quad (\forall t_{cpd} \in T_{cpd})$	$t_{cpd} \rightarrow [w] \quad (\forall t_{cpd} \in T_{cpd})$

Figure 7: Structured representation of compound words (French data set) and related matrix grammars.

Experiment 4 This experiment contrasts *early update* with *max violation update*. Max violation update is trained over 12 epochs, while early update is trained over 25 epochs

Experiment 5 This last experiment contrasts the use of morphology. We remove (*no-morph*) the templates under mood, agreement and subcategories in Figure 6 in order to assess their impact.

6.2 Results

Results with respect to accurracy are given in table 2. Experiments are carried out on the development set with gold part of speech tags (Table 2, left) and predicted part of speech tags (Table 2, right).

Experiment 1 The parser achieves its best result with a beam of size 8 ($K = 8$). Quite surprisingly it achieves already a very correct score with a beam of size 4. We observe that increasing the size of the beam does not proide significant improvements. Using beams of size 16 (or even 32) only bring marginal accurracy improvements, if any, at the expense of more important parsing times.

Experiment 2 This second experiment is apparently more disappointing: synchronisation does not seem to play a significant role on accurracy, or a detrimental one if any. This effect seems caused by a property of the dataset. A more careful analysis of the parser error patterns shows that the parsing model naturally misses many unary reductions. Since the *naive* automaton is biased towards predicting longer sequences with higher weights, it somehow helps to favor longer derivations containing more unary reductions, hence improving the accurracy.

French dev (gold tags)				French dev (predicted tags)			
Expérience	F$\leq$ 40	F	Cov	Expérience	F$\leq$ 40	F	Cov
K=2	85.40	82.74	98.6	K=2	83.24	80.42	98.9
K=4	86.52	83.69	99.5	K=4	84.32	81.34	99.4
K=8	86.80	84.31	99.9	K=8	84.43	81.79	99.8
K=16	86.49	83.95	99.9	K=16	84.59	81.94	99.8
sync	86.41	83.66	99.6	sync	84.06	81.14	99.9
naive	86.52	83.69	99.5	naive	84.32	81.34	99.4
cpd	86.30	83.30	99.2	cpd	83.84	81.17	99.0
base	86.52	83.69	99.5	base	84.32	81.34	99.4
Max Violation	85.98	83.49	99.5	Max Violation	83.42	80.56	99.5
Early Update	86.52	83.69	99.5	Early Update	84.32	81.34	99.4
no-morph	85.23	82.43	99.8	no-morph	83.68	81.05	99.8
all-morph	86.52	83.69	99.5	all-morph	84.32	81.34	99.4

Table 2: Experimental results (development)

Experiment 3 This experiment highlights the problems related to further constraining a parser with approximative search: we observe that parsing coverage is reduced. This can be explained by the fact that further constraining the grammar creates less success states in the automaton and that the parser sometimes has to perform less local decisions without all the necessary information available. This suggests for further work that a more constrained matrix grammar should be used with a more robust search strategy than simple beam search.

Experiment 4 In experiment 4, we observe that max violation update converges twice as fast as early update but we experienced more overfitting problems explaining the lower scores. It is however harder to achieve fair comparisons since the number of iterations is significantly different.

Experiment 5 This last experiment is probably the most significative. We observe that morphology is the variable that allows the parser to improve significantly on French (dev F=81.79). This result thus confirms those observed by (Hohensee and Bender, 2012) on several languages for dependency parsing, yet we had to isolate agreement, refined subcategories and verbal mood to get significant improvements (Figure 6).

Final tests In order to compare this proposal with current state of the art parsers, we provide comparative measures of speed and accurracy with the Berkeley parser (Petrov et al., 2006), known to be representative of the state of the art in accurracy and in speed on French and in accurracy on English (Table 3).

French Test (gold tags)	F$\leq$ 40	F	Cov
K=8	87.14	84.20	99.8
Berkeley	86.44	83.96	99.9
French Test (predicted tags)	F$\leq$ 40	F	Cov
K=8	84.33	81.43	99.8
Berkeley	83.16	80.73	99.9
French test (raw text)	F$\leq$ 40	F	Cov
Berkeley	83.59	81.33	99.9
English test (gold tags)	F$\leq$ 40	F	Cov
K=8	90.2	89.5	100
English test (pred tags)	F$\leq$ 40	F	Cov
K=8	89.7	89.1	100
English test (raw text)	F$\leq$ 40	F	Cov
Berkeley	-	90.1	-

Table 3: Experimental results (test)

Interestingly parsing the English dataset with templates designed on French data is almost state of the art on English (dev F=89.3, test F=89.1). This suggests that feature engineering is a less important issue than often thought and it also suggests that the parser is likely to be easy to adapt to other languages by generalizing the method used to parse English.

Although the differences are modest, the parser is state of the art on French (F=81.43) if we compare with the Berkeley parser (F=80.73) known to be indicative of the state of the art on French (Seddah et al., 2013) and if we ignore ensemble and semi-supervised parsers.

The most important difference is related to speed. (Petrov et al., 2006) is reported to be the fastest phrase structure parser for English by (Huang and Sagae, 2010) where the authors compare with (Charniak, 2000). Yet (Petrov et al., 2006) is a polynomial time parser, while this one has linear time behaviour. We compared the speed of both parsers on the same hardware (ignoring input/output times) and we find (Petrov et al., 2006) has an average parse time of $t_\mu = 0.28$s with maximum $t_{max} = 10.27$s while our linear time algorithm has mean time $t_\mu = 0.06$s and maximum $t_{max} = 0.1$s with beam 4 and $t_\mu = 0.1s, t_{max} = 0.5s$ with beam 8 (Figure 8). Further constraining the automaton shows to be useful for speed, since for experiment 3, $t_\mu = 0.04s$ with $K = 4$ which is clearly faster.

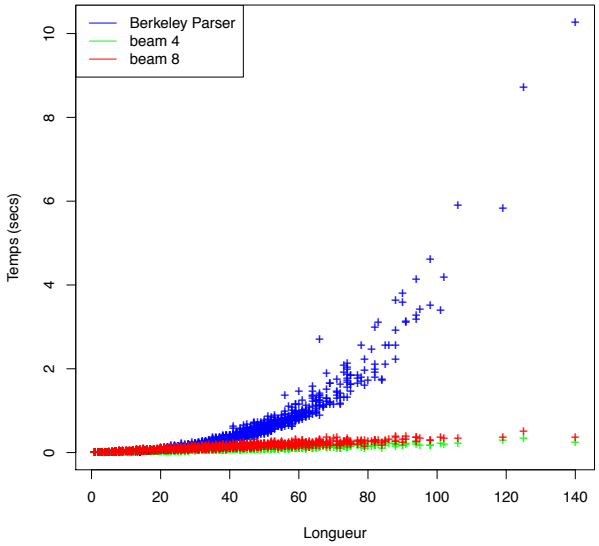


Figure 8: Parsing times

We finally observe on English that the LR strategy performs reasonably accurately and faster than the implementation of (Petrov et al., 2006) whereas optimisations of PCFG-LA described by (Bodenstab et al., 2011) are significantly less accurate and not significantly faster. Although it is currently hard to to compare directly on French with the most similar proposal, the Chinese/English parser described by (Zhu et al., 2013), since it does not handle morphology. With respects to speed, their impressive result on English suggests that there is still room for speed improvements without loss of accurracy.

7 Conclusion

To our knowledge, this is the first formulation of a discriminative LR-automaton driven parser for natural language. This LR inspired algorithm shares many properties with shift reduce parsers for phrase structure grammar described by (Sagae and Lavie, 2006) and (Zhu et al., 2013). (Sagae and Lavie, 2006) describe a first version of this kind of algorithm using a weighting system based on a local maximum entropy classifier. It thus enables them to use a best first search strategy that allows in principle to achieve near-optimal parsing. By contrast, like (Zhu et al., 2013), we use here a global perceptron algorithm together with a beam based breadth first search to which we add an explicit LR component. The LR automaton allows us to guarantee that the parser generates a viable prefix. We believe the LR framework can also shed light on theoretical, practical and experimental issues related to phrase structure parsing by comparison with dependency parsing. However using the LR automaton to constrain the parsing model

for multi-word expressions turns out to be disappointing since it forces the parser to take less local decisions for which the beam approximation is not well suited. This suggests for future work to explore search methods aiming to achieve optimality (Zhao et al., 2013).

Mirroring a common practice in dependency parsing, the parser also provides a first support for phrase structure parsing of morphologically rich languages thanks to structured word representations. The richer lexical structure makes morphological information available during the parsing process. For the case of French it enables, among others, to integrate agreement in the parsing model. This simple integration of morphology then allows the parser to achieve state of the art accurracy on French. Since in principle nothing in the algorithm is specific to French, we expect to generalize and experiment with the model on other morphologically rich languages. Further work for such languages is expected to involve a refinement of the interface with morphology along the lines of (Hatori et al., 2012; Bohnet et al., 2013).

Acknowledgements

I am grateful to Maximin Coavoux who helped at some stages of the implementation. I am also grateful to Benoit Sagot for his encouragments and several discussions that helped to clarify the contents of this paper, and finally to Djamé Seddah who brought his expertise with the actual data sets.

References

Anne Abeillé, L. Clément, and F. Toussenel. 2003. Building a treebank for french. In *Treebanks*. Kluwer.

Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, and Monica S. Lam. 2006. *Compilers: Principles, Techniques, and Tools*. Addison Wesley.

Abhishek Arun and Frank Keller. 2005. Lexicalization in crosslinguistic probabilistic parsing: The case of french. In *Association for Computational Linguistics*.

Nathan Bodenstab, Aaron Dunlop, Keith Hall, and Brian Roark. 2011. Beam-width prediction for efficient context-free parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, Portland, Oregon, June. Association for Computational Linguistics.

Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajic. 2013. Joint morphological and syntactic analysis for richly inflected languages. *TACL*, 1:415–428.

Ted Briscoe and John A. Carroll. 1993. Generalized probabilistic lr parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–59.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *North American Association for Computational Linguistics*.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

Benoit Crabbé and Marie Candito. 2008. Expériences d'analyses syntaxique statistique du français. In *Actes deTALN 2008*.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning . In LREC 2006. 2006. Generating typed dependency parses from phrase structure parses. In *Language ressources and evaluation conference*.

Pascal Denis and Benoît Sagot. 2012. Coupling an annotated corpus and a lexicon for state-of-the-art pos tagging. *Language Resources and Evaluation*, 46(1).

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of the Association for Computational Linguistics*.

Gerald Gazdar, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag. 1985. *Generalized Phrase Structure Grammar*. Cambridge, MA: Harvard University Press and Oxford: Basil Blackwell's.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2012. Incremental joint approach to word segmentation, pos tagging, and dependency parsing in chinese. In *ACL (1)*, pages 1045–1053.

Matt Hohensee and Emily M. Bender. 2012. Getting more from morphology in multilingual dependency parsing. In *HLT-NAACL*, pages 315–326.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *North American Association for Computational Linguistics*.

Donald Knuth. 1965. On the translation of languages from left to right. *Information and Control*, 8(6).

Mark-Jan Nederhof and Giorgio Satta. 2010. Algorithmic aspects of natural language processing. In M.J. Atallah and M. Blanton, editors, *Algorithms and Theory of Computation Handbook*. CRC press.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia. Association for Computational Linguistics.

Kenji Sagae and Alon Lavie. 2006. A best-first probabilistic shift-reduce parser. In *Proceedings of the COLING/ACL*.

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Wolińsk, Alina Wróblewska, and Éric Villemonte De La Clergerie. 2013. Overview of the SPMRL 2013 Shared Task: A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Conference on Empirical Methods in Natural Language Processing*.

Masaru Tomita. 1985. An efficient context free parsing algorithm for natural language. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Masaru Tomita. 1988. Graph structured stack and natural language parsing. In *Proceedings of the Association for Computational Linguistics (ACL)*.

Joseph P. Turian and I. Dan Melamed. 2006. Advances in discriminative parsing. In *ACL*.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1).

Kai Zhao, James Cross, and Liang Huang. 2013. Optimal incremental parsing via best-first dynamic programming. In *EMNLP*, pages 758–768.

Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. 2013. Fast and accurate shift-reduce constituent parsing. In *Association for Computational Linguistics*.

Modeling Review Argumentation for Robust Sentiment Analysis

Henning Wachsmuth
Universität Paderborn
s-lab – Software Quality Lab
Paderborn, Germany
henningw@upb.de

Martin Trenkmann, Benno Stein
Bauhaus-Universität Weimar
Webis Group
Weimar, Germany
<1st>.<last>@uni-weimar.de

Gregor Engels
Universität Paderborn
s-lab – Software Quality Lab
Paderborn, Germany
engels@upb.de

Abstract

Most text classification approaches model text at the lexical and syntactic level only, lacking domain robustness and explainability. In tasks like sentiment analysis, such approaches can result in limited effectiveness if the texts to be classified consist of a series of arguments. In this paper, we claim that even a shallow model of the argumentation of a text allows for an effective and more robust classification, while providing intuitive explanations of the classification results. Here, we apply this idea to the supervised prediction of sentiment scores for reviews. We combine existing approaches from sentiment analysis with novel features that compare the overall argumentation structure of the given review text to a learned set of common *sentiment flow patterns*. Our evaluation in two domains demonstrates the benefit of modeling argumentation for text classification in terms of effectiveness and robustness.

1 Introduction

Text classification is a key technique in natural language processing and information retrieval that is applied for several tasks. Standard classification approaches map a text to a vector of lexical and shallow syntactic surface-level features, from which class information is inferred using supervised learning (Manning et al., 2008). Even though the results of such approaches can hardly be explained, they have proven effective for narrow-domain texts with explicit class information (Joachims, 2001; Pang et al., 2002).

However, surface-level features often do not help to classify out-of-domain texts correctly, because they tend to model the domain of the texts and not the classes to be inferred, as we observe in (Wachsmuth and Bujna, 2011) among others. Moreover, they are likely to fail on texts where the class information is implicitly represented by the argumentation of the writer. Such texts are in the focus of popular tasks like authorship attribution, automatic essay grading, and, above all, sentiment analysis. As an example, consider the short hotel review at the top and bottom of Figure 1. It contains more positive than negative statements. Hence, a surface-level analysis would probably classify the review to have a positive overall sentiment polarity. In fact, the argumentation of the review text reveals a clear negative sentiment.

The analysis of argumentation is recently getting more attention (cf. Section 2 for details). With respect to sentiment, related approaches analyze discourse relations (Mukherjee and Bhattacharyya, 2012), identify the different aspects mentioned in a text (Lazaridou et al., 2013), or the like. While these approaches can infer implicit class information from argumentative texts like reviews, they do not address the domain dependency problem of sentiment analysis (Wu et al., 2010). In addition, they still lack explainability, which limits end user acceptance in case of wrong results (Lim and Dey, 2009).

In this paper, we consider the question of how to capture the argumentation of reviews for a domain-robust and explainable text classification. As Figure 1 illustrates, we rely on a shallow model of review argumentation, which represents a text as a sequence of statements that express local sentiment on domain concepts and that are connected by discourse relations. We claim that, by focusing on features that model the abstract argumentation structure of a text, a more robust sentiment analysis can be achieved. At the same time, such an analysis can explain its results based on the underlying model.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 553–564, Dublin, Ireland, August 23-29 2014.

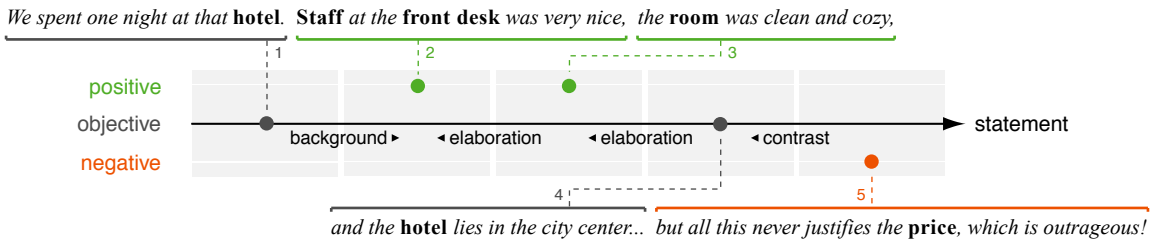


Figure 1: Illustration of our shallow model of review argumentation for a sample review text from the hotel domain. Domain concepts, such as "front desk", are marked in bold. Each circle denotes a statement with local sentiment. The statements are connected by directed discourse relations like "elaboration".

Concretely, here we address the supervised prediction of sentiment scores. To this end, we combine a number of existing argumentation-related features with a novel approach that learns common patterns in sequences of local sentiment through a cluster analysis in order to capture a review's overall argumentation structure. Inspired by explicit semantic analysis (Gabrilovich and Markovitch, 2007), we then compute the similarity of a given review text to each of these *sentiment flow patterns* and we use these similarities as features for sentiment scoring. To explain a predicted score and, hence, to increase user acceptance, both the underlying model and the sentiment flow patterns can be visualized.

We evaluate our approach on reviews of the hotel domain and the movie domain. In comparison to standard baselines, we demonstrate the effectiveness and robustness of modeling argumentation. Our results suggest that especially the sentiment flow patterns learned in one domain generalize well to other domains. Altogether, the contributions of this paper are:

1. A shallow model of review argumentation for text classification that enables a more domain-robust and explainable sentiment analysis (Section 3).
2. A novel feature type named *sentiment flow patterns* that, for the first time, captures the abstract overall argumentation structure of review texts, irrespective of their domain (Section 4).
3. Experimental evidence for the existence of common patterns in the argumentation structure of review texts across domains (Section 5).

2 Related Work

Argumentation plays a key role in human communication and cognition. Its purpose is to provide persuasive information for or against a decision or claim. This involves the identification of facts and warrants justified by a backing or countered by a rebuttal (Toulmin, 1958). Argumentation is studied in various disciplines, such as logic, philosophy, and artificial intelligence. We consider the linguistics perspective, where it is pragmatically viewed as a regulated sequence of speech or text (Walton and Godden, 2006).

In particular, we analyze *monological argumentations* in written text as opposed to dialogical argumentations where participants persuade each other with arguments (Cabrio and Villata, 2012). In terms of text, one of the most obvious forms of monological argumentation can be found in reviews. A review comprises a *positional argumentation*, where an author collates and structures a choice of facts, pros, and cons in order to inform intended recipients about his or her beliefs (Besnard and Hunter, 2008).

According to Mochales and Moens (2011), an *argumentation analysis* targets at "the content of serial arguments, their linguistic structure, the relationship between the preceding and following arguments, recognizing the underlying conceptual beliefs, and understanding within the comprehensive coherence of the specific topic." The authors work on *argumentation mining*, i.e., the detection of different arguments for justifying a conclusion as well as their interactions. Our model of argumentation matches the quoted definition. Similar to the distinction between shallow and deep parsing (Jurafsky and Martin, 2009), our approach can be seen as a *shallow argumentation analysis* in that we consider only the sequence of arguments. This abstraction appears very promising to address text classification.

Unlike *argumentative zoning* (Teufel et al., 2009), which classifies segments of scientific articles according to argumentative functions, we predict the sentiment scores of reviews from a sequence of classified segments. Sentiment scoring is tackled in both computational linguistics (Pang and Lee, 2005) and

information retrieval (Wang et al., 2010). Such kind of sentiment analysis benefits from modeling argumentative discourse (Villalba and Saint-Dizier, 2012). Related works already employ discourse features to detect sentiment polarity. Some rely on complex discourse parsing (Heerschop et al., 2011), whereas others argue that a lightweight approach is more robust for noisy texts (Mukherjee and Bhattacharyya, 2012). We rather follow the latter, but we see discourse only as one part of review argumentation.

In accordance with Lazaridou et al. (2013) who address *aspect-based sentiment analysis*, we additionally analyze the connection of local sentiment to domain concepts and discourse relations. Even more important for us is the *local sentiment flow* in a text. This term was introduced by Mao and Lebanon (2007), who infer a text's global sentiment from its sequence of local (sentence) sentiments, classified with conditional random fields. Their approach converts each sentiment in the sequence to a single feature and learns a mapping from the features to global sentiment. By that, it actually disregards the ordering of local sentiment. In contrast, our sentiment flow patterns measure the similarity between complete sequences of local sentiment. This resembles *explicit semantic analysis* (Gabrilovich and Markovitch, 2007), which classifies texts based on their relatedness to concepts modeled by complete texts.

In (Wachsmuth et al., 2014), we reveal correlations between a review's sentiment score and its local sentiment flow. Similar to Socher et al. (2013), we therefore argue that global sentiment emanates from the composition of local sentiment. The authors model the semantic compositionality of words in given sentences, thus capturing the language of a given domain. Conversely, our sentiment flow patterns focus on the structure of complete texts in order to reduce domain dependency, which is a general problem in text classification (Wu et al., 2010). Among others, existing strategies to tackle this problem align features of the source and the target domain, as we do in (Prettenhofer and Stein, 2010).

Given a vector of features, text classification approaches typically output only a class label (Manning et al., 2008). This renders the understanding and debugging of classification results hard (Kulesza et al., 2011). Instead, our approach explains results by making the argumentation of texts visible. Thereby, we increase intelligibility and, thus, support user acceptance (Lim and Dey, 2009).

3 A Shallow Model of Review Argumentation

This section first sketches our general hypothesis. Then, we present our model of review argumentation.

3.1 Hypothesis behind Modeling Argumentation for Text Classification

Several text classification tasks relate to the argumentation of a text. As an obvious example, *automated essay scoring* explicitly rates argumentative texts, mostly targeting at structural aspects (Dikli, 2006). In *genre identification*, a central concept is the form of texts. Some genre-related tasks address argumentation, e.g. by classifying texts according to their function (Wachsmuth and Bujna, 2011). Criteria in *text quality assessment* often measure structure (Anderka et al., 2012), while *readability* is connected to discourse (Pitler and Nenkova, 2008). *Authorship attribution* profits from argumentation clues like unconsciously used function words (Stamatatos, 2009), and *plagiarism detection*, in the end, aims to check if the argumentation in a fragment of a text refers to the author of the text (Potthast et al., 2013).

We hypothesize that in these and further tasks the class of a text is often decided by the structure of its argumentation rather than by its content, while the content adapts the argumentation to the domain at hand. Following Besnard and Hunter (2008), an argumentation consists of a composition of arguments used to justify a decision or claim. Each argument can be seen as a statement with some evidence. Under our hypothesis, an explicit model of statements and their composition hence supports the identification of domain-independent patterns. Together with the content, the statements enable a fine-grained analysis, while serving as the basis for an explanation. Since the relevant types of statements vary among tasks, we argue that such a model should be task-specific. Below, we investigate reviews on products and services from a sentiment analysis perspective. Because of its positional nature (cf. Section 2), review argumentation makes its arguments explicit, i.e., facts and opinions on different product features and aspects.

3.2 Modeling Review Argumentation for Sentiment Analysis

We consider reviews that comprise a text about some product or service as well as a numerical overall rating. Any other metadata that might be given for reviews is ignored in the following. Our assumption

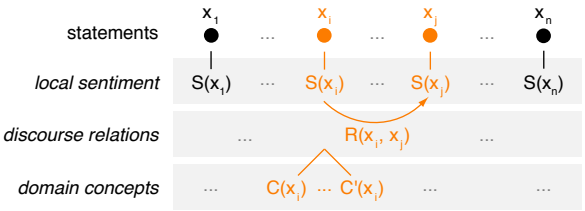


Figure 2: Our shallow model of review argumentation defined by a segmentation into statements and by three functions based on the statements.

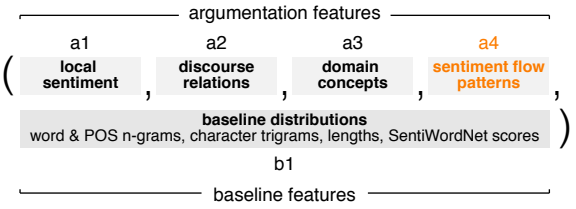


Figure 3: A vector with all five considered feature types including the novel sentiment flow patterns. Each found pattern becomes a single feature in a4.

is that the overall rating denotes a *sentiment score* y from a metric sentiment scale that quantifies the possibly implicit conclusion of the review text in terms of its global sentiment.

Statements To capture a review's argumentation, we model the review's text as a sequence of $n > 0$ statements $x_1, \ldots, x_n$. Here, we define a *statement* x syntactically to be a main clause together with all its subordinate clauses. The notion behind is that, in our experience, such a text segment is usually meaningful on its own while bearing at most one sentiment. Many sentences in reviews comprise series of statements. For instance, the following excerpt from Figure 1 consists of two statements, x_4 and x_5:

x_4: and the hotel lies in the city center... x_5: but all this never justifies the price, which is outrageous!

Based on the set of all statements $\mathbf{X}$, we capture the structure and content of review texts as follows:

Local Sentiment We assume each statement to represent either an objective fact obj, a positive opinion pos, or a negative opinion neg (for a wide applicability, we ignore sentiment intensity). So, there is an unknown function that maps each statement to a local sentiment, e.g. x_4 to obj and x_5 to neg:

$$\textit{local sentiment}: \quad \mathbf{X} \rightarrow \{S(x) \mid S \in \{pos, neg, obj\}\}$$

Discourse Relations As for x_4 and x_5, the composition of statements in a text is, in general, not coincidental. Rather, it implies a structure made up of an ordered choice of statements as well as of a number of directed discourse relations. We define a discourse relation to have some type R of a set of relation types $\mathbf{R}$ and to relate two (typically neighboring) statements, e.g. *contrast(x_5, x_4)* in the example above. The following function hence can be understood as a shallow version of the *rhetorical structure theory* (Mann and Thompson, 1988):

$$\textit{discourse relations}: \quad \mathbf{X} \rightarrow \{R(x_i, x_j) \mid 1 \leq i, j \leq n \; ; \; R \in \mathbf{R}\}$$

Domain Concepts The argumentation structure of a text is bound to the domain at hand through the text's content. In particular, a review text discusses a subset of the *domain concepts* $\mathbf{C}$ that are associated to a product or service, each being referred to in one or more statements. For instance, x_5 discusses the price of the hotel, i.e., *price(x_5)*. We capture the domain concepts in statements as follows:

$$\textit{domain concepts}: \quad \mathbf{X} \rightarrow \{C(x_i) \mid 1 \leq i \leq n \; ; \; C \in \mathbf{C}\}$$

Altogether, our model represents a review text as a sequence of interrelated statements of certain types and content. Figure 2 illustrates the defined functions. An instance of the model is visualized in Figure 1. The model is an abstraction of argumentation, covering some information only implicitly if at all (e.g. lexical or syntactic clues). However, it can be extended by further information, as we do below.

4 Features for Robust Sentiment Analysis and Explanation

We now present different types of features for supervised learning that capture both distributional and structural aspects of review argumentation based on our shallow model. Here, we assume that all information represented in the model is given, but Section 5 analyzes the effects of inferring the information from a text. Figure 3 gives an overview of the vector with all feature types that we consider, including a common set of baseline features (b1). The goal of all argumentation features (a1–a4) is twofold: (1) To enable an effective and robust sentiment analysis. (2) To provide means to explain analysis results.

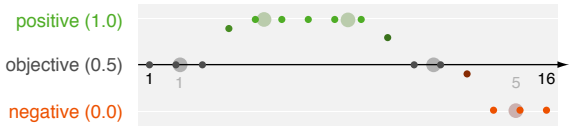


Figure 4: Illustration of a length-normalized version (small circles) of the sample local sentiment flow from Figure 1 (big circles) for length 16.

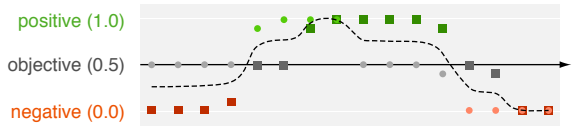


Figure 5: Sketch of the construction of a sentiment flow pattern (dashed curve), here from two sample local sentiment flows (circles and squares).

4.1 Quantification of Distributional Argumentation Aspects

In terms of the distributional aspects of the three functions introduced in Section 3, we combine a selection of ideas from existing sentiment analysis approaches that are related to review argumentation. Some features of the types described in the following are selected only if they occur frequently in a given set of training texts. Thus, the concrete numbers of features vary, as we see in the evaluation in Section 5.

Local Sentiment (a1) In (Wachsmuth et al., 2014), we stress the impact of the distribution of local sentiment. Accordingly, here we determine the frequencies of all types of local sentiment in the given text as well as of series of statements with the same type and of changes from one type to another. Also, we have features that denote the local sentiment at specific positions like the first and last two statements, and we compute the average local sentiment. For the latter, we map pos to 1.0, obj to 0.5, and neg to 0.0.

In addition, we follow Mao and Lebanon (2007) in that we capture the local sentiment flow based on the defined mapping. To preserve the original flows as far as possible, we length-normalize the sequence of values using non-linear interpolation with subsequent sampling. Figure 4 shows an example.

Discourse Relations (a2) We count the occurrences of different discourse relation types from (Mann and Thompson, 1988), e.g. *contrast* or *elaboration* (in Section 5, we distinguish a subset of ten types). To model connections between sentiment and discourse, we do the same for all frequently occurring combinations of discourse relation types and local sentiment of the related statements, e.g. *contrast(*pos*,* neg*)* or *contrast(*neg*,* pos*)*. By that, we imitate Lazaridou et al. (2013) to some extent.

Domain Concepts (a3) With the same intention, we determine the most frequent domain concepts in the given training set and we compute how often each concept cooccurs with each type of local sentiment. Examples from the sample text in Figure 1 are *hotel(*obj*)* or *price(*neg*)*. Moreover, we count the number of different domain concepts as well as the instances of all possibly distinguished types of domain concepts, which would be *product* (like "hotel") and *product feature* (like "price") in the given case.

Types a1–a3 refer to important characteristics of review argumentation. However, none of them captures a review's overall argumentation structure. Even the local sentiment flow in a1 rather measures the impact of local sentiment at different positions. The reason behind is that the flow positions are represented by individual features. Hence, common learning approaches like regression will naturally tend to assign positive weights to all positions, not considering the sentiment flow as a whole.

4.2 Learning of Structural Argumentation Aspects

To capture the impact of the structure of an argumentation, we introduce a novel feature type based on the local sentiment flows of texts only. The idea behind resembles *explicit semantic analysis* (Gabrilovich and Markovitch, 2007) in that every single feature represents the similarity to a complete flow:

Sentiment Flow Patterns (a4) We first construct a set of common *sentiment flow patterns* from a set of known training review texts, where each pattern denotes the average of a set of similar local sentiment flows of normalized length. Given an unknown review text, we then measure the similarity of its normalized local sentiment flow to each constructed pattern. The set of these similarities forms a4.

Figure 5 exemplifies the pattern construction. Our hypothesis behind sentiment flow patterns is that similar local sentiment flows entail similar sentiment scores. Accordingly, flows that construct a pattern should be as similar as possible and flows of different patterns as dissimilar as possible. Therefore, we apply *clustering* (Manning et al., 2008) to partition the flows of all texts from the given training set based on some flow similarity function (in Section 5, we use the manhattan distance). The centroid of each ob-

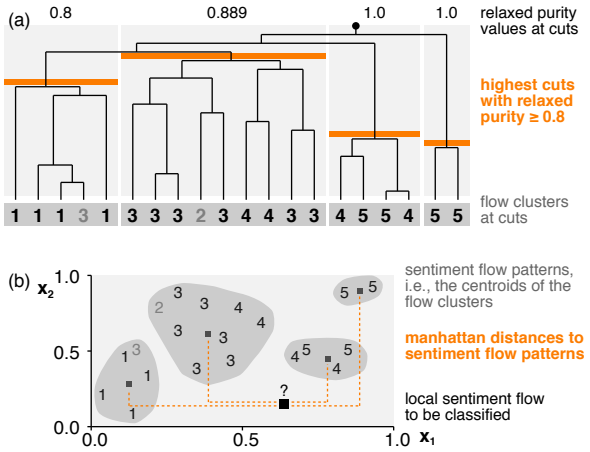


Figure 6: (a) A purity threshold of 0.8 derives four clusters from a hierarchical clustering of 20 local sentiment flows represented by their scores from 1 to 5. (b) 2D plot of computing distances to the sentiment flow patterns, i.e., the clusters' centroids.

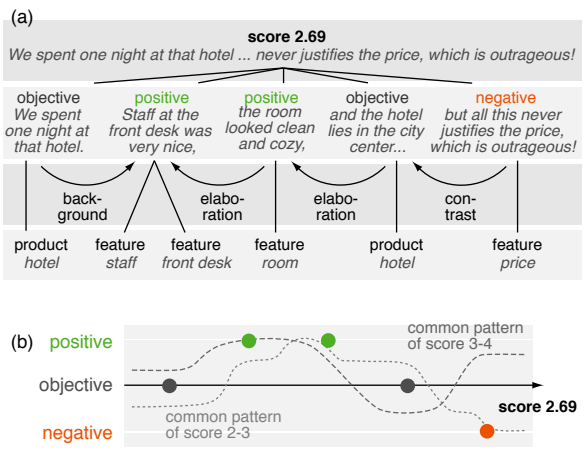


Figure 7: Two possible explanations of scoring the sample text from Figure 1: (a) Graph visualization of our model of review argumentation. (b) Comparison of the local sentiment flow of the text with the two most similar sentiment flow patterns.

tained cluster ω then becomes a sentiment flow pattern. Since we know the sentiment scores associated to the flows in the training set, we can measure the *purity* of a cluster ω, which here denotes the fraction of those flows ω_{y^*} in ω whose score equals the majority score y^* in ω (Manning et al., 2008). The original purity definition, however, assumes exactly one correct score for each flow. Here, this would mean that a flow alone decides a score. Instead, for larger sentiment scales, we propose to relax the purity measure by assuming also the dominant neighbor of the majority score as correct:

$$relaxed\ purity(\omega) \;=\; \left(|\omega_{y^*}| + \max(|\omega_{y^*-1}|,|\omega_{y^*+1}|)\right) / \,|\omega|$$

We seek for clusters with a high purity, because such clusters support that similarities between flows and patterns indicate specific sentiment scores. At the same time, the number of clusters should be small in order to achieve a high average cluster size and, thus, a high commonness of the patterns. For this purpose, we rely on *hierarchical clustering*, where we can easily find a flat clustering with a certain number of clusters through cuts at appropriate nodes in the binary tree of the associated hierarchy. Pattern construction profits from compact clusters, suggesting to compute distances between clusters from their *group-average links* (Manning et al., 2008). To minimize the number of clusters, we search for all nodes closest to the tree's root that represent clusters with a purity above some threshold, e.g. 0.8 in the example in Figure 6(a). The centroids of these clusters become sentiment flow patterns, if they are made up of some minimum number of flows.

At the end of the clustering process, we remain with one feature for each constructed sentiment flow pattern. Given a review text to be classified, we then compute its normalized local sentiment flow and we measure the flow's distance to all patterns. Each distance represents one similarity in the feature type a4. Figure 6(b) sketches the computation of the distances, mapped into two dimensions.

4.3 Comparison with Baseline Approaches

In the evaluation below, we compare the feature types a1 to a4 with the well-known sentiment scoring approach of Pang and Lee (2005) in terms of effectiveness. Our focus, however, is the robustness of modeling argumentation structure in contrast to standard text classification features employed in many other approaches, such as n-grams or variations of them (Qu et al., 2010). To this end, we also integrate the following baseline features, most of which model a text at the lexical and syntactic level:

Baseline Distributions (b1) We compute the distributions of all word and part-of-speech unigrams, bigrams, and trigrams as well as of all character trigrams that frequently occur in a given training set. In addition, we determine the length of the given text in different units and some average *SentiWordNet* scores with respect to both the first and the average senses of its words (Baccianella et al., 2010).

4.4 Explanation of Sentiment Scores

We propose a shallow statistical argumentation analysis that learns to predict sentiment scores on a training set of review texts. After prediction, we can directly exploit the information captured in our model as well as the values of the feature types a1–a4 in order to explain the predicted score. Two possible explanations are visualized in Figure 7, while a combination of them is exemplified in Figure 1. We believe that such explanations can increase a user's confidence in statistical analysis results and, hence, the acceptance of corresponding applications. To demonstrate the analysis and explanation of review argumentation, we provide a free-to-use tool and webservice at http://www.arguana.com.

5 Evaluation of Modeling Argumentation for Sentiment Scoring

In this section, we evaluate the effectiveness and domain robustness of modeling argumentation for sentiment scoring with a focus on the sentiment flow patterns. The source code of the evaluation can be found at http://www.arguana.com. Our experiments are based on two English text corpora with reviews from the hotel domain and the movie domain, respectively. In both cases, we leave out the review titles for generality, because our approach targets at arbitrary reviews including those without a title.

Text Corpora On the one hand, we process the *ArguAna TripAdvisor corpus* that we have introduced in (Wachsmuth et al., 2014). This corpus compiles a collection of 2,100 reviews of hotels from seven locations, balanced with respect to their sentiment scores between 1 and 5. All of the reviews' texts are segmented into statements with an average of 14.8 statements per text. Each statement is annotated as an objective fact, a positive, or a negative opinion. Moreover, all mentions of domain concepts are marked as such. The corpus is also available at http://www.arguana.com, free for scientific use. In the experiments, we rely on the provided corpus split with 900 reviews from three hotel locations in the training set, and 600 reviews from two locations in the validation set and test set each.

On the other hand, we use the *Sentiment Scale dataset* (Pang and Lee, 2005) consisting of 5,006 movie reviews that are split into four text corpora according to their authors (*Author a*, *b*, *c*, and *d*). From these, we have discarded eight reviews due to encoding problems. We choose the provided sentiment scale from 0 to 2, so we can logically map the scale of the hotel reviews (1–5) to it for a domain transfer. In particular, scores 1–2 are mapped to 0, 3 to 1, and 4–5 to 2. On average, the movie reviews are much longer with 36.1 statements per text. Since no local sentiment annotations are given, we also process the *subjectivity dataset* (Pang and Lee, 2004) and the *sentence polarity dataset* (Pang and Lee, 2005) in order to develop classifiers for sentence sentiment. Accordingly, we assume each movie review sentence to denote one statement. To directly compare our results to those of Pang and Lee (2005), we perform 10-fold cross-validation separately on the dataset of each single author, averaged over five runs.

Preprocessing For feature computations, we preprocess all texts with a tokenizer, a sentence splitter, and the part-of-speech tagger from (Schmid, 1995). We employ lexicon-based extractors for discourse relations and domain concepts, which aim at a high precision while not being able to recognize unseen instances. The former resembles the lightweight approach of Mukherjee and Bhattacharyya (2012). Primarily, it looks for conjunctions that indicate certain discourse relations, such as "but" or "because". The latter detects exactly those domain concepts that are annotated largely consistently in the training set of the ArguAna TripAdvisor corpus. Thus, it helps only on the hotel reviews. These reviews are segmented into statements with a respective algorithm that comes with the corpus.

For both domains, we have trained linear support vector machines (SVMs) from Chang and Lin (2011) that classify the subjectivity of each statement (opinion or fact) and the polarity of each opinion (positive or negative). They use 1k to 2k features of different types: word and part-of-speech unigrams, character trigrams, SentiWordNet scores (Baccianella et al., 2010), and some special features like the first word of a statement or its position in the text. On the test set of the hotel domain, the classifiers have an accuracy of 78.1% for subjectivity and of 80.4% for polarity. In the movie domain, we achieve a subjectivity accuracy of 91.1%, but a polarity accuracy of only 73.8% (measured through 10-fold cross-validation).

Feature Computation We determine one distinct feature set for each evaluated text corpus made up of the feature types presented in Section 4. Where necessary, we divide the computed feature values by

the length of the text (in tokens or statements, as appropriate), in order to ensure that all feature values always lie between 0 and 1.

Local sentiment flows are normalized to length 30 in case of the hotel reviews and to length 60 in case of the movie reviews, which allows us to represent most of the original flows without loss. Altogether, feature type a1 sums up to 50 and 80 features, respectively. For a2, a3, and b1, we consider only those features whose frequency in the training texts exceeds some specified threshold. For instance, a word unigram is taken into account within b1 only if it occurs in at least 5% of the hotel reviews or 10% of the movie reviews, respectively. As a result, the number of evaluated features varies depending on the processed text corpus. Concretely, we obtain 64 to 78 features for discourse relations (a2), 78 to 114 for domain concepts (a3), and 1026 to 2071 baseline features (b1). More details are given in the instruction and configuration files that come with the provided source code.

To construct sentiment flow patterns (a4), we have developed an agglomerative hierarchical clusterer that implements the approach from Section 4.2. After some tests with different settings, we decided to measure flow and cluster similarity using group-average link clustering based on the manhattan distance between the length-normalized local sentiment flows. For the hierarchy tree cuts, we use a purity threshold of 0.8, where we take the relaxed purity for the sentiment scale 1–5 of the hotel reviews, but the original purity for the movie reviews (because of the limited scale from 0 to 2). All centroids of clusters with at least three flows become a sentiment flow pattern, resulting in 16 to 86 features in a4.

Sentiment Scoring On the hotel reviews, we compute the root mean squared error of linear sentiment score regression trained using stochastic gradient descent (SGD) from *Weka 3.7.5* (Hall et al., 2009). Both the regularization parameter and the learning rate of SGD are set to 10^{-5}, whereas we determine the epochs parameter of SGD on the validation set. Then, we measure the error on the test set.

For the comparison to (Pang and Lee, 2005), we predict the scores of the movie reviews using classification, which additionally stresses the domain change. In particular, we measure the accuracy of a linear 1-vs.-1 multi-class SVM with probability estimates and normalization. While we optimize the cost parameter of the SVMs in the in-domain task, we rely on the default value (1.0) for the domain transfer.

5.1 Effectiveness of Modeling Argumentation

First, we measure the theoretically possible scoring effectiveness of all feature types within one domain. To this end, we compare the feature types based on the ground-truth annotations of the ArguAna TripAdvisor corpus. The column *Corpus* of Table 1 lists the resulting root mean squared errors. As can be seen, all argumentation feature types clearly outperform the baseline distributions (b1) and improve strongly over random guessing. The distributional local sentiment (a1) does best with an error of 0.77, whereas the domain concepts perform worst among a1 to a4. Still, they result in an 0.12 lower root mean squared error than the baseline distributions (b1). Overall, the lowest observed error is 0.75, achieved by the SVM with all features as well as by two subsets of the argumentation features alone.

In practice, no ground-truth annotations are given, so we need to create annotations in the review texts ourselves using the preprocessing described above. This in turn changes the feature set and the respective values of the argumentation features. The third column of Table 1 (*Self*) shows that such a resort to self-created annotations leads to a root mean squared error increase of 0.14 to 0.22 for the types a1 to a4. Nevertheless, the argumentation features succeed over the baseline distributions with 0.94 as opposed to 1.11, which demonstrates the effectiveness of modeling the argumentation of hotel reviews.

5.2 Robustness of Modeling Argumentation Structure

We hypothesize that the developed structure-based argumentation features are robust against domain transfer to a wide extent. To investigate this, we classify sentiment scores using SVMs based either on all or on one single feature type (except for a3, for lack of movie domain concept extractors) in two tasks on the four movie datasets: (1) with training in the movie domain (through 10-fold cross-validation), and (2) with training out-of-domain on the hotel review training set.

Figure 8 contrasts the accuracy results for the two tasks and compares them to the best SVM approach of Pang and Lee (2005), i.e., *ova* (open squares). In the in-domain task, our SVM based on all feature types (black squares) is significantly better than *ova* on one dataset (*Author a*) and a little worse on

Feature type		Corpus	Self
none	Random guessing	1.41	1.41
a1	Local sentiment	0.77	0.99
a2	Discourse relations	0.84	1.01
a3	Domain concepts	0.99	1.13
a4	Sentiment flow patterns	0.86	1.07
b1	Baseline distributions	1.11	1.11
a1–a4	Argumentation features	0.76	0.94
a2, a3, a4	w/o local sentiment	0.79	0.99
a1, a3, a4	w/o discourse relations	0.76	0.97
a1, a2, a4	w/o domain concepts	**0.75**	0.95
a1, a2, a3	w/o sentiment flow patterns	**0.75**	0.95
all	All features	**0.75**	**0.93**

Table 1: Root mean squared error of sentiment score regression on the hotel review test set for all evaluated features types and for different combinations of these types. Features are computed based on ground-truth annotations (*Corpus*) or based on self-created annotations (*Self*).

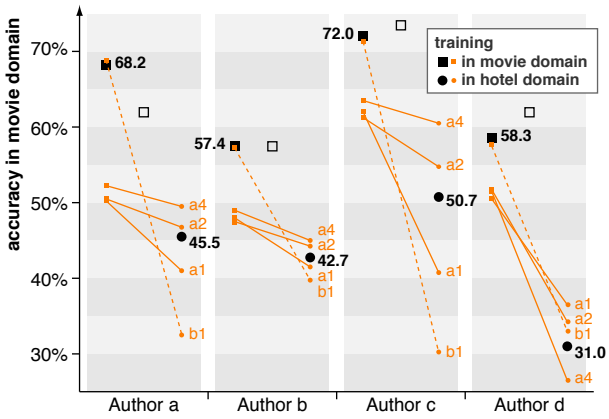


Figure 8: Sentiment scoring accuracy of the SVM based on feature type a1, a2, a4, and b1 (black icons) and of each SVM based on one of these types (orange icons) on the four movie datasets, when trained on movie reviews (squares) or on hotel reviews (circles). Open squares: *ova* from (Pang and Lee, 2005).

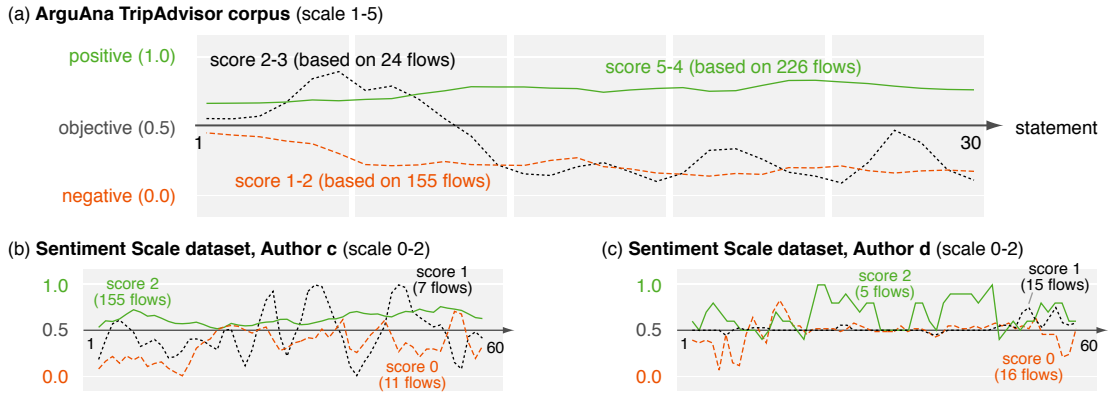


Figure 9: (a) The three most common sentiment flow patterns in the training set of the ArguAna TripAdvisor corpus, labeled with their associated sentiment scores. (b–c) The according sentiment flow pattern for each possible score of the texts of *Author c* and *Author d* in the Sentiment Scale dataset, respectively.

two other datasets (*Author c* and *Author d*). On all four datasets, a4 classifies sentiment scores more accurately than both a1 and a2, but none of the argumentation feature types can compete with the baseline distributions (b1). We suppose that the reason behind mainly lies in the limited effectiveness of our opinion polarity classifier, which reduces the impact of all features that rely on statement sentiment.

Conversely, b1 fails completely in the out-of-domain task (from squares to circles) with accuracy drops of up to 41% (on *Author c*). This indicates a large *covariate shift* (Shimodaira, 2000) in the distribution of the baseline features. In contrast, a1, a2, and a4 suffer much less from the domain transfer. Especially the accuracy of the sentiment flow patterns (a4) remains stable on three of the four datasets and, hence, provides strong support for our hypothesis. In case of *Author c*, the SVM based on a4 alone even achieves a significantly higher accuracy than the SVM based on all features (60.5% as opposed to 50.7%), thus offering evidence for the decisiveness of the structure of an argumentation. Only on *Author d*, all four evaluated feature types similarly fail when trained on hotel reviews with a4 being the worst. Apparently, the argumentation structure in the texts of *Author d* differs from the others, which is reflected by the found sentiment flow patterns and which we therefore finally analyze.

5.3 Insights into Sentiment Flow Patterns

In Figure 9, we plot the three most common sentiment flow patterns in the training set of the ArguAna TripAdvisor corpus (with self-created annotations) as well as the respective patterns in the movie reviews

of *Author c* and *Author d* for each possible sentiment score. In total, we found 38 sentiment flow patterns in the hotel reviews, meaning that a4 consists of 38 features in this case. As depicted in Figure 9(a), they are constructed from the local sentiment flows of up to 226 texts. One of the 75 patterns of *Author c* results from 155 flows, whereas each of the 41 patterns of *Author d* represents at most 16 flows.

With respect to the depicted sentiment flow patterns, the movie reviews show less clear sentiment but more changes of local sentiment than the hotel reviews. While there appears to be a certain similarity in the overall argumentation structure between the hotel reviews and the movie reviews of *Author c*, two of the three patterns of *Author d* contain only little clear sentiment at all, especially in the middle parts. The disparity of the *Author d* dataset is additionally emphasized by the different proportions of opinions in the evaluated text corpora. In particular, 79.7% of all statements in the ArguAna TripAdvisor corpus are opinions, but only 36.5% of the sentences of *Author d* are classified as subjective. The proportions of the three other movie datasets at least range between 58.4% and 66.5%. These numbers also serve as a general explanation for the limited accuracy of a1, a2, and a4 in the movie domain.

A solution to achieve higher accuracy and to further improve the domain robustness of the structure-based argumentation features might be to construct flow patterns from the subjective statements or from the changes of local sentiments only, which we leave for future work. Here, we conclude that our novel feature type a4 does not yet solve the domain dependency problem, but it still defines a promising step towards a more domain-robust sentiment analysis.

6 Conclusion

Text classification tasks like sentiment analysis are domain-dependent and tend to be hard on texts that comprise an involved argumentation, such as reviews. To classify the sentiment scores of reviews, we model a review's text as a composition of local sentiment, discourse relations, and domain concepts. Based on this shallow model of argumentation, we combine existing sentiment analysis approaches with novel features that capture the abstract overall argumentation structure of reviews irrespective of their domain and their linguistic style. In particular, we learn common sequences of local sentiment in reviews through clustering in order to then compare a given review to each of these learned *sentiment flow patterns*. Our evaluation on hotel and movie reviews suggests that the sentiment flow patterns generalize well across domains and it indicates the effectiveness of modeling argumentation. In addition, both the patterns and our model help to explain sentiment scoring results, as exemplified.

Due to errors in the preprocessing of texts, some obtained effectiveness gains are rather small, though. In the future, we seek to develop features that are less affected from preprocessing. A promising variation in this respect is e.g. to learn patterns based on the changes of local sentiment only. Also, we plan to analyze common sequences of discourse relations in order to capture the argumentation structure of a text in an even more domain- and language-independent manner. By that, we contribute to the general research on robust and explainable text classification. As outlined in Section 3, many text classification tasks can profit from modeling argumentation. For this purpose, other types of statements, relations, and domain concepts will be needed as well as, in some cases, a deeper argumentation analysis.

Acknowledgments

This work was funded by the German Federal Ministry of Education and Research (BMBF) under contract number 01IS11016A as part of the project "ArguAna", http://www.arguana.com.

References

Maik Anderka, Benno Stein, and Nedim Lipka. 2012. Predicting Quality Flaws in User-generated Content: The Case of Wikipedia. In *Proceedings of the 35th International ACM Conference on Research and Development in Information Retrieval*, pages 981–990.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 2200–2204.

Philippe Besnard and Anthony Hunter. 2008. *Elements of Argumentation*. The MIT Press.

Elena Cabrio and Serena Villata. 2012. Combining Textual Entailment and Argumentation Theory for Supporting Online Debates Interactions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers*, pages 208–212.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27.

Semire Dikli. 2006. An Overview of Automated Scoring of Essays. *Journal of Technology, Learning, and Assessment*, 5(1).

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, pages 1606–1611.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18.

Bas Heerschop, Frank Goossen, Alexander Hogenboom, Flavius Frasincar, Uzay Kaymak, and Franciska de Jong. 2011. Polarity Analysis of Texts Using Discourse Structure. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 1061–1070.

Thorsten Joachims. 2001. A Statistical Learning Model of Text Classification for Support Vector Machines. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 128–136.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. Prentice-Hall, 2nd edition.

Todd Kulesza, Simone Stumpf, Weng-Keen Wong, Margaret M. Burnett, Stephen Perona, Andrew Ko, and Ian Oberst. 2011. Why-oriented End-user Debugging of Naive Bayes Text Classification. *ACM Transactions on Interactive Intelligent Systems*, 1(1):2:1–2:31.

Angeliki Lazaridou, Ivan Titov, and Caroline Sporleder. 2013. A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1630–1639.

Brian Y. Lim and Anind K. Dey. 2009. Assessing Demand for Intelligibility in Context-aware Applications. In *Proceedings of the 11th International Conference on Ubiquitous Computing*, pages 195–204.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8(3):243–281.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Yi Mao and Guy Lebanon. 2007. Isotonic Conditional Random Fields and Local Sentiment Flow. *Advances in Neural Information Processing Systems*, 19:961–968.

Raquel Mochales and Marie-Francine Moens. 2011. Argumentation Mining. *Artificial Intelligence and Law*, 19(1):1–22.

Subhabrata Mukherjee and Pushpak Bhattacharyya. 2012. Sentiment Analysis in Twitter with Lightweight Discourse Analysis. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1847–1864.

Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity. In *Proceedings of 42th Annual Meeting of the Association for Computational Linguistics*, pages 271–278.

Bo Pang and Lillian Lee. 2005. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 115–124.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs Up?: Sentiment Classification Using Machine Learning Techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, pages 79–86.

Emily Pitler and Ani Nenkova. 2008. Revisiting Readability: A Unified Framework for Predicting Text Quality. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 186–195.

Martin Potthast, Matthias Hagen, Michael Völske, and Benno Stein. 2013. Crowdsourcing Interaction Logs to Understand Text Reuse from the Web. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1212–1221.

Peter Prettenhofer and Benno Stein. 2010. Cross-Language Text Classification using Structural Correspondence Learning. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 1118–1127.

Lizhen Qu, Georgiana Ifrim, and Gerhard Weikum. 2010. The Bag-of-Opinions Method for Review Rating Prediction from Sparse Text Patterns. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 913–921.

Helmut Schmid. 1995. Improvements in Part-of-Speech Tagging with an Application to German. In *Proceedings of the ACL SIGDAT-Workshop*, pages 47–50.

Hidetoshi Shimodaira. 2000. Improving Predictive Inference under Covariate Shift by Weighting the Log-Likelihood Function. *Journal of Statistical Planning and Inference*, 90(2):227–244.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Efstathios Stamatatos. 2009. A Survey of Modern Authorship Attribution Methods. *Journal of American Society for Information Science and Technology*, 60(3):538–556.

Simone Teufel, Advaith Siddharthan, and Colin Batchelor. 2009. Towards Discipline-independent Argumentative Zoning: Evidence from Chemistry and Computational Linguistics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1493–1502.

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

Maria Paz Garcia Villalba and Patrick Saint-Dizier. 2012. Some Facets of Argument Mining for Opinion Analysis. In *Proceedings of the 2012 Conference on Computational Models of Argument*, pages 23–34.

Henning Wachsmuth and Kathrin Bujna. 2011. Back to the Roots of Genres: Text Classification by Language Function. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 632–640.

Henning Wachsmuth, Martin Trenkmann, Benno Stein, Gregor Engels, and Tsvetomira Palakarska. 2014. A Review Corpus for Argumentation Analysis. In *Proceedings of the 15th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 115–127.

Douglas Walton and David M. Godden, 2006. *Considering Pragma-Dialectics*, chapter The Impact of Argumentation on Artificial Intelligence, pages 287–299. Erlbaum.

Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent Aspect Rating Analysis on Review Text Data: A Rating Regression Approach. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 783–792.

Qiong Wu, Songbo Tan, Miyi Duan, and Xueqi Cheng. 2010. A Two-Stage Algorithm for Domain Adaptation with Application to Sentiment Transfer Problems. In *Information Retrieval Technology*, volume 6458 of *Lecture Notes in Computer Science*, pages 443–453.

Biber Redux: Reconsidering Dimensions of Variation in American English

Rebecca J. Passonneau
Center for Computational Learning Systems
Columbia University
New York, New York USA
becky@ccls.columbia.edu

Nancy Ide
Department of Computer Science
Vassar College
Poughkeepsie, New York USA
ide@cs.vassar.edu

Songqiao Su
Department of Computer Science
Columbia University
New York, New York USA
ss4555@columbia.edu

Jesse Stuart
Department of Computer Science
Vassar College
Poughkeepsie, New York USA
jestuart@cs.vassar.edu

Abstract

Genre classification has been found to improve performance in many applications of statistical NLP, including language modeling for spoken language, domain adaptation of statistical parsers, and machine translation. It has also been found to benefit retrieval of spoken or written documents. At its base, however, classification assumes separability. This paper revisits an assumption that genre variation is continuous along multiple dimensions, and an early use of principal component analysis to find these dimensions. Results on a very heterogeneous corpus of post-1990s American English reveal four major dimensions, three of which echo those found in prior work and the fourth depending on features not used in the earlier study. The resulting model can provide a basis for more detailed analysis of sub-genres and the relation between genre and situations of language use, as well as a means to predict distributional properties of new genres.

1 Introduction

Although a precise definition of the term "genre" has traditionally proven to be elusive, it cannot be disputed that a genre represents a set of *shared regularities* among written or spoken documents that enables readers, writers, listeners and speakers to signal discourse function, and that conditions their expectations of linguistic form. Genre distinctions are therefore an important aspect of language use and understanding. They clearly have a role to play in statistical language processing, which relies on regularities of form as well as content. Indeed, with the advent of the Web, statistical methods for genre differentiation have been applied to information retrieval to limit search criteria and organize results (Karlgren and Cutting, 1994; Kessler et al., 1997; Mehler et al., 2010; Ward and Werner, 2013), and the study of genres on the web has become a sub-field in its own right (see for example (Mehler et al., 2010)). More recently, the development of genre-dependent models for a variety of natural language processing (NLP) tasks such as parsing (Ravi et al., 2008; McClosky et al., 2010; Roux et al., 2012), speech recognition (Iyer and Ostendorf, 1999), word sense disambiguation (Martinez and Agirre, 2000), and machine translation (Wang et al., 2012) has been found to significantly improve performance. The ability to match documents by genre has also become important for collecting data to train language models for spoken language understanding, given the difficulty of creating large repositories of transcribed spoken language corpora (Bulyko and Ostendorf, 2003; Sarikaya et al., 2005).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 565–576, Dublin, Ireland, August 23-29 2014.

While the utility of document characterization by genre for empirical language analysis is widely acknowledged, there is relatively little agreement on methodology. In part, this stems from the difficulty of providing a comprehensive list of genres or even an operational definition of what constitutes a distinct genre, much less a definitive set of features to characterize genre differences. The earliest large-scale statistical study of genre is that of Biber (Biber, 1988), who applied principal component analysis (PCA) to a one-million word corpus consisting of heterogeneous varieties of spoken and written discourse in order to identify multiple dimensions of variation in language. Biber argued that linguistic variation was continuous along six dimensions: involved vs. informational, narrative vs. non-narrative, explicit vs. situation-dependent reference, overt expression of persuasion, abstract vs. non-abstract information, and on-line information elaboration; he identified features associated with each dimension, and characterized kinds of discourse by joint assessment of similarities and differences across these dimensions. Interestingly, since Biber's study, there has been comparatively little investigation of how genres vary using multivariate distributional methods (see, for example, the discussion in (Kilgarriff, 2001)).

Biber's work, which was completed in the mid-1980's, relied on a large number of features extracted using somewhat *ad hoc* methods and reported no reliability measures. Given the renewed interest in genre classification and the increasing interest in automatic techniques to adapt NLP tools across different kinds of corpora, we feel it is worth subjecting Biber's thesis to a new test, utilizing state-of-the-art methods for extracting features from a high quality, very heterogeneous corpus. In addition to replicating Biber's basic approach with more reliable features, we include newer genres (e.g., email, blogs, tweets) in an attempt to verify that these methods can generalize over different kinds of data. We use a smaller feature set that overlaps with Biber's for the most part, but which also includes features unavailable in the earlier work. In our set, each feature was identified using freely available NLP tools and was manually validated. In our use of different features, our experiment constitutes a strong test of Biber's claim that the dimensions of variation he identified arise from underlying constraints on usage. We find three components similar to his, and a new one he did not find, based on our use of Named Entity features. We find that genres that are separable on one component are often co-extensive on another. To quantify the distinctiveness of each of the genres relative to the others, we use a metric that has previously been used to measure separability of classes.

2 Related work and motivation

Our work builds on Biber's 1988 study, but differs in the corpus and features used. Biber's corpus and MASC (Ide et al., 2010), the corpus used in our study, differ in source language (British English versus American English), time coverage (skewed towards a single year versus three decades), and the situations of use. Biber's corpus was drawn from the Lancaster-Oslo-Bergen (LOB) Corpus of British English, consisting of works published in 1961, the London-Lund corpus of spoken English, consisting of 87 texts of British English from private conversation, public interviews and panel discussions, telephone conversations, radio broadcasts, spontaneous speeches and prepared speeches produced in the 1970s. To these Biber added a collection of his own professional and personal letters. MASC represents a larger time slice (1990s to present) and is more heterogeneous, including a wider range of traditional genres as well as new social media (email, blogs, twitter) and collectively generated fiction (ficlets). We take advantage of MASC's rich set of validated annotations to include features that would not have been (easily) available at the time of Biber's study, and reconsider the use of some features used in his work.

Some work on genre classification contrasts with Biber's approach, which assumes that documents fall discretely into distinct classes or clusters. Genre classification has been treated as a standalone task (Karlgren and Cutting, 1994; Kessler et al., 1997; Feldman et al., 2009; Stamatatos et al., 2000a; Santini, 2004), or combined with topic classification (Rauber and Müller-Kögler, 2001; Lee and Myaeng, 2002). All of these studies assume that documents fall discretely into distinct classes or clusters. These studies vary in their approach to determining the genre of text, either by using corpora with pre-defined classes (Karlgren and Cutting, 1994), manually refining pre-existing classes (Kessler et al., 1997), creating genre classes using annotators, or locating *a priori* classifications (e.g., web product reviews). The feature sets in genre studies have remained rather stable over the past three decades, mostly utilizing word-based

features similar to many of Biber's such as individual lexical items and/or their orthographic characteristics (e.g., contractions), part-of-speech (POS), punctuation (Kessler et al., 1997; Stamatatos et al., 2000b), derivative statistics (e.g., average word/sentence length, ratios among lexical or POS classes), and POS-ngrams (Santini, 2004; Feldman et al., 2009).

Karlgren and Cutting (1994) apply discriminant analysis to pre-defined classes from the Brown corpus using easily identifiable information such as POS counts, type/token ratios, and sentence length. They achieve relatively low accuracy of 52%. Kessler et al. (1997) also use the Brown corpus and classify documents into three facets: brow, narrative, and genre. They extract 55 features, avoiding features at the syntactic level that are computationally expensive to identify, and characterize them as lexical, character-level, and derivative (log ratios and their sums). They achieve nearly 80% accuracy on their six *genre* classes (reportage, editorial, scitech, legal, non-fiction, fiction). Feldman et al. (2009) create a corpus of eight genres of speech and web text and test an approach to factor documents by genre, formality and number of speakers. They achieve accuracy of 55% using quadratic discriminant analysis on a representation consisting of features based on POS tags, words, and punctuation, reduced using PCA. Santini (2004) applies high-dimensional POS trigram vectors to ten BBC genres (four spoken, six written) with Naïve Bayes classification. A document representation using a length-835 vector achieves 82.6% accuracy for 10-fold cross-validation on all 10 genres, and a Kappa agreement of 0.80.

Rauber and Müller-Kögler (2001) apply self-organizing maps (Kohonen, 1995) for both topic and genre clustering, using features typical of readability measures (e.g., sentence and word lengths, punctuation frequency). Lee and Myaeng (2002) address classification of web text and also do simultaneous genre and subject (topic) classification, using a Naive Bayes learner. Tests on seven genres for both English and Korean achieve 0.80 micro-averaged f-measure or 0.87 cosine similarity.

More recent work finds good performance from the use of ngram features for words, characters and part-of-speech (Gries et al., 2009; Kanaris and Stamatatos, 2009; Sharoff et al., 2010). Gries et al. (2009) relies only on word ngrams of various lengths to produce clusters with high maximum average silhouette width, where higher widths represent more homogeneous clusters that are more distinct from one another. They find that trigrams do best. Kanaris and Stamatatos (2009) uses frequently occurring character ngrams without regard to their discriminatory power, and Sharoff et al. (2010) find that character ngrams outperform word and pos ngrams. On benchmark corpora with from 4 to 8 genres, the latter two works achieve accuracies of up to 96-97% on some corpora. They assume that genres can be taken as a given, although Sharoff et al. (2010) note that chance-corrected human agreement on the gold standard is only moderate.

Another strand of investigation addresses genre variation as a requirement for achieving better performance in new domains, as in language modeling for speech applications (Bulyko and Ostendorf, 2003; Sarikaya et al., 2005) or statistical parsers applied to text (Ravi et al., 2008; McClosky et al., 2010; Roux et al., 2012), where downstream applications can include assignment of semantic argument structure. Bulyko and Ostendorf (2003) select web text for class-based n-gram language modeling. They locate relevant documents using queries representative of conversational speech, rather than characterizing the documents as a whole in terms of statistical features, but demonstrate a significant reduction in Word Error Rate (WER) for their enhanced language models. Sarikaya et al. (2005) achieve even higher improvements using a similar query methodology, then use BLEU scores, a machine translation similarity method (Papineni et al., 2002), to find sentences that are closest to a domain sample. Ravi et al. (2008) propose a method to predict parser accuracy based on properties of the new domain of interest and properties of the domain on which the parser was trained. Lexical features for words other than the 500 most frequent were found to generalize less well than features such as POS and sentence length. Subsequent work models corpus differences using regression models to predict parser accuracy McClosky et al. (2010), or incorporates explicit genre classifiers Roux et al. (2012).

In our initial exploration of genre variation in MASC, we exploited a set of features that subsume most of those discussed in the works reviewed above. We applied a variety of methods, including k-means clustering, discriminative classifiers such as Naïve Bayes, and PCA. Through comparison of results, we discovered that classification had variable performance, and that PCA provided an explanation: docu-

Genre	Code	No. words	Pct corpus
Court transcript	CT	30052	6%
Debate transcript	DT	32325	6%
Email	EM	27642	6%
Essay	ES	25590	5%
Fiction	FT	31518	6%
Gov't documents	GV	24578	5%
Journal	JO	25635	5%
Letters	LT	23325	5%
Newspaper	NP	23545	5%
Non-fiction	NF	25182	5%
Spoken	SP	25783	5%
Technical	TC	27895	6%
Travel guides	TG	26708	5%
Twitter	TW	24180	5%
Blog	BG	28199	6%
Ficlets	FC	26299	5%
Movie script	MS	28240	6%
Spam	SM	23490	5%
Jokes	JK	26582	5%
TOTAL		506768	

(a) Genre distribution in MASC

Annotation type	No. words
Logical	506659
Token	506659
Sentence	506659
POS/lemma (GATE)	506659
POS (Penn)	506659
Noun chunks	506659
Verb chunks	506659
Named Entities	506659
FrameNet	39160
Penn Treebank	506659
Coreference	506659
Discourse structure*	506659
Opinion	51243
TimeBank	*55599
PropBank	88530
Committed Belief	4614
Event	4614
Dependency treebank	5434

(b) Summary of MASC annotations

Figure 1: Composition of the Manually Annotated Sub-Corpus

ments from distinct classes often fell within an identifiable region on one or more dimensions discovered by PCA, but these regions overlapped one another along other dimensions. We concluded that whether or not a set of documents can be categorized into relatively distinct classes by their linguistic forms rather than content depends on how the documents are selected, how the classes are defined, and what features are used. Our goal here is to refine a method to learn key dimensions of variation relevant for the same types of applications referenced in work on genre identification, as discussed in Section 7.

3 Corpus and data preparation

MASC is a 500,000 word corpus of post 1990s American English comprised of texts from nineteen genres of spoken and written language data in roughly equal amounts, shown in Figure 1a). Roughly 15% of the corpus consists of spoken transcripts, both formal (court and debate) and informal (face-to-face, telephone conversation, etc.); the remaining 85% covers a wide range of written genres, including social media (tweets, blogs). The annotation types and coverage in MASC are given in Figure 1b); all MASC annotations are hand-validated or manually produced. The corpus is fully open and freely available.[1]

To prepare the data, we developed a framework in Groovy[2] (a dialect of Java) to extract linguistic features, using version 1.2.0 of the GrAF API[3] to access the MASC data and annotations. Most texts in MASC comprise complete discourse units, e.g. full conversations, letters, chapters from a book, etc., with the exception of tweets, jokes, and (to some extent) ficlets.[4] As shown in Figure 1a), although each MASC genre contains roughly 25,000 tokens, the number of texts in any given genre varies widely, from as few as two to over 100. To standardize the number of data points per genre, the texts in each genre were concatenated and then divided into samples of even length, rounded to the nearest sentence boundary. Portions of the texts containing email headers, bibliographic references, and computer code, which contain an excess of certain punctuation and other special characters, were eliminated prior to creating the samples.

Initially, we created sample sets consisting of 1,000 tokens per sample,[5] motivated by Biber's observation that even rare linguistic features are relatively stable across samples of this size (Biber, 1993). Our

[1] MASC is downloadable from http://www.anc.org/data/masc and available from the Linguistic Data Consortium (LDC).

[2] http://groovy.codehaus.org

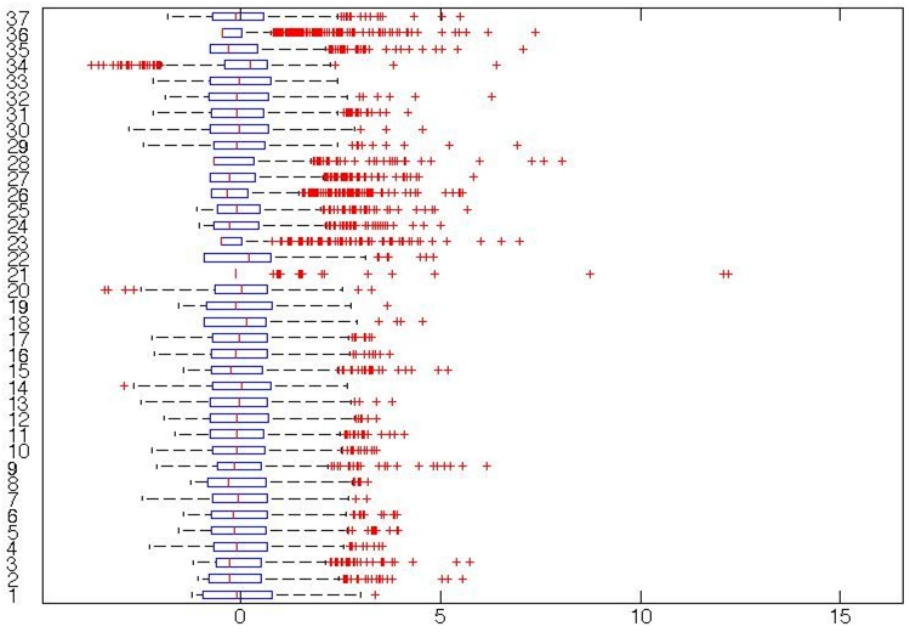
[3] http://sourceforge.net/projects/iso-graf/

[4] Ficlets are story fragments to which "prequels" or "sequels" are added by online participants.

[5] We use tokens as the unit of analysis rather than blank-separated words (strings), which, given the MASC tokenization strategy, means that hyphenated words such as "able-bodied" and possessive markers (*'s*) are treated as individual tokens.

1	1st/2nd person pro.
2	3rd person pro.
3	Pronoun *it*
4	Copula verbs
5	All NEs
6	NEs w/o date
7	Verbs, base
8	Verbs, past
9	Gerunds/Pres. ptp.
10	Past ptp.
11	1st/2nd pres. sg. V
12	3rd pres. sg. V
13	Common nouns
14	All verbs
15	Proper nouns
16	Adjectives
17	Adverbs
18	Superlatives
19	All pers. pro.
20	Prepositions
21	Foreign words
22	Exist. *there*
23	Interjec.
24	NEs, person
25	NEs, date
26	NEs, location
27	NEs, org.
28	Suasive verbs
29	Stative verbs
30	Noun chunk length
31	Verb chunk length
32	Tokens/sentence
33	Characters/token
34	Periods
35	Questions
36	Exclamations
37	Commas

(a) Thirty-seven features



(b) Boxplots of the 37 features: the box shows the range of the 25th to 75th percentiles with the median value identified by the vertical red bar. The black whiskers show the extreme values not considered outliers, and the red are the outliers. The most extreme outliers of feature 21 were dropped to save space.

Figure 2: Feature names and boxplots

experiments showed, however, that for the features used here, results were comparable using 500-token chunks, which enabled us to work with a set of data points of the same size as Biber's. Our process generated 965 500-token chunks, with roughly 50 chunks per genre.

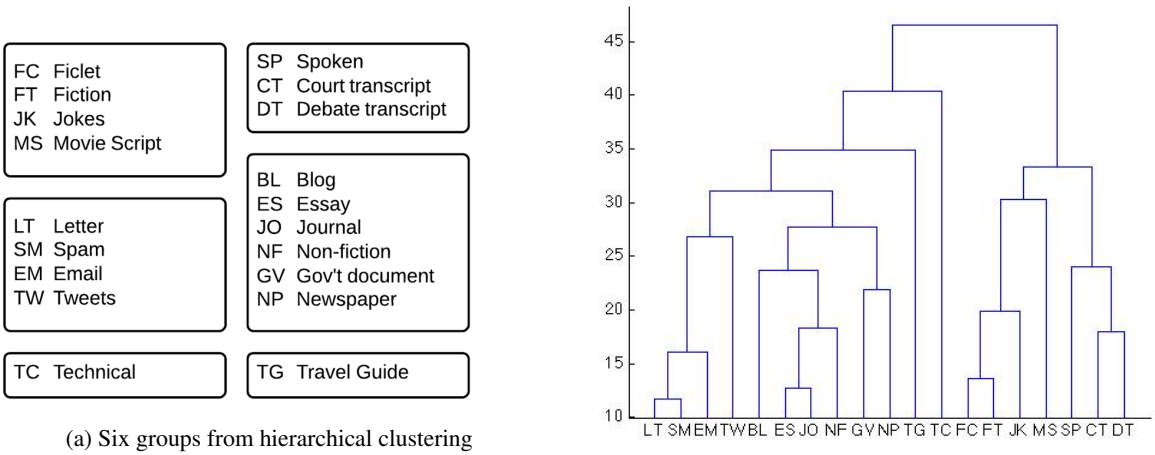
4 Features and feature analysis

Biber used sixty-seven features consisting primarily of lexical items and groups, parts of speech, and quasi-syntactic features such as coordination, negation, relative pronoun deletion, *that*-clauses, and so on. Many of the features in our set overlap with Biber's, but we also exploit annotations in MASC to provide additional features. All the MASC annotations have been manually validated, including those produced by automated tools such as POS-taggers, NE recognizers, and shallow parsers.

PCA is appropriate for data with normally distributed values and can be used to reduce the number of features to include only those that are the least correlated. It highlights features with the greatest variation. Figure 2b) shows boxplots of thirty-seven features we began with. These are mainly frequencies normalized by the total token count in the document samples we created. They also include the average characters per word, and average tokens per sentence, noun chunk, and verb chunk. Figure 2a) lists the features by number. Features 21, 23, 28 and 36, which are foreign words, interjections, suasive verbs and exclamations, have median values (red line within the box) near the 25th percentile, so are highly skewed. We therefore dropped these and carried out the PCA with the remaining thirty-three.[6]

Hierarchical clustering of the dataset by MASC genre yields the dendogram in Figure 3. We used the city block metric (also known as taxicab distance), which is similar to Euclidean distance but less sensitive to outliers. The legend identifies six major clusters for the 19 genres, with two singletons (Travel guides and Technical documents), a cluster with three spoken genres (Court and Debate transcripts, and transcripts of face-to-face and telephone conversations), two four-genre clusters, and one six-genre

[6]To insure comparability of feature influence, all our features were re-scaled in [-1,1] with mean 0.



(a) Six groups from hierarchical clustering

(b) Hierarchical clustering

Figure 3: Hierarchical clustering of 19 MASC genres

cluster. These larger clusters consist of "story-telling" genres (ficlets, fiction, jokes and movie scripts), offline-interactive genres (letters, spam, email and tweets), and discursive text (blog, essay, journal, non-fiction, government documents, and news). Thus the distribution of our features across the data predict groupings that correspond well with our intuitions about the genres defined in MASC, providing some justification for both our feature selection and the genre assignments in the corpus. The groupings also reflect several of Biber's dimensions of variation, as discussed in Section 7.

Here, we describe PCA in general terms to present four principal components identified in our analysis. We focus on features associated with the components, and on the six MASC document clusters.

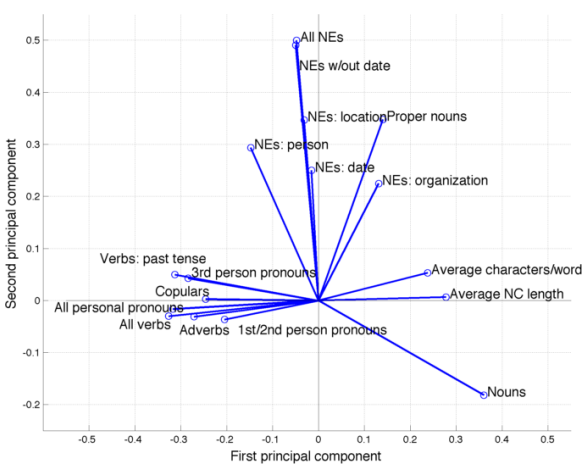
PCA starts with a covariance matrix of all features: a square matrix where each cell value is the covariance of feature x_i with feature x_j for $i, j \in M$. Covariance of x_i, x_j is analogous to variance: for all datapoints $n \in [1 : N]$, you subtract x_{i_n} from $\overline{x_i}$, x_{j_n} from $\overline{x_j}$, sum the products of these differences, and normalize by n-1.[7] A common explanatory visualization will show a scatterplot of hypothetical data values in a sausage shape at a diagonal to the x-axis. A line along the maximum width of the sausage represents the dimension of greatest variation. A second axis can be placed orthogonal to this first component; it will account for less of the variance in the data, and in a different direction. PCA consists of computation of these axes (eigenvectors) from a covariance matrix.

5 PCA results

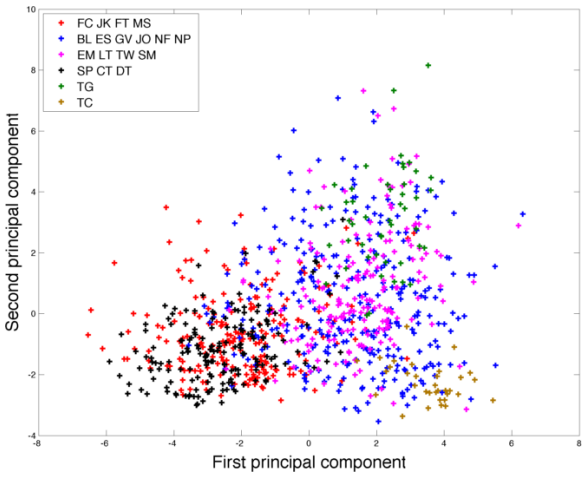
Figure 4a) shows a plot of our first principal component by the second component and the features that contribute most to each, based on the features' loadings (weights) on the new components. The components are rotated to become the new x,y axes and centered at zero. Projection of the individual features onto the rotated axes shows which features contribute most directly to each dimension. Figure 5a) shows a similar plot for the third and fourth components. Twenty-seven features have loadings of at least 0.2 on any component. Many have similar loadings (e.g., commas and prepositions on the fourth component), indicating the data could be represented with fewer, uncorrelated features.

Past tense verbs, copula verbs, personal pronouns, and adverbs load heavily on one pole of the first principal component, while characters per word, noun chunk length and nouns load higher on the opposite pole. This component corresponds rather well to Biber's first component, which had similar loadings for personal pronouns, adverbs, nouns and word length, and which he interpreted as *involved versus informational*–i.e., interactive, unplanned, primarily spoken data vs. polished written documents conveying (sometimes dense) information about a given topic.

[7] See any text on covariance for an explanation of why n-1 is a better normalization term than n.

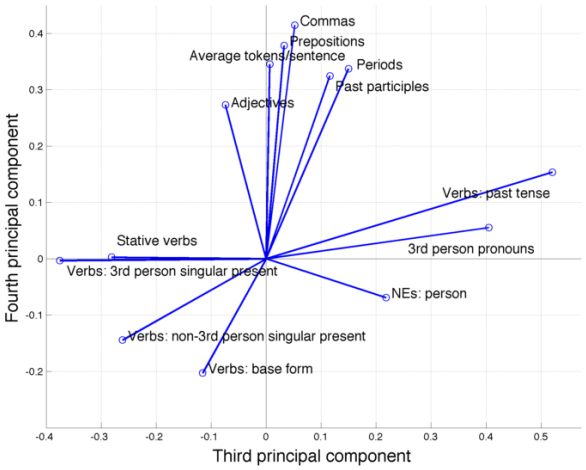


(a) First and second principal components

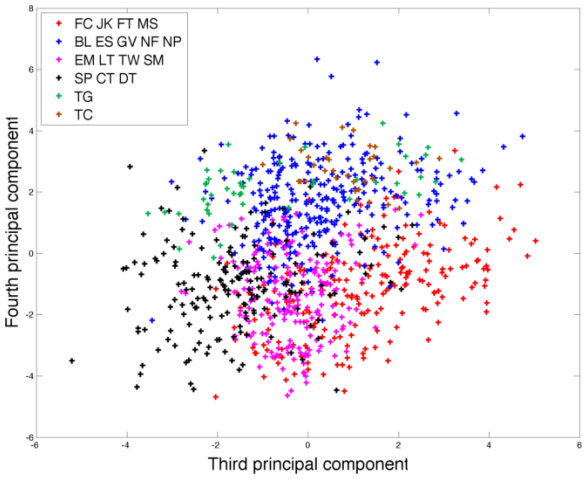


(b) Document regions for components one and two

Figure 4: First and Second Principal Components



(a) Third and fourth principal components



(b) Document regions for components three and four

Figure 5: Third and Fourth Principal Components

Our second principal component is defined almost entirely by the contrast between NEs and common nouns. It corresponds to none of Biber's components; he had no NE features. Our third component has loadings from 3rd person present tense verbs (and other verb forms) at one end, and past tense verbs, third person pronouns, and person NEs at the other. It corresponds to Biber's second component, which had similar loadings for past tense verbs and third person pronouns, and somewhat less for present tense verbs. He interpreted this dimension as representing the variation from non-narrative to narrative.

Our fourth component corresponds to Biber's fifth, which he characterized as abstract versus non-abstract. At one extreme we have commas, prepositions, sentence length (in tokens) and past participles, with base verbs loading to some degree on the other extreme. The features loaded on Biber's fifth component were conjuncts, which might correlate with longer sentence length, past participles, and agentless passives. In the corresponding scatterplots (Figures 4b and 5b), each datapoint (document chunk) has been color-coded according to the six clusters found in the preceding section. There are clearly distinct regions along the first component for spoken interactions (black), story telling (red), offline interaction (pink) and discursive (blue), but with a great deal of overlap. Travel guides (green) and technical (gold) are at the blue extreme, but at different locations along the second dimension. Moving from left to right in Figure 4b), each next color has greater dispersion along the second component, apart from green and gold, which have clearly separate locations from each other, at the top and bottom,

	Story telling	Discursive	Offline Interaction	Spoken Interaction	Travel Guide	Technical
Story Telling	0.00	0.23	0.13	<u>0.06</u>	0.63	0.91
Discursive	0.23	0.00	0.21	0.24	0.15	0.35
Offline Interaction	0.13	0.21	0.00	0.07	0.57	**1.07**
Spoken Interaction	<u>0.06</u>	0.24	0.07	0.00	0.68	0.88
Travel Guide	0.63	0.15	0.57	0.68	0.00	0.78
Technical	0.91	0.35	**1.07**	0.88	0.78	0.00

Table 1: Mean Bhattacharyya Distance of all Genre Pairs using PCA Scores

respectively. In Figure 5b), the overall dispersion is more even across both dimensions, with separate centers for each of the four major colors (black, pink, red and blue), but again without sharp separation.

6 Genre Distance Measurement

A metric that summarizes how separable a pair of genres are in the defined PCA space would be more convenient than the visualizations in Figures 4b and 5b. Bhattacharyya distance, which measures the similarity of two discrete or continuous probability distributions, has been used in image segmentation and signal selection, to minimize the probability of misclustering for segmentations (Coleman and Andrews, 1979), or the probability of misclassifying different signals (Kailath, 1967). Here we illustrate its use in summarizing the separability of a pair of genres across the four principal components.

In statistics, the Bhattacharyya distance measures the similarity of two discrete or continuous probability distributions. It is closely related to the Bhattacharyya coefficient, which measures the amount of overlap between two statistical samples or populations.

The Bhattacharyya coefficient for two continuous probability distributions $p(x)$ and $q(x)$ is:

$$\text{Bhattacharyya coefficient} = \rho = \int_C \sqrt{q(x)p(x)}dx$$

Where C is the domain of probability density $p(x)$ and $q(x)$. The Bhattacharyya coefficient takes on values in [0,1]. Bhattacharyya distance maps the Bhattacharyya coefficient to [0,∞]:

$$\text{Bhattacharyya distance} = B = -\ln \rho$$

We take the mean Bhattacharyya Distance of a pair of genres across all four components as a summary measure of seprability. As an illustration, consider the two clusters of offline interaction (pink) and discursive text (blue) from Figures 4b) and 5b). Their Bhattacharyya Distances on the first through fourth components, using the PCA scores, are: 0.05, 0.01, 0.14, 0.63. They have the largest distance on the fourth component, the axis of abstract vs non-abstract, which is consistent with the visualizations. The summary statistic is then the mean of the four individual distances: 0.21.

Table 1 gives the mean Bhattacharyya Distance of each pair of genres for the four components. The pair of genres that is the closest on all four components is story telling and spoken interaction (0.06; underlined). The pair that is the most distant on all four components is technical and offline interaction (1.07; in bold). Bhattacharyya Distance can also be computed for each pair of genres using the original normalized feature values. In three cases the Bhattacharyya Distance in the PCA space is the same as in the original feature space, but in all other cases the Bhattacharyya Distance is much greater.

7 Discussion

Strong patterns of similarity in dimensions of variation across many genres of English emerge from our comparison with Biber's study, despite differences in the features used, the contrast between American and British English, and the use of new media types. The results support the view that relatively stable dimensions of variation arise from properties of the situations of use across varieties of English. This applies as well to genres that did not exist in Biber's time (email, twitter, spam), which group with the interactive genre included in Biber's corpus (letters) and are similar to other offline discourse despite representing an interactive form–albeit an "offline interactive" form–of discourse.

A significant departure from Biber's results concerns the component defined primarily by Named Entities (NEs), which emerges as the second strongest dimension of variation in our study. This demonstrates that additional features–in particular, features beyond those based on orthographic and morpho-syntactic properties that have figured in most genre studies to date–can dramatically impact Biber's original model and extend the range of properties that can characterize particular text types. It also suggests that higher-level linguistic properties and other more complex features can contribute substantially to genre characterization and discrimination, a topic we plan to pursue in the future.

In what follows, we discuss similarities and differences in the two PCA analyses, the conclusions this leads to regarding the feasibility of genre classification, and ways in which the analysis can support retrieval, language modeling, and domain adaptation.

Our first principal component is very similar to Biber's first factor, which he interpreted as differentiating situations of use with more of an informational focus from those with an interactive or affective function. In addition, he noted a contrast between *online* and *offline* production–i.e., spoken vs. written production modes. The heavily loaded features the two analyses have in common are consistent with the interpretation: 1st/2nd person pronouns, many verb features, and adverbs are at one pole, with word length and nouns at the other. He claimed that this distinction *is obviously a very powerful factor . . . not an artifact of the factor extraction technique*, meaning that it arises from differences between the demands of face-to-face, online interaction and those of offline, expository discourse. Having found a very similar dimension using different (correlated) features, we agree with this claim. Figure 4b) shows that the spoken interaction documents in MASC fall on the "involved" side of this dimension, while expository texts fall on the "informational" side.

Interestingly, the genres that did not exist in Biber's time (email, twitter, spam) group with the interactive genre included in Biber's corpus (letters), and they are similar to other offline discourse despite representing an interactive form–albeit an "offline interactive" form–of discourse. This provides a strong argument for the validity of the first component and its link to underlying situational factors of language use. In Figure 4b), the hypothetical centroid of the pink (offline interactive) region seems somewhat less to the right on the x-axis than a corresponding centroid for the blue (expository) set, but the pink and blue are relatively co-extensive, and in particular, are clearly separated from both the black (face-to-face online interaction) and red (storytelling) genres. This makes intuitive sense, as storytelling genres often depict face-to-face interaction ("so the elephant says to the camel"), and therefore mimic its immediacy.

Our second principal component is defined primarily by Named Entities (NEs), which has no correlate in Biber's study; his features included proper nouns but not NEs. Person NEs load with past tense verbs and third person pronouns on our third component, which resembles Biber's narrative dimension. Most of the MASC genres seem to be dispersed all along our second dimension, suggesting that NE frequency varies across texts in these genres; the exception is travel guides, which consistently include larger numbers of NEs. The explanation here is less on production constraints than on function, as travel guides survey geographical points of interest, historical monuments and persons, hotels and restaurants, and so on.

As noted in Section 5, our third component is very similar to Biber's second (narrative versus non-narrative), and our fourth is somewhat similar to Biber's fifth (abstract versus non-abstract). Note that the fourth dimension shows a greater separation of expository (blue) and offline-interactive (pink) genres, which substantially overlap on the first dimension. This provides a good example of how the 4-dimensional visualization provided by the scatterplots reveals potentially very different relations among genres across the components, which in turn explains why fixed definitions of genre are difficult, if not impossible, and why genre classification can be hard to achieve. We observe that the genre classes can be more or less separable on one dimension but not another. As another example, travel guides and technical documents are at distinct locations on the second component, but span the same locations on the first.

This lack of separability on one or more dimensions is true for nearly all pairs of our six genre classes, as well as for any pair of dimensions. This suggests that an application that requires genre classification could use PCA to find dimensions of variation that lead to the best separation, and summarize the separability using the mean Bhattacharyya distance. As the number of genres one needs to classify increases,

it could be that the number of orthogonal dimensions required to lead to the best separation might also increase. In Table 1, for example, with the exception of the row for Discursive Text, all rows have at least one cell with a value close to or above 0.80, indicating that each of the six genres can be clearly separated from at least one other genre. We would predict that Discursive Text would be the most difficult to classify using genre features alone.

The strong similarities among the major components in Biber's study and ours support the view that genre variation is continuous along multiple dimensions due to contextual properties such as cognitive constraints, interactivity, and function. As such, we view the dimensions as arising from observable properties of discourse situations. Given a new genre, it should be possible to predict where it would be located in the PCA space defined here. We would predict that chats, for example, would pattern more closely with face-to-face interaction than with offline interactive genres. The same methodology could be applied to a sub-genre, such as the discursive texts, to discover more specific dimensions to differentiate among them.

Because language use changes over time, and new genres arise, we do not view the 4-dimensions as a definitive representation of genre space. We do, however, envision a concrete application of this particular representation, namely to measure corpus similarity in a multivariate fashion. Because our PCA analysis makes it possible to locate new documents in the defined space, it would be possible to identify which MASC documents a new set of documents is most similar to. PCA scores could be computed on the four dimensions for corresponding features in the new documents. This approach could be used in any application where it is desirable to find similar documents, such as retrieval, language modeling, or domain adaptation. For example, in recent work on domain adaptation of parsers, McClosky et al. (2010) present a confusion matrix with six corpora to demonstrate how performance of a Charniak parser (Charniak, 2000) varies depending on which corpus it is trained on. They assume that a new target domain will be a mixture of their six source domains and build a simple regression (three features) to predict which of the six parsers will perform best on a new corpus. They subsequently state that an alternative approach could use a high-dimensional vector space to compare corpora. Inspired by this suggestion, we are currently developing a web service that will allow researchers to locate their corpora in the 4-dimensional space identified in this study, and to compute the values of their PCA scores. This would make it possible to use Bhattacharrya distance as described in Section 6 to measure the similarity of corpora in genre space, which could be quite relevant for adapting parsers or other NLP tools. This contrasts with the similarity measures used in Ravi and Knight (Ravi et al., 2008) and McClosky (McClosky et al., 2010), which are based on lexical features.

8 Conclusion

Using a relatively small set of under three dozen features to represent the linguistic forms in discourse, PCA reveals four principal components of variation in a very heterogeneous corpus of post 1990s American English that are comparable to those identified in Biber's work, as well as additional dimensions based on features not included in that earlier study. Six genres derived from the MASC corpus using hierarchical clustering are separable on some but not all components. These differences in separability potentially explain the variations in performance across different works that do genre classification. The resulting 4-dimensional genre space provides a basis for more detailed analysis of sub-genres, for a better understanding of the relation between genre and situations of language use, and for predicting the distributional properties of new genres. In future work, we plan to build on this basis to develop an increasingly detailed and, at the same time, generalizable characterization of genre.

Our results depict a *big picture* for how discourse in English varies with respect to style or form, and how different genres are conditioned by aspects of the situations of language use. We believe that exploration of genre in these terms can provide a more viable approach to measuring distinctions among texts than the approach used in most recent work, and can provide a more informed basis to incorporate genre distinctions in information retrieval, language modeling, and domain adaptation for statistical NLP.

Acknowledgements

This work was supported in part by NSF CRI-1059312.

References

Douglas Biber. 1988. *Variation across Speech and Writing*. Cambridge University Press, Cambridge, UK.

Douglas Biber. 1993. The multi-dimensional approach to linguistic analyses of genre variation: An overview of methodology and findings. *Computers and the Humanities*, 26:331–345.

Ivan Bulyko and Mari Ostendorf. 2003. Getting more mileage from web text sources for conversational speech language modeling using class-dependent mixtures. In *Proc. HLT-NAACL 2003*, pages 7–9.

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

Guy Barrett Coleman and Harry C Andrews. 1979. Image segmentation by clustering. *Proceedings of the IEEE*, 67(5):773–785.

Sergey Feldman, Marius Marin, Julie Medero, and Mari Ostendorf. 2009. Classifying factored genres with part-of-speech histograms. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 173–176, Boulder, Colorado, June. Association for Computational Linguistics.

Stefan Th. Gries, John Newman, Cyrus Shaoul, and Philip Dilts. 2009. N-grams and the clustering of genres. Paper presented at the workshop on Corpus, Colligation, Register Variation at the 31st Annual Meeting of the Deutsche Gesellschaft fr Sprachwissenschaft.

Nancy Ide, Collin Baker, Christiane Fellbaum, and Rebecca Passonneau. 2010. The Manually Annotated Sub-Corpus: A Community Resource for and by the People. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 68–73, Uppsala, Sweden, July. Association for Computational Linguistics.

Rukmini Iyer and Mari Ostendorf. 1999. Relevance weighting for combining multi-domain data for n-gram language modeling. *Computer Speech & Language*, 13(3):267–282.

Thomas Kailath. 1967. The divergence and Bhattacharyya distance measures in signal selection. *Communication Technology, IEEE Transactions on*, 15(1):52–60.

Ioannis Kanaris and Efstathios Stamatatos. 2009. Learning to recognize webpage genres. *Information Processing and Management*, 45(5):499–512, September.

Jussi Karlgren and Douglass Cutting. 1994. Recognizing text genres with simple metrics using discriminant analysis. In *Proceedings of the 15th Conference on Computational Linguistics - Volume 2*, COLING '94, pages 1071–1075, Stroudsburg, PA, USA. Association for Computational Linguistics.

Brett Kessler, Geoffrey Nunberg, and Hinrich Schütze. 1997. Automatic detection of text genre. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, ACL '98, pages 32–38, Stroudsburg, PA, USA. Association for Computational Linguistics.

Adam Kilgarriff. 2001. Comparing corpora. *International Journal of Corpus Linguistics*, 6(1):1–37.

Teuvo Kohonen. 1995. *Self-organizing Maps*. Springer-Verlag, Berlin.

Yong-Bae Lee and Sung Hyon Myaeng. 2002. Text genre classification with genre-revealing and subject-revealing features. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 145–150, New York, NY, USA. ACM Press.

David Martinez and Eneko Agirre. 2000. One sense per collocation and genre/topic variations. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, pages 207–215, Stroudsburg, PA, USA. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 2010. Automatic domain adaptation for parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 28–36, Stroudsburg, PA, USA. Association for Computational Linguistics.

A. Mehler, S. Sharoff, and M. Santini. 2010. *Genres on the Web: Computational Models and Empirical Studies*. Text, Speech and Language Technology. Springer.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Andreas Rauber and Alexander Müller-Kögler. 2001. Integrating automatic genre analysis into digital libraries. In *First ACM-IEEE Joint Conference on Digital Libraries*, pages 1–10.

Sujith Ravi, Kevin Knight, and Radu Soricut. 2008. Automatic prediction of parser accuracy. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 887–896, Honolulu, Hawaii, October. Association for Computational Linguistics.

Joseph Le Roux, Jennifer Foster, Joachim Wagner, Rasul Samad, Zadeh Kaljahi, and Anton Bryl. 2012. DUC-Paris13 systems for the SANCL 2012 shared task.

Marina Santini. 2004. A shallow approach to syntactic feature extraction for genre classification. Technical Report ITRI-04-02, Information Technology Research Institute, University of Brighton. Also published in Proceedings of the 7th Annual Colloquium for the UK Special Interest Group for Computational Linguistics, Birmingham, UK.

Ruhi Sarikaya, Agustín Gravano, and Yuqing Gao. 2005. Rapid language model development using external resources for new spoken dialog domains. In *International Congress of Acoustics, Speech, and Signal Processing (ICASSP)*, pages 573–576, Philadelphia, PA, USA. IEEE, Signal Processing Society.

Serge Sharoff, Zhili Wu, and Katja Markert. 2010. The web library of Babel: evaluating genre collections. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).

Efstathios Stamatatos, Nikos Fakotakis, and George Kokkinakis. 2000a. Text genre detection using common word frequencies. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 2*, COLING '00, pages 808–814, Stroudsburg, PA, USA. Association for Computational Linguistics.

Efstathios Stamatatos, George Kokkinakis, and Nikos Fakotakis. 2000b. Automatic text categorization in terms of genre and author. *Computational Linguistics*, 26(4):471–495, December.

Wei Wang, Klaus Macherey, Wolfgang Macherey, Franz Och, and Peng Xu. 2012. Improved domain adaptation for statistical machine translation. In *AMTA-2012*.

Nigel G. Ward and Steven D. Werner. 2013. Using dialog-activity similarity for spoken information retrieval. In Frédéric Bimbot, Christophe Cerisara, Cécile Fougeron, Guillaume Gravier, Lori Lamel, François Pellegrino, and Pascal Perrier, editors, *14th Annual Conference of the International Speech Communication Association, Interspeech*, pages 1569–1573. ISCA.

Cross-lingual Discourse Relation Analysis: A corpus study and a semi-supervised classification system

Junyi Jessy Li[1], **Marine Carpuat**[2] **and Ani Nenkova**[1]
[1] University of Pennsylvania, Philadelphia, PA 19104, USA
{ljunyi, nenkova}@seas.upenn.edu
[2] National Research Council Canada, Ottawa, ON K1A 0R6, Canada
marine.carpuat@nrc.gc.ca

Abstract

We present a cross-lingual discourse relation analysis based on a parallel corpus with discourse information available only for one language. First, we conduct a corpus study to explore differences in discourse organization between Chinese and English, including differences in information packaging, implicit/explicit discourse expression divergence, and discourse connective ambiguities. Second, we introduce a novel approach to learning to recognize discourse relations, using the parallel corpus instead of discourse annotation in the language of interest. Our resulting semi-supervised system reaches state-of-art performance on the task of discourse relation detection, and outperforms a supervised system on discourse relation classification.

1 Introduction

The analysis of the way spans of text semantically connect with each other to create a coherent text has a rich theoretical and empirical tradition (Mann and Thompson, 1988; Marcu, 1997; Di Eugenio et al., 1997; Allbritton and Moore, 1999; Schilder, 2002). Because of the difficulty in annotation, however, labelled datasets were rare and rather small.

The release of the Penn Discourse Treebank (PDTB) (Prasad et al., 2008) brought about a new sense of maturity in discourse analysis, finally providing a high-quality large-scale resource for training discourse parsers for English. Based on the PDTB, a number of studies have provided insightful analysis of the use of discourse connectives in English news text and have developed methods for the identification of discourse relations and their arguments (Wellner and Pustejovsky, 2007; Pitler et al., 2008; Pitler and Nenkova, 2009; Pitler et al., 2009; Lin et al., 2009; Prasad et al., 2010; Park and Cardie, 2012; Lin et al., 2014). Some have applied the insights and classifiers to standard natural language processing tasks such as assessing text coherence and text quality (Pitler and Nenkova, 2008; Lin et al., 2011), detecting causal dependencies of events (Do et al., 2011), and machine translation (Meyer and Popescu-Belis, 2012).

A resource like the PDTB is extremely valuable, and it would be desirable to have a similar resource in other languages as well. Following the release of the PDTB, smaller corpora annotated with discourse relations have been developed for Hindi (Oza et al., 2009), Turkish (Zeyrek and Webber, 2008), Arabic (Al-Saif and Markert, 2010), and the effort is on-going with Chinese (Zhou and Xue, 2012).

On the other hand, for the vast majority of languages, such well-annotated resource for discourse relations is not available. In our work we carry the valuable annotations in the PDTB over to another language—Chinese—using parallel corpora. Projecting information available in one language onto another has been explored in areas such as part-of-speech tagging (Yarowsky et al., 2001; Das and Petrov, 2011), grammar induction (Hwa et al., 2005; Ganchev et al., 2009) and semantic role labeling (Pado and Lapata, 2005; Johansson and Nugues, 2006; van der Plas et al., 2011). For discourse relations, prior work has shown that a parallel corpus is helpful for disambiguating certain explicit discourse connectives (Meyer et al., 2011). To the best of our knowledge, the work we present here is the first study that directly infers discourse relations using resources only available in another language.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 577–587, Dublin, Ireland, August 23-29 2014.

The goal of our work is not only to measure the accuracy with which discourse relations can be identified in another language without annotations beyond the PDTB, but also to catalog the differences in discourse relation realization across different languages, Chinese and English in our case. We show that the two languages vastly differ in how information is packaged into a sentence, which also leads to differences in the implicit/explicit expression of discourse relations and the ambiguities in discourse connectives. These differences challenge the currently accepted distinctions between syntax and discourse between the two languages for applications such as machine translation. Then we present our semi-supervised learning algorithm to recognize explicit discourse relations in Chinese, relying solely on discourse information available in English. For multiway classification, our system outperforms a supervised system trained on the existing pilot dataset of discourse relations in Chinese (Zhou and Xue, 2012). In the task of binary classification for identifying specific discourse relations, the performance of our system is within 4% accuracy of that of the supervised system for all but one relations.

2 Data

As our parallel corpus, we use the newswire portion of the GALE Chinese-English Word Alignment and Tagging Training corpus (parts 1 and 2). The corpus contains 2,175 newswire articles, corresponding to 6,255 translation segments with 248,999 Chinese characters. These articles were translated into English by human translators. Gold standard word alignments are available for this corpus. A *minimal match* alignment approach (Li et al., 2010) was adopted for creating the gold standard, namely, alignments are between an English word and only the necessary Chinese *characters*. We repurpose this resource created for machine translation research for our cross-lingual discourse analysis. The availability of manual alignments between Chinese discourse connectives and their English translation makes it possible to conduct a reliable analysis by focusing on actual cross-lingual divergences, without noise introduced by potential errors from automatic aligners. [1]

We use a highly accurate supervised classifier for English explicit discourse relations (Pitler and Nenkova, 2009)[2] to automatically annotate the English portion of the GALE parallel corpus. The classifier was trained on the PDTB to identify discourse relations explicitly signaled by a set of 100 discourse connectives such as *however, because, while* or *for example*. For each instance of the 100 words or expressions, the classifier predicts if the expression is used as a discourse connective or if the instance is a non-discourse connective sense of the phrase or word. For each instance predicted to be a discourse connective, the classifier identifies the discourse relation signaled by the connective: TEMPORAL, COMPARISON, CONTINGENCY or EXPANSION. In our work we predict the same five categories for Chinese expressions which can serve as discourse connectives.

For evaluation and the study of discourse connective ambiguities, we use a development set from the Chinese Discourse Treebank (CDTB) (Zhou and Xue, 2012) consisting of 170 documents[3]. In the CDTB, an annotation style similar to the PDTB is applied on the texts from the Chinese Treebank corpus (Xue et al., 2005). For a discourse connective, one of eight discourse relation senses is annotated. All of these classes are subsumed by the four top-level relations in the PDTB. We map them to the PDTB relation senses according to their definitions:

Alternative → Expansion; Causation → Contingency; Conditional → Contingency; Conjunction → Expansion; Contrast → Comparison; Expansion → Expansion; Purpose → Contingency; Temporal → Temporal.

3 Information packaging characteristics

The notion of sentence in Chinese is very different from that in English. Punctuation marks were introduced in the early 20^{th} century; sentences resemble more a collection of related information than structurally well-defined syntactic units as in English. In fact, commas are often ambiguous, signaling

[1] While cross-lingual projection could be directly applied to automatic word alignments, discourse relation analysis raises some specific challenges because the main target of analysis (discourse connectives) are function words, which do not have as much of an impact on the final analysis in applications focusing on content words. As a result, we exclusively use manual alignment links in this study, and will address issues raised by automatic alignments in future work.

[2] The classifier is available at http://www.cis.upenn.edu/∼epitler/discourse.html

[3] This is an on-going annotation project. We are grateful to the authors for providing us with their valuable development set.

	% data	avg-length	std-length
1-many	18.83	61.42	28.85
1-1	81.17	35.73	25.34

Table 1: Percentage, length and standard deviation of sentences for which one Chinese source sentence is translated into one (1-1) or multiple (1-many) English sentences. Length is calculated based on the number of Chinese characters.

either clausal subordination, coordination or end-of-sentence (as construed from an English-centric point of view). Automatic systems have been developed to disambiguate the function of commas (Jin et al., 2004; Xue and Yang, 2011). This is a rather interesting phenomenon for discourse processing, as the English equivalents of Chinese sentences are in fact multi-sentential discourses in English.

The GALE corpus allows us to examine how often this mismatch of discourse organization occurs. Here we look for Chinese source sentences that were translated into multiple English sentences by the human translators. Consider the following example in which the corresponding clauses on both sides are numbered and marked in square brackets:

source [近年来“救灾外交”、“救灾援助”等新名词不断出现]$_1$，[各国围绕救灾问题展开了暗中的竞争与较量]$_2$，[一些国家谋求以救灾为名成立各种国际联盟]$_3$。

ref [In recent years, new phrases such as “disaster relief diplomacy” and “disaster relief aid” have appeared constantly]$_1$. [In relation to the issue of disaster relief, all countries have been silently competing with one another and comparing offerings]$_2$. [Some countries are trying to establish various kinds of international alliance in the name of disaster relief]$_3$.

In this example, the Chinese sentence packed the following related content into a single sentence: the occurrence of the new phrases about disaster relief, the competition among the countries related to disaster relief, and alliances in the name of disaster relief. The phrases expressing this information are separated by the commas in the source Chinese sentence because they are about a single concept “disaster relief”. However, this information needs to be partitioned into three different sentences, each with different subjects, when translated to English.

In the GALE corpus, we identified 1,178 (out of total 6,255) source sentences with reference translations containing more than one sentence. In other words, sentence/discourse mismatch between Chinese and English occurs for 18.83% of the data. Table 1 shows the portion of data involved in such mismatch, with percentage, mean and standard deviation of source sentence length. Not surprisingly, Chinese sentences that require multiple sentences in their English translation are much longer. These long sentences are fairly common, which suggests that the difference in information packaging is highly prevalent and could potentially affect key applications such as machine translation, where systems are trained on a sentence to sentence basis.

We will return to the discussion of this mismatch later, when we discuss how English and Chinese also appear to differ in the way discourse relations are signaled. Briefly, the issue is that relations that are explicit in one language may become implicit in the other, easily inferred by the reader but not marked by a discourse connective. Also, there is an increase in the sense ambiguity of discourse connectives related to EXPANSION relations in Chinese.

4 Implicit and explicit relations

In this section, we present two other differences between the two languages related to discourse organization. One is the need for a discourse relation expressed implicitly in one language to be expressed explicitly in another. The other is the difference of the ambiguity of discourse connectives across the two languages. Before the discussion of these interesting asymmetries, we first present the method for direct projection of discourse relations using the GALE gold standard alignments, which we use to gather a set of explicit discourse connectives in Chinese.

4.1 Direct projection

Thus far we have available a parallel Chinese/English corpus, discourse connectives automatically tagged with their senses on the English side and manual alignments of atomic units between English and Chi-

	Comparison	Contingency	Expansion	Temporal
CH/EN mismatch	63	109	360	195
all	551	469	1198	885
% data	11.43	23.24	30.05	22.03

Table 2: Numbers and percentages of Chinese/English implicit/explicit mismatches.

nese. So for each discourse connective in an English sentence, it is straightforward to identify the corresponding expressions in the Chinese sentence following the gold standard alignments. Then the aligned Chinese expression can be assigned a discourse tag—non-discourse use or one of the four main discourse relation types—which is the same as in the English translation. We call the resulting annotation on the Chinese sentences *discourse projection*.

Further we discard potential expressions of Chinese connectives if they occurred with the same part of speech only once in the entire corpus. The result is a list of a total of 118 Chinese discourse connectives harvested using direct projection.

4.2 Implicit or Explicit?

A discourse relation can be expressed either with an explicit connective (e.g. *however, since*), or implicitly without a connective, in which case the relation would have to be inferred by the reader. Languages may differ in how they express discourse relations.

We investigate such implicit/explicit mismatch using direct projection. Specifically, we study the cases in which an English discourse connective is not aligned to any part of its corresponding Chinese sentence. In this case, the human translator explicitly expressed a discourse relation that was implicitly conveyed in the corresponding Chinese sentence.

The following four examples illustrate a Chinese/English implicit/explicit mismatch for each of the TEMPORAL, COMPARISON, CONTINGENCY and EXPANSION relation, respectively. On the Chinese side we also mark the position of the inserted English connective.

source [当地时间4月27日]$_1$，[阿富汗首都喀布尔举行的抗击苏联入侵胜利阅兵式遭到袭击]$_2$，when$_{\text{TEMPORAL}}$ [阿富汗总统卡尔扎伊和其他政要慌忙撤离现场]$_3$。
ref [On april 27 local time]$_1$, [Afghan president Karzai and other important officials were forced to flee the scene]$_3$ [when$_{\text{TEMPORAL}}$ a military parade in Kabul, Afghanistan commemorating victory in the fight against the soviet invasion was attacked]$_2$.

source [但在目前普遍使用的十种语言中]$_1$，[阿拉伯语仅名列第四位]$_2$，while$_{\text{COMPARISON}}$[英语名列第一]$_3$。
ref [However, of the ten commonly-used languages today]$_1$, [Arabic only ranks fourth]$_2$, [while$_{\text{COMPARISON}}$ English ranks first]$_3$.

source ["中华航空"上海代表处首席代表董大伟告诉记者]$_1$："[现在的两岸包机还不是真正意义上的'直航']$_2$，since$_{\text{CONTINGENCY}}$[还需要经过香港飞行情报区]$_3$。"
ref [Tung Ta-Wei, head representative for China Airlines in Shanghai, told reporters]$_1$, "[presently, the cross-strait charter flights are still not 'direct flights' in the true sense of the term]$_2$, [since$_{\text{CONTINGENCY}}$ they still have to pass through the hong kong flight information region]$_3$. "

source [柳斌杰说]$_1$，[中国出版业下一个发展的重点将是参与国际竞争]$_2$，and$_{\text{EXPANSION}}$[今后双方可就此加大合作力度]$_3$。
ref [Liu Binjie said]$_1$, [a key area of development for the Chinese publishing industry will be participating in international competition]$_2$, [and$_{\text{EXPANSION}}$ in the future the two sides can strengthen their cooperation in this area]$_3$.

The first example is particularly interesting from a discourse point of view as it combines information ordering considerations along with the implicit/explicit expression of discourse relations: not only is the connective *when* missing in Chinese but the two arguments of the connective appeared in reverse order in the English translation of the sentence, with the comma omitted.

In Table 2, we show the numbers and percentages of Chinese/English implicit/explicit mismatches for each relation. We also list the ten connectives that are most frequently associated with the mismatch (i.e., were added to the reference translation), in the format of *connective (# mismatches)* below:

and (341), when (120), while (45), if (37), so that (29), but (23), after (22), so (22), as (21), then (18)

This analysis reveals that the EXPANSION relation is more likely to be implicitly expressed in Chinese, although in other relations this phenomenon is also present.

Connective	Senses	Connective	Senses
而	COMPARISON (7) EXPANSION (2)	又	CONTINGENCY (1) EXPANSION (1)
则	COMPARISON (1) EXPANSION (2)	在...同时	TEMPORAL (3) EXPANSION (2)
如	CONTINGENCY (1) EXPANSION (3)	同时	TEMPORAL (1) EXPANSION (1)

Table 3: Ambiguous Chinese connectives, according to manual annotations in the development CDTB.

A similar mismatch also happens when an English discourse connective is aligned to a punctuation mark in Chinese, illustrated in the following example, where the comma underlined in the source sentence was translated to *and*, thus to an explicit EXPANSION in English:

source [这个总队所属驻边境线中队]$_1$，[大都驻地偏远]$_2$，[自然条件艰苦]$_3$，[信息化建设比较滞后]$_4$。
ref [Most of the contingent's squadrons garrisoned along the border]$_1$ [are stationed in remote areas]$_2$ [where the natural conditions are rough]$_3$ [and$_{\text{EXPANSION}}$ the construction of informatization relatively lags behind]$_4$.

The insertion of the explicit discourse connective *and* makes the use of punctuation between "rough conditions" and "informatization" unnecessary in English. Through our direct projection we found 136 such implicit to explicit transformations with commas and 5 with semicolons. All of them are of the relation EXPANSION, further highlighting the differences in information packaging between the two languages.

4.3 Ambiguity of connectives

Although most of the English discourse connectives identified in the PDTB are not ambiguous, some of the most frequently used ones are (Pitler et al., 2008; Miltsakaki et al., 2008). For example, *while* can signal both TEMPORAL and COMPARISON relations; *since, as* can signal both TEMPORAL and CONTINGENCY relations. Discourse connectives in different languages have different ambiguities; prior work has shown that it is easier to disambiguate the sense of an ambiguous connective when parallel corpora are available (Meyer et al., 2011). The two languages analyzed in Meyer et al. (2011), English and French, are closely related European languages; here we investigate such differences in ambiguities between English and Chinese connectives.

Specifically, using the connectives collected from direct projection, we inspect the relations annotated for these connectives in the Chinese Discourse Treebank development set, and extract connectives such that the majority sense they signal constitutes less than 90% of their total occurrences. Unlike in English where the vast majority of ambiguities are between TEMPORAL and some other sense, we find that all such connectives in Chinese are ambiguous between some relation and EXPANSION. An example of ambiguity between TEMPORAL and EXPANSION is shown below:

source 这样杜伊才能在拿足所有合同内工资的同时$_{\text{TEMPORAL}}$，又乐得清闲，冷眼旁观。
ref Only in this way can Dujkovic sit back and do nothing and look on others disinterestedly when$_{\text{TEMPORAL}}$ getting his full salary per contract.

source 在减少开车出行的同时$_{\text{EXPANSION}}$，还往汽油里掺上从餐馆回收来的食油。
ref While$_{\text{EXPANSION}}$ reducing driving time, they are also mixing gasoline with cooking oil recycled from restaurants.

In the first case, there is a synchrony relation between Dujkovic's "sitting back and doing nothing", and "getting his full salary". In the second case, "reducing driving time" and "mixing gasoline with cooking oil" are a list of methods for saving gasoline.

In Table 3 we list these ambiguous Chinese connectives, their senses and the frequency with which they were annotated. The ambiguities we see here are very different from those in English where the TEMPORAL—CONTINGENCY and COMPARISON—CONTINGENCY ambiguities are most prominent.

5 Predicting discourse relation sense in Chinese

Our analysis so far has revealed considerable differences in the expression of discourse relations in Chinese and English. We now show that projected annotations can be used to disambiguate Chinese discourse connectives despite these differences.

5.1 Learning with unlabeled data

The main idea of learning by projection across parallel corpora is to use a classifier to annotate the English portion of the data, then project the discourse relation sense labels onto the corresponding Chinese sentences. Then a classifier can be trained using features gathered on the Chinese portion of the data.

However, labels gathered from direct projections are not suitable for learning systems without extra processing. If an English connective is aligned to one of the Chinese connectives, we can transfer its label from English to the Chinese connective. However, it is highly likely that a Chinese connective appears in the source sentence but the reference translation used an alternative expression or paraphrase rather than the 100 identified connectives in the PDTB. It is difficult to distinguish through direct projection if an explicit discourse connective in Chinese was expressed implicitly in English or if the Chinese expression was used in a non-discourse sense.

The possibilities described above imply that in our work, we cannot assume that through direct projection we have a fully labeled dataset for discourse connective senses in Chinese. Instead we have a mixture of data with labeled positive examples (when an explicit English connective was aligned to the phrase) and unlabeled examples (where there was no explicit discourse connective in English, so the Chinese expression is either used in a non-discourse sense or is expressed implicitly or using alternative expressions in English, and thus the label is unknown).

Luckily, learning from positive and unlabeled examples, especially for binary classification, is a fairly well studied problem in machine learning (Lee and Liu, 2003; Liu et al., 2003; Elkan and Noto, 2008). We adopt such methods as part of our semi-supervised learning system.

In this work, we propose the following components for relation classification:

(Noisy) data labeling Classify each instance of a possible connective on the English side of the corpus into either *non-discourse use*, or one of TEMPORAL, CONTINGENCY, COMPARISON or EXPANSION. If the English connective signals one of the four relations, transfer the labels to the connectives expressed in the corresponding Chinese sentences through alignments, as described in Section 4.1.

Train sense classifier This classifier is trained only on the Chinese expressions labeled as one of the four main classes of discourse relation. We can train either a binary classifier to predict if a connective expresses a particular relation, or a 4-way classifier which assigns the most probable sense to each connective. The potentially problematic labels for the *non-discourse* class are not used in this stage.

Train discourse use classifier This classifier has to use the potentially problematic data, where we cannot distinguish negative examples from untagged positive examples. The problem is solved as a cascade of classifiers, an approach developed in Elkan and Noto (2008). The idea is to train a noisy classifier that produces a soft score for the data—a probability of being in the class rather than a strict class assignment.

Let y be the *true* discourse use class to be predicted: $y = 1$ for examples of discourse use, and $y = 0$ for examples of non-discourse use. Let l indicate whether the example is *labeled* as discourse use ($l = 1$), or *unlabeled* ($l = 0$, unknown or non-discourse use). First, we use a logistic regression classifier LR to estimate $P(l = 1|y = 1)$. Let's call this estimate e. Using LR, e can be estimated as $\sum_{x \in P} LR(x)/|P|$, where P is the set of the original positively labeled examples, $LR(x)$ is the probability of expression x to be labeled positively. We then use the estimator e to calculate the estimated value of $P(y = 1|l = 0)$, the probability of an expression being discourse use from the original *unlabeled* examples:

$$w = \frac{LR(x)}{e} / \frac{1 - LR(x)}{1 - e}$$

In the second stage, each of the *unlabeled* examples are duplicated, once as a positive example with weight w and once as a negative example with weight $1 - w$. Our second stage classifier—linear-kernel SVM with weights for each example—is trained on the combined set of positive examples (discourse use) and the duplicated version of the unlabeled examples (unknown and non-discourse use class). When w is close to 0.5, the example is practically noise (with labels 0.5 and -0.5) and does not affect the learning of parameters much. Weights closer to 1 practically reassign the originally non-discourse use example to

the discourse use class (labels 1 and 0); a weight close to 0 leaves the example as one of the non-discourse use instances (with labels 0 and -1).

Test phase In testing, first the second-stage SVM model for discourse vs. non-discourse use is applied. For only the expression predicted to be discourse connectives (discourse use), we run the sense classifier to do binary or multiway relation classification. Binary classification labels whether a connective signals a particular relation; multiway classification labels one of the five possible classes: *non-discourse use*, TEMPORAL, COMPARISON, CONTINGENCY and EXPANSION. This series of classifiers results in a system that can assign the same labels as the classifiers trained for English.

To complete our presentation of the approach, we now turn to describe the features used to represent instances of potential discourse connectives.

5.2 Features

The following set of features for each expression we need to classify are extracted solely from the Chinese part of the corpus[4]. The syntactic parse trees were obtained automatically (Levy and Manning, 2003).

Connective The connective expressions themselves. The vast majority of connectives (at least in English) are unambiguous, so using the identity of the connective is a hard-to-beat baseline for sense prediction (Pitler et al., 2008).

Categories The syntactic category of the expression itself, as well as that of its parents, and its left and right siblings (if any). These features are adapted from Pitler and Nenkova (2009).

Depth Depth of the expressions's syntactic category in the parse tree for the sentence.

POS bigram Bigram of part-of-speech tags of the entire sentence.

Production pairs Parent-child node category pairs, gathered from subtrees of two ancestors starting from the parent of the expression's self-category. For example, a subtree IP→NP VP would yield the features (IP NP) and (IP VP). Production rules have shown to be effective for implicit discourse relation classification (Lin et al., 2009; Park and Cardie, 2012). This is a less sparse adaptation of such features.

Punctuation This class corresponds to two features. The first feature takes one of the three possible values: if the expression starts a sentence, if there is a punctuation to the immediate left of the expression, or none of above. The second feature has two values corresponding to whether there is a punctuation to the expression's immediate right.

Sequence pairs Left-to-right sequence pairs of node categories, gathered from subtrees of two ancestors starting from the parent of the expression's self-category. For example, a subtree IP→NP VP PU would yield the features (NP VP) and (VP PU).

Size of ancestor nodes The number of children a node has, calculated with three ancestors starting from the parent of the expression's self-category.

 # characters The number of Chinese characters in the connective expression.

5.3 Classification results

In this section, we demonstrate the effectiveness of learning discourse relations through parallel data projection and semi-supervised learning. We use the GALE corpus for training and the Chinese Discourse Treebank development set (CDTB-dev) for testing. There are 5,136 training instances and 490 testing instances. In addition, we compare performance with 10-fold cross validation results over CDTB-dev. We obtain predictions for each fold and evaluate on the combined data from all folds, instead of averaging performance for each fold. In this way the results from 10-fold validation and those from the semi-supervised classifier trained on projected data are directly comparable. The LIBLINEAR package (Fan et al., 2008) was used for binary classification (including the discourse use classifier[5]), and SVM-Multiclass (Tsochantaridis et al., 2004) with linear kernel was used for multiway classification.

[4] As a reminder, the list of possible connectives was derived from direct projection after pruning items that occurred only once with a particular part-of-speech. There is a total of 118 such expressions for Chinese.

[5] http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/#weights_for_data_instances

	Baseline			Cascade			Supervised		
	A	F	P/R	A	F	P/R	A	F	P/R
connective (C)	67.62	51.23	75.45/38.79	70.29	65.88	66.35/65.42	77.96	75.00	75.00/75.00
C+tree depth	69.39	55.36	77.50/43.06	69.80	66.36	65.18/67.59	78.57	75.86	75.34/76.39
C+categories	66.94	52.63	71.43/41.67	70.82	66.97	66.82/67.13	83.67	82.61	77.87/87.96
C+size of ancestor	67.96	52.57	75.65/40.28	71.22	67.59	67.12/68.06	76.12	72.73	73.24/72.22
C+POS bigram	70.00	58.12	75.56/47.22	74.29	70.42	71.43/69.44	81.84	80.35	76.79/84.26
C+punctuation	67.96	52.85	75.21/40.74	73.67	70.48	69.68/71.30	82.65	80.81	78.85/82.87
above, combined	70.61	60.00	75.00/50.00	**75.10**	**71.63**	**71.96/71.30**	82.04	80.00	78.57/81.48

Table 4: Accuracy, F-measure and precision/recall for classifying discourse/non-discourse use of connective expressions, for top features and for the combined feature set.

	5-way Baseline	Projection	5-way Supervised
connective (C)	0.6332	0.6434	0.6114
C+tree depth	0.5959	0.6367	0.6384
C+punctuation	0.6224	0.6776	0.6425
C+size of ancestor	0.5939	0.6469	0.6073
C+categories	0.5837	0.6633	0.6359
C+POS bigram	0.6469	0.6980	0.6714
above, combined	0.6245	**0.7020**	0.6355

Table 5: Multiway discourse relation classification accuracies, for top and the combined features.

Discourse vs. non-discourse To demonstrate the cascade learning component in our system, we first show results from the intermediate stage of the discourse vs. non-discourse prediction task. We compare three systems: our cascade approach for handling noisy labels for non-discourse use, a baseline trained only on the original noisy non-discourse labels (this corresponds to the hard-label performance of the first stage classifier in our approach) and a supervised system trained on CDTB-dev (where predictions are obtained in 10-fold cross validation fashion).

In Table 4 we show the accuracy, precision/recall and F measure for each system, using connective expressions themselves and the five features that gave the best performance on the test set.

Cascade learning achieved a strong boost over the baseline with significant improvements on recall, although it does not perform as well as the fully supervised system. The features most useful for this task are POS bigrams and punctuations; syntactic category features are very useful for the supervised system, but not as useful for the cascade system.

Multiway classification Now we show how our system performs for the complete task of multiway classification of discourse relations for Chinese, recognizing each expression either as *non-discourse use* or one of the four discourse relation senses. We compare our semi-supervised multiway classification system against: *(i)* a baseline system that performs 5-way classification with the noisy labels from direct projection in the GALE data (again corresponding to the hard-label performance of the first stage classifier in our approach); *(ii)* a supervised system for 5-way classification trained on CDTB-dev (where predictions are obtained in 10-fold cross-validation fashion).

Table 5 records the accuracies for the connective expression and the five features performed best for this task. The top features for multiway relation classification, in addition to connectives, are part-of-speech bigrams, punctuations, and syntactic categories.

Notably, without any annotated data on the Chinese side, the projected semi-supervised system outperforms the 5-way supervised system for all but one of the features, and is significantly better when the top features are combined (70.2% vs. 63.55%). This finding justifies the idea and feasibility of using parallel corpora for discourse relation classification.

Binary classification Finally, we present results and the most informative features for binary classification of each relation sense individually. The semi-supervised projection system is compared against a fully supervised binary classification system over 10-fold CDTB-dev, with accuracies and F scores

	Projection		Supervised		Feature set
	A	F	A	F	
COMPARISON	94.49	59.70	96.33	57.14	Connective, categories, size of ancestor, # characters, POS bigram
CONTINGENCY	92.65	41.94	96.33	70.97	Connective, production pairs
EXPANSION	85.10	69.20	87.96	77.20	Connective, categories, production pairs, sequence pairs, POS bigram
TEMPORAL	88.37	48.65	94.08	60.47	Connective, categories, production pairs, sequence pairs

Table 6: Accuracy and F measure for binary classification for each relation, including features that significantly improves performance beyond the identity of the connective itself.

shown in Table 6. The feature sets included are the ones that significantly improve the F measure of a relation compared to that when using the connective expressions alone.

For accuracies, the semi-supervised system is only slightly (1.8-3.7%) below that of the supervised system for three of the four relations. On the other hand, F measures of the semi-supervised system are not as good as the supervised system except for the COMPARISON relation. The feature categories indicate that for Chinese discourse connectives, different feature sets are appropriate for different relations.

6 Conclusion

We investigated the tasks of discourse analysis and recognition without manual annotation. Instead, we used parallel corpora to project automatic annotations available on one side (English) to the other (Chinese). First, we conducted a corpus study which demonstrates the differences in information packaging and discourse organization between English and Chinese. We highlighted the existence of long sentences in Chinese that correspond to multiple sentences in English, mismatches between discourse expressions that are implicit vs. explicit in the two languages, and differences in the ambiguity of discourse connectives. Second, we presented a semi-supervised system that learns to predict discourse relations from the noisy annotations derived from parallel corpora. On the multiway discourse relation classification task, our system outperforms a fully supervised system trained using clean gold-standard annotation in the targeted language.

References

Amal Al-Saif and Katja Markert. 2010. The Leeds Arabic Discourse Treebank: Annotating discourse connectives for Arabic. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

David Allbritton and Johanna Moore. 1999. Discourse cues in narrative text: Using production to predict comprehension. In *AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 600–609.

Barbara Di Eugenio, Johanna D Moore, and Massimo Paolucci. 1997. Learning features that predict cue usage. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 80–87.

Quang Xuan Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 294–303.

Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 213–220.

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

Kuzman Ganchev, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. In *Proceedings of the Joint Conference of the Annual Meeting of the ACL and the International Joint Conference on Natural Language Processing of the AFNLP (ACL)*, pages 369–377.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11(3):311–325, September.

Meixun Jin, Mi-Young Kim, Dongil Kim, and Jong-Hyeok Lee. 2004. Segmentation of Chinese long sentences using commas. In *Proceedings of the SIGHAN Workshop on Chinese Language Processing*, pages 1–8.

Richard Johansson and Pierre Nugues. 2006. A FrameNet-based semantic role labeler for Swedish. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 436–443.

Wee Sun Lee and Bing Liu. 2003. Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 3, pages 448–455.

Roger Levy and Christopher D. Manning. 2003. Is it harder to parse Chinese, or the Chinese Treebank? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 439–446.

Xuansong Li, Niyu Ge, Stephen Grimes, Stephanie Strassel, and Kazuaki Maeda. 2010. Enriching word alignment with linguistic tags. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the Penn Discourse Treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 343–351.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 997–1006.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, 20:151–184, 4.

Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. 2003. Building text classifiers using positive and unlabeled examples. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 179–186.

William C. Mann and Sandra Thompson. 1988. Rhetorical Structure Theory: toward a Functional Theory of Text Organization. *Text*, 8(3):243–281.

Daniel Marcu. 1997. The rhetorical parsing of natural language texts. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 96–103. Association for Computational Linguistics.

Thomas Meyer and Andrei Popescu-Belis. 2012. Using sense-labeled discourse connectives for statistical machine translation. In *Proceedings of the Joint Workshop on Exploiting Synergies between Information Retrieval and Machine Translation and Hybrid Approaches to Machine Translation (ESIRMT-HyTra)*, pages 129–138.

Thomas Meyer, Andrei Popescu-Belis, Sandrine Zufferey, and Bruno Cartoni. 2011. Multilingual annotation and disambiguation of discourse connectives for machine translation. In *Proceedings of the Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 194–203.

Eleni Miltsakaki, Livio Robaldo, Alan Lee, and Aravind Joshi. 2008. Sense annotation in the Penn Discourse Treebank. In *Proceedings of the International Conference on Computational Linguistics and Intelligent Text Processing (CICLing)*, pages 275–286.

Umangi Oza, Rashmi Prasad, Sudheer Kolachina, Dipti Misra Sharma, and Aravind Joshi. 2009. The Hindi Discourse Relation Bank. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 158–161. Association for Computational Linguistics.

Sebastian Pado and Mirella Lapata. 2005. Cross-linguistic projection of role-semantic information. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP)*, pages 859–866.

Joonsuk Park and Claire Cardie. 2012. Improving implicit discourse relation recognition through feature set optimization. In *Proceedings of the Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 108–112.

Emily Pitler and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 186–195.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP Conference: Short Papers*, pages 13–16.

Emily Pitler, Mridhula Raghupathy, Hena Mehta, Ani Nenkova, Alan Lee, and Aravind Joshi. 2008. Easily identifiable discourse relations. In *Proceedings of the Conference on Computational Linguistics (COLING): Posters*, page 87–90.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. In *Proceedings of the Joint Conference of the Annual Meeting of the ACL and the International Joint Conference on Natural Language Processing of the AFNLP (ACL)*, pages 683–691.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2010. Realization of discourse relations by other means: Alternative lexicalizations. In *Proceedings of the Conference on Computational Linguistics (COLING): Posters*, pages 1023–1031.

Frank Schilder. 2002. Robust discourse parsing via discourse markers, topicality and position. *Natural Language Engineering*, 8(3):235–255.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the International Conference on Machine Learning (ICML)*, page 104.

Lonneke van der Plas, Paola Merlo, and James Henderson. 2011. Scaling up automatic cross-lingual semantic role annotation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 299–304.

Ben Wellner and James Pustejovsky. 2007. Automatically identifying the arguments of discourse connectives. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 92–101.

Nianwen Xue and Yaqin Yang. 2011. Chinese sentence segmentation as comma classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT): Short Papers*, pages 631–635.

Naiwen Xue, Fei Xia, Fu-dong Chiou, and Marta Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238, June.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the First International Conference on Human Language Technology Research (HLT)*, pages 1–8.

Deniz Zeyrek and Bonnie Webber. 2008. A discourse resource for Turkish: Annotating discourse connectives in the METU corpus. In *Proceedings of the 6th Workshop on Asian Language Resources*, pages 65–72.

Yuping Zhou and Nianwen Xue. 2012. PDTB-style discourse annotation of Chinese text. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 69–77.

Enforcing Topic Diversity in a Document Recommender for Conversations

Maryam Habibi
Idiap Research Institute and EPFL
Rue Marconi 19, CP 592
1920 Martigny, Switzerland
maryam.habibi@idiap.ch

Andrei Popescu-Belis
Idiap Research Institute
Rue Marconi 19, CP 592
1920 Martigny, Switzerland
andrei.popescu-belis@idiap.ch

Abstract

This paper addresses the problem of building concise, diverse and relevant lists of documents, which can be recommended to the participants of a conversation to fulfill their information needs without distracting them. These lists are retrieved periodically by submitting multiple implicit queries derived from the pronounced words. Each query is related to one of the topics identified in the conversation fragment preceding the recommendation, and is submitted to a search engine over the English Wikipedia. We propose in this paper an algorithm for diverse merging of these lists, using a submodular reward function that rewards the topical similarity of documents to the conversation words as well as their diversity. We evaluate the proposed method through crowdsourcing. The results show the superiority of the diverse merging technique over several others which not enforce the diversity of topics.

1 Introduction

We present a diverse retrieval technique for ranking documents that are spontaneously retrieved and recommended to people during a conversation. These documents represent potentially useful information for the conversation participants. The information needs of the participants are represented by implicit queries which are built in the background based on their current speech, specifically from keywords obtained from the conversation transcripts. Since people usually mention several topics even during a short conversation span, such keyword sets are made of content words related to different topics. When juxtaposed in an implicit query, these topics may have noisy effects on the retrieval results (Bhogal et al., 2007; Carpineto and Romano, 2012).

The purpose of this paper is to present a method for merging lists of documents retrieved through multiple implicit queries prepared for short conversations spans. Several topically-separated queries are constructed from keywords, and generate several lists of documents. The goal of the method proposed here is to generate a unique and concise list of documents that can be recommended in real time to the conversation participants. The list should cover the maximum number of implicit queries and therefore topics. To merge the lists of documents according to these criteria, we use inspiration from extractive text summarization (Lin and Bilmes, 2011; Li et al., 2012) and from our own previous work on diverse keyword extraction (Habibi and Popescu-Belis, 2013). The method proposed here rewards at the same time topic similarity – to select the most relevant documents to the conversation fragment – and topic diversity – to cover the maximum number of implicit queries and therefore topics in a concise and relevant list of recommendations, if more than one topic is discussed in the conversation fragment.

Several studies have been previously carried out on merging lists of results in information retrieval. Despite the superficial similarity, the problem here is in fact different from distributed information retrieval, where several lists of results from *different* search engines for the *same* query must be merged. Moreover, many studies addressed the topic diversification approach for re-ranking the retrieved results of a single query. However, these approaches are not directly applicable to multiple queries.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 588–599, Dublin, Ireland, August 23-29 2014.

The paper is organized as follows. In Section 2 we review existing techniques for merging and re-ranking lists of search results which are applicable here. We then explain the general framework of our document recommender system in Section 3. In Section 4 we describe the proposed algorithm for diverse merging of lists of recommendations. Section 5 presents the data, the parameters setting, and evaluation tasks for comparing document lists. In Section 6 we first demonstrate empirically the benefits, for just-in-time document recommendation, of separating users' information needs into multiple topically-separated queries rather than using a unique query. Then, we compare the proposed diverse merging technique with several alternative ones, showing that it outperforms them according to human judgments of relevance, and also exemplify the results on one conversation fragment given in the Appendix A.

2 Related Work

Just-in-time document retrieval systems have been designed to recommend to their users documents which are potentially relevant to their activities, e.g. individual users authoring documents or browsing various repositories, or small groups holding business or private meetings (Hart and Graham, 1997; Rhodes and Maes, 2000; Popescu-Belis et al., 2008). When using a document recommender system, people are generally unwilling to examine a large number of recommended documents, mainly because this would distract them from their main activity. Several solutions to this problem have been proposed.

For instance, the Watson document recommender system (Budzik and Hammond, 2000), designed for reading or writing activities, clustered the document results and selected from each cluster the best representative to generate a list of recommendations. Clustering results is not suitable for our application where the mixture of topics in a single query will degrade the document results aimed to be clustered (Bhogal et al., 2007; Carpineto and Romano, 2012), and consequently may have a damaging effect on the clusters' representatives. The second part of the method, which selected the best representative of the clusters in the final document list can be helpful; however, its effectiveness relies on having clusters with the same level of importance (Wu and McClean, 2007).

Many studies in information retrieval addressed the problem of diverse ranking, which can be stated as a tradeoff between finding relevant versus diverse information (Robertson, 1997). The existing diverse ranking proposals differ in their diversifying policies and definitions, which can be categorized into implicit methods (Carbonell and Goldstein, 1998; Zhai et al., 2003; Radlinski and Dumais, 2006; Wang and Zhu, 2009) or explicit ones (Agrawal et al., 2009; Carterette and Chandar, 2009; Santos et al., 2010; Vargas et al., 2012). The implicit approaches assume that similar documents will cover similar aspects of a query, and have to be demoted in the ranking to promote relative novelty and reduce overall redundancy. In one of the earliest approaches, Carbonell and Goldstein (1998) introduced Maximal Marginal Relevance (MMR) to re-rank documents based on a tradeoff between the relevance of document results and relative novelty as a measure of diversity. MMR was also used by Radlinski and Dumais (2006) to re-rank results from a query set which is generated for a user query and represents a variety of potential user intents.

Instead of implicitly accounting for the aspects covered by each document, another option is to explicitly model these aspects within the diversification approach. Agrawal et al. (2009) introduced a submodular objective function to minimize the probability of average user dissatisfaction by assuming a taxonomy of information and modeling user query aspects at the topical level of this taxonomy. Alternatively, Santos et al. (2010) proposed another submodular objective function to maximize coverage and minimize redundancy with respect to query aspects modeled in a keyword-based representation form instead of a predefined taxonomy.

In our case, the recommender system for conversational environments requires diversity in the results of multiple topically-separated queries, rather than of a single ambiguous query. Therefore, a new approach will be proposed, and will be compared in particular to a version of the explicit diversification approach (Santos et al., 2010) adapted to our problem.

3 Framework of our Document Recommender System

We have designed the Automatic Content Linking Device (ACLD), a speech-based just-in-time document recommender system for business meetings (Popescu-Belis et al., 2008; Popescu-Belis et al., 2011).

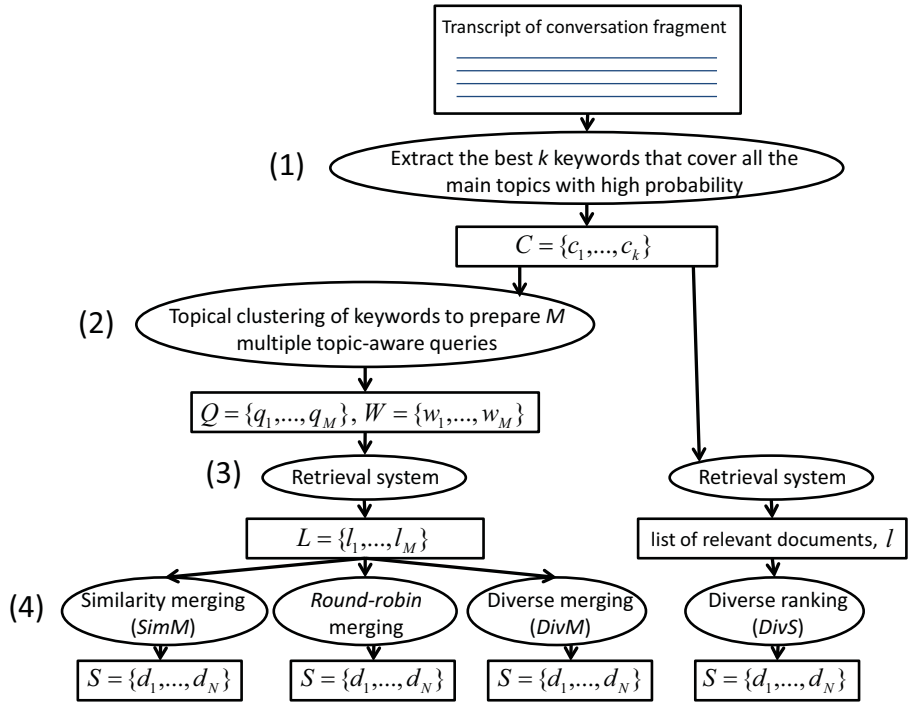


Figure 1: The four stages of our document recommendation approach (shown vertically: 1–4) and the four options considered in this paper (bottom line: *SimM*, *Round-robin*, *DivM*, and *DivS*).

The ACLD monitors the ongoing conversation, and formulates queries based on the words detected by a real-time automatic speech recognition (ASR) system (Garner et al., 2009). The queries are fired periodically to retrieve documents which are then recommended to users by displaying their titles along with relevant excerpts. As these queries are built and triggered in the background, they are referred to as 'implicit queries', as opposed to 'explicit' ones that could be formulated by users. Just-in-time document recommendation in the ACLD system proceeds according to the steps shown in Figure 1, which displays at step 4 the various options for merging lists of results that are the focus of this paper.

Prior to the first processing step outlined in Figure 1, the ACLD must decide when to make a recommendation, and what portion of the conversation prior to that moment should be used. This question is beyond the scope of this paper, and remains to be fully investigated, using verbal and non-verbal criteria. Here, for the reasons explained in Section 5.2, the ACLD recommends documents every two minutes, segmenting the conversation at the end of the nearest utterance and using the entire conversation fragment since the previous recommendation. Although in practice the results of the current recommendation process are merged with the previous ones (using a weighted mechanism that embodies the idea of "persistence" of documents over time), in this paper we will consider the recommendation for each fragment independently of the previous one.

The recommendation process represented in Figure 1 starts by extracting a set of keywords, C, from the words recognized by the ASR system from the users' conversations. The keywords are extracted using the diverse keyword extraction technique that we proposed (Habibi and Popescu-Belis, 2013), which maximizes the coverage of the topics of a text by the extracted keyword set, as we also target in this paper. Then, implicit queries which express the users' information needs are formulated using the keyword set, following two alternative approaches depicted in step 2 of Figure 1. In a baseline model (right side of the figure), a single query is built for the conversation fragment using the entire keyword list as an implicit query. In the approach we are advocating, multiple topically-separated queries are produced for the conversation fragment (step 2, left side of the figure). This is described in a separate document (Habibi and Popescu-Belis, submitted), but can be outlined as follows. The implicit queries are obtained by clustering the above-mentioned keyword set into several topically-separated subsets, each one corresponding to an abstract topic obtained using topic modeling techniques (similarly to the model

presented in Subsection 4.1). Each subset is an implicit query, and is weighted based on the importance of the topic to which it is associated.

In step 3, we separately submit each implicit query to the Apache Lucene search engine over the English Wikipedia and obtain several lists of relevant articles. Finally, we merge and re-rank these lists before recommendation (step 4). One baseline alternative is the explicit diverse ranking technique proposed by Santos et al. (2010) for diversifying the primary search results retrieved for a single query, shown on the right side of the figure. To compare the methods, we adapted this latter method to make it applicable to our system when a single implicit query is built for a conversation fragment, by defining query aspects using the abstract topics employed for query and document representation. The method is noted *DivS* as it *diversifies* documents from a *single* list.

Our proposal lies at step 4. As represented on the left side of Figure 1, in our system, we merge the lists of documents retrieved for multiple implicit queries. We thus propose a new method noted *DivM* and we compare it with two other merging techniques. The first one, noted *SimM*, ignores the diversity of topics in the list of results and ranks documents only by considering their topic similarity to the conversation fragment. The second one is the merging technique used by the above-mentioned Watson system (Budzik and Hammond, 2000), which uses Round robin merging, hence it is noted *Round-robin*. In contrast, our proposed method, *DivM*, is a *diverse merging* technique which we now proceed to define formally.

4 Diverse Merging of the Results of Multiple Queries

The diverse merging of retrieved document lists is the process of creating a short, diverse and relevant list of recommended documents which covers the maximum number of topics of each conversation fragment. The merging algorithm rewards diversity by decreasing the gain of selecting documents from a list as the number of its previously selected documents increases. The method proceeds in two steps. First, we represent queries and the corresponding list of candidate documents from the Apache Lucene search engine using topic modeling techniques, and then we rank documents by using topical similarity and rewarding the coverage of different lists.

4.1 Document and Query Representation

A topic model represents the abstract topics which occur in a collection of documents – here, preferably, a collection that is representative of the domain of the conversations. Once trained, topic models such as Probabilistic Latent Semantic Analysis (PLSA) or Latent Dirichlet Allocation (LDA) can be used to determine the distribution of abstract topics in each set of words composing either a conversation fragment, or a query, or a document. LDA implemented in the Mallet toolkit (McCallum, 2002) is used here to train topic models because it does not suffer from the over-fitting of PLSA (Blei et al., 2003).

We first learn a probability model for observing a word v in a document d through the set of abstract topics $T = \{t_1, ..., t_z, ..., t_Z\}$, where Z is the number of topics, using the Mallet toolkit:

$$p(v|d) = \sum_{z=1}^{Z} p(v|t_z) \cdot p(t_z|d) \quad (1)$$

The topic-word distribution $p(v|t_z)$ and the document-topic distribution $p(t_z|d)$, which are obtained using topic modeling, respectively show the contribution of the word v in the construction of the topic t_z, and the distribution of topic t_z in the document d with respect to the other topics.

We represent each new text or fragment A (e.g. from a conversation or document) by a set of probability distributions over all abstract topics T noted as $P(A) = \{p(t_1|A), ..., p(t_z|A), ..., p(t_Z|A)\}$ where $p(t_z|A)$ is inferred using the Gibbs sampling implemented by the Mallet toolkit given the topic models previously learned. We associate to each new document d_i and query q_j a set of topic probabilities according to the above definition noted respectively as $P(d_i) = \{p(t_1|d_i), ..., p(t_z|d_i), ..., p(t_Z|d_i)\}$ and $P(q_j) = \{p(t_1|q_j), ..., p(t_z|q_j), ..., p(t_Z|q_j)\}$.

4.2 Diverse Merging Problem

As stated above, our goal is to recommend a short ranked list of documents answering the users' information needs hypothesized in a conversation fragment, which are modeled by multiple topic-aware implicit

queries as described in Section 3. We build the final list of recommended documents by merging the document lists, one from each implicit query, with the objective of the maximum coverage of the topics of the conversation fragment. Since each document list contains documents found by a search engine given an implicit query, which was prepared for one of the main topics of the conversation fragment, we merge the lists by selecting documents from the maximum number of lists in addition to maximizing their topical similarity to the conversation fragment.

The problem of diverse merging of lists thus amounts to finding a ranked subset of documents $S \subset \cup_{i=1}^{M} l_i$, which are the most representative of all the result lists l_i, and potentially the most informative with respect to the conversation fragment and the information needs that are implicitly stated. This problem is an instance of the maximum coverage problem, which is known to be NP-hard. Our formulation and solution proceed as follows.

Let us consider a set of implicit queries $Q = \{q_1, ..., q_M\}$, and the corresponding set of document lists $L = \{l_1, ..., l_M\}$ resulting from each query. M is the number of implicit queries of the fragment, and each l_i is a list of documents $\{d_1, ..., d_{N_i}\}$ which are retrieved for query q_i. We define the weight w_i of each query q_i as the importance within the conversation fragment of the topics represented in the query q_i, and compute it as the topical similarity of q_i to the fragment, as shown in Equation 2. In this equation, q is the query made from the whole keyword set, which we call a *collective query*, and includes keywords for all the main topics of the conversation fragment in one query. In turn, we associate to q a set of probabilities over abstract topics, $P(q) = \{p(t_1|q), ..., p(t_Z|q)\}$, similar to the representation of implicit queries explained in Subsection 4.1.

$$w_i = \sum_{z=1}^{Z} p(t_z|q_i) \cdot p(t_z|q) \tag{2}$$

4.3 Defining a Diverse Reward Function

Although the maximum coverage problem is NP-hard, it has been shown that a greedy algorithm can find an approximate solution guaranteed to be within a factor of $(1 - 1/e) \simeq 0.63$ of the optimal one if the coverage function is submodular and monotone non-decreasing[1] (Nemhauser et al., 1978). Several monotone submodular functions have been proposed in various domains for a similar underlying problem, such as explicit diverse re-ranking of retrieval results (Agrawal et al., 2009; Santos et al., 2010; Vargas et al., 2012), extractive summarization of a text (Lin and Bilmes, 2011; Li et al., 2012), or our own model of diverse keyword extraction from a text (Habibi and Popescu-Belis, 2013).

We define a monotone submodular function for diverse merging of document lists inspired by the latter two applications, who proposed a power function with a scaling exponent between 0 and 1 for diverse selection of sentences (or keywords) covering the maximum number of topics of a given document with a fixed number of items. To adapt these techniques to the problem of diverse merging, from the perspective of capturing users' information needs in the set of recommended documents, we define here a reward function enforcing the diverse merging of the lists of document results.

We first estimate the topical similarity of the document subset $S_i = S \cap l_i$ to the collective query q (see Subsection 4.2) as r_{S_i}:

$$r_{S_i} = \sum_{d \in S_i} \sum_{z=1}^{Z} p(t_z|d) \cdot p(t_z|q) \tag{3}$$

We then propose the following reward function f for each S_i containing relevant documents selected from l_i (results of implicit query q_i), where w_i is the topical similarity of q_i to the conversation fragment (see Equation 2), and λ is an exponent parameter between 0 and 1. This reward function is submodular because it has the diminishing returns property when r_{S_i} increases.

$$f : r_{S_i} \rightarrow w_i \cdot r_{S_i}^{\lambda} \tag{4}$$

[1] A function F is *submodular* if $\forall A \subseteq B \subseteq T \setminus t$, $F(A + t) - F(A) \geq F(B + t) - F(B)$ (diminishing returns) and is *monotone non-decreasing* if $\forall A \subseteq B$, $F(A) \leq F(B)$.

The set S is ultimately ranked by maximizing the cumulative reward function $R(S)$ over all the lists, written as follows:

$$R(S) = \sum_{i=1}^{M} w_i \cdot r_{S_i}^{\lambda} \tag{5}$$

The probability of selecting documents from the list of results for q_i thus depends on w_i, the topical similarity of the query to the conversation fragment. This is in contrast to choosing the best representative document from the list of documents relevant to each query, like in the Watson system, which does not select more documents for queries with higher weight before considering lower weight ones. Our model rewards diversity to increase the chance of choosing documents from all the lists of results retrieved for implicit queries.

4.4 Finding the Optimal Document List

Since $R(S)$ is a monotone submodular function, we propose a greedy algorithm (Alg. 1) to maximize $R(S)$. If $\lambda = 1$, the reward function ignores the diversity constraint, because it does not penalize multiple selections from the same list l_i and ranks documents only depending on their similarity to the collective query and on the weights of implicit queries. However, when $0<\lambda<1$, as soon as a document is selected from the list of results of an implicit query, other documents from the same list start having diminishing returns as competitors for selection. Decreasing the value of λ increases the impact of the diversity constraint on ranking documents, which augments the chance of recommending documents from other document lists.

Input : query set Q of size M with probabilities, set of weights W, set of lists of document results L with probabilities, number of recommended documents k
Output: set of recommended documents S
$S \leftarrow \emptyset$;
for $i = 1$ ***to*** M ***step*** 1 **do**
  $S_i \leftarrow \emptyset$;
end
while $|S| \leq k$ **do**
  $S \leftarrow S \cup argmax_{d \in ((\cup_{i=1}^{M} l_i) \setminus S)}(g(d))$ where $g(d) = \sum_{i=1}^{M} w_i \cdot [r_{\{d\} \cap l_i} + r_{S_i}]^{\lambda}$;
  for $i = 1$ ***to*** M ***step*** 1 **do**
    $S_i = l_i \cap S$;
  end
end
return S;

Algorithm 1: Diverse merging of document results for recommendation.

5 Data, Settings and Evaluation Method

The experiments were performed on conversational data from the ELEA Corpus (Emergent LEader Analysis, Sanchez-Cortes et al. (2012)). Implicit queries were formulated as presented above in Figure 1 using keywords extracted from each conversation fragment, defined as below (Subsection 5.1). Each subset of keywords obtained by topical clustering of the keyword set resulted in an implicit query. The lists of document results for each implicit query were obtained by submitting the query to the Apache Lucene search engine[2] over the English Wikipedia[3]. These initial lists of results were ultimately merged into final recommendation lists of documents using the four alternative methods from Figure 1, including the one we proposed. This section presents the data, system parameters, and evaluation methods used in our experiments.

[2]Available from `http://lucene.apache.org`.
[3]A local copy was downloaded from `http://dumps.wikimedia.org`.

5.1 Conversational Corpus

The ELEA Corpus comprises nearly ten hours of recorded meetings in English and French. Each meeting consists in a role play game in which participants play survivors of an airplane crash in a mountainous region. They must rank a list of 12 items with respect to their utility for surviving until they are rescued. We used from the ELEA corpus four English conversations of around fifteen minutes each, which have been manually transcribed and segmented at the speaker turn level.

One of the most important issues for a just-in-time document recommender system is to determine the appropriate timing of the recommendations, and the size of the context to use for computing them. Here, awaiting future investigations[4], we decided to make recommendations approximately every two minutes, at the end of an ongoing speaker turn, and consider as input the words uttered since the previous recommendation. A segment size of two minutes enables us to collect an appropriate number of words (neither too small nor too large) in order to extract keywords, model the topics, and formulate implicit queries. Based on our experience with the ACLD, it also corresponds to an acceptable frequency for receiving suggestions.

Therefore, our test data comprises 26 two-minute segments, each of them ending at a speaker change. On average, segments contain 278 words (including stop words). Once topic modeling is applied, the average number of topics per fragment is 5, with an observed minimum of 3 and a maximum of 9.

5.2 Parameter Settings for Experimentation

As document search is performed over the English Wikipedia, we trained our topic models on this corpus as well. We used only a subset of it for tractability reasons, i.e. about 125,000 articles as in other studies (Hoffman et al., 2010). The subset is randomly selected from the entire English Wikipedia. As in previous studies, we fixed the number of topics at 100 (Boyd-Graber et al., 2009; Hoffman et al., 2010).

The exponent of the submodular function was set to $\lambda = 0.75$, as in our diverse keyword extraction study (Habibi and Popescu-Belis, 2013). This was found to be the best value for diverse merging of lists of results, as it leads to a reasonable balance between relevance and diversity in the aggregated list of documents. Of course, if sufficient training data were available, this could be used to optimize λ.

The number of recommended documents was fixed at five in our experiments. This value was selected again based on user preferences observed with the ACLD. Moreover, this is also the value of the average number of topics in a conversation fragment, which allows the system to cover on average one result per topic. Experiments with other values were not carried out due to the cost of evaluation.

5.3 Evaluation Protocol and Metrics

We designed a task that measures the relevance of recommended document lists for each of the test conversation fragment. Based on validation experiments in our previous work (Habibi and Popescu-Belis, 2012), the task requires subjects to compare two lists obtained by two different methods. Using a web browser, the subjects had to read the conversation transcript, answer several control questions about its content, and then decide which of the two lists provides more relevant documents, with the following options: the first list is better than the second one; the second is better than the first; both are equally relevant; or both are equally irrelevant. The position of each system (first or second) was randomized across the tasks.

The 26 comparison tasks (one for each ELEA fragment) were crowdsourced via Amazon's Mechanical Turk as "human intelligence tasks" (HITs). For each HIT we recruited ten workers, only accepting those with greater than 95% approval rate and more than 1000 previously approved HITs (qualification control). We only kept answers from the workers who answered correctly our control questions about each HIT. Each worker could answer the entire set of 26 HITs, or part of it. We observed that the average time spent per HIT was around 90 seconds.

[4]For instance, they could combine an analysis of non-verbal information to detect "interruptibility" and of verbal information to detect topic changes and perform online segmentation (Mohri et al., 2010). Topic changes, however, are not appropriate moments to make recommendations because it would be useless to recommend documents about a topic that the users no longer discuss (Jones and Brown, 2004).

To consolidate the comparative judgments over a large number of subjects and conversation fragments, and compute an aggregated score, we applied a qualification control factor to the human judgments (to reduce the effect of judgments which disagree with the majority vote) and another one to the HITs (to reduce the impact of undecided HITs on the global scores). This was done by using the PCC-H metric, defined and validated in our previous work (Habibi and Popescu-Belis, 2012), which provides two scores, one for each document list, summing up to 100%; a higher value indicates a better list. In addition to PCC-H, we also provide below (Table 1) the raw preference scores for each comparison, i.e. the number of times a system was preferred over another one, although PCC-H was shown to be a more reliable indicator of quality.

6 Experimental Results

We merged and re-ranked the document lists intended to be recommended during a conversation by the four methods presented above in Section 3 and Figure 1. Three methods merge lists of results from topically-separated queries: *SimM* only considers their similarity with the fragment; *Round-robin* picks the best document in each list; and our proposal, *DivM*, considers the diversity and importance of topics. A fourth method, *DivS*, uses one query made of all keywords extracted from the conversation fragment, and ranks the documents using the diverse re-ranking technique proposed by Santos et al. (2010).

Binary comparisons were performed between pairs of techniques, using crowdsourcing over 26 conversation fragments of the ELEA Corpus, and aiming to minimize the number of binary comparisons while still ordering completely the methods according to their perceived quality.

6.1 Diverse Re-ranking vs. Similarity Merging

We first performed a comparison between the top five documents generated by two recommendation strategies, *DivS* and *SimM*, over 26 conversation fragments of the ELEA Corpus. The consolidated relevance score (PCC-H) is 75% for *SimM* vs. 25% for *DivS*, as shown in Table 1. These scores indicate the superiority of *SimM* over *DivS*. In other words, separating the mixture of topics of a fragment into multiple topically-separated queries mitigates the negative effect of the mixture of topics on the suggestions.

6.2 Comparison across Merging Techniques

Binary comparisons were then performed between pairs of merging techniques (*SimM*, *Round-robin*, and *DivM*), using the same experimental settings. The PCC-H scores are 62% for *DivM* vs. 38% for *Round-robin*, 59% for *DivM* vs. 41% for *SimM*, and 56% for *Round-robin* vs. 44% for *SimM*, as shown in Table 1. The scores show that the diverse merging of lists of documents improves recommendations, and indicate the following high to low ranking: *DivM* $>$ *Round-robin* $>$ *SimM*.

SimM ranks lowest in this ordering, likely because of the ignorance of diversity in the list of results. *Round-robin* is second, likely because it disregards the major differences of importance among implicit queries in a conversation fragment. The results of the comparisons confirm that the *DivM* technique, which merges lists of documents by considering the diversity of topics in the list of recommendations, in proportion to their importance in the conversation, is the most satisfying to the majority of human subjects.

6.3 Impact of the Topical Diversity of Fragments

To further examine the benefits of our method, we studied its sensitivity to the number of topics in the conversation fragments. For this purpose, we divided the set of test fragments into two subsets. The first one (noted 'A' in Table 1) gathers the fragments for which fewer than or exactly five main topics (and therefore implicit queries) have been computed. The other fragments, with more than five main topics, form the second subset (noted 'B'). The value of five corresponds to the average number of main topics per fragment as well as to the number of recommended documents in our experiments.

As shown in Table 1, although there is an improvement in the comparison scores of *DivS* over *SimM* when the number of conveyed topics in the fragments is higher than the number of recommended documents (subset B), the comparison scores indicate the superiority of *SimM* over *DivS* in both cases, and

Compared methods (m_1 vs. m_2)	PCC-H relevance score (%)						Raw preferences (%)	
	A		B		A ∪ B		A ∪ B	
	m_1	m_2	m_1	m_2	m_1	m_2	m_1	m_2
SimM* vs. *DivS	**80**	20	**70**	30	**75**	25	**70**	30
Round-robin* vs. *SimM	33	**67**	**68**	32	**56**	44	**52**	48
DivM* vs. *Round-robin	**64**	36	**60**	40	**62**	38	**58**	42
DivM* vs. *SimM	**54**	46	**60**	40	**59**	41	**58**	42

Table 1: Comparative scores of the recommended document lists from four methods: *DivS*, *SimM*, *Round-robin*, and *DivM*, evaluated by human judges over the ELEA Corpus. Subset A gathers fragments with fewer than or exactly five topics, while subset B gathers all the other fragments. The results imply the following ranking: *DivM* > *Round-robin* > *SimM* > *DivS*.

confirm the benefit of the diverse merging techniques. When comparing *Round-robin* versus *SimM*, the scores show the superiority of the former method when the number of conveyed topics in fragments is higher than the number of recommended documents, because it provides a diverse lists of documents in which documents relevant to less important topics are not displayed. However, when the number of topics is smaller than the number of recommendations, *SimM* provides better results. The reason of the decrease in the scores of *Round-robin* is likely the ignorance of the actual importance of the main topics when ranking documents. Overall, as shown in Table 1, regardless of the number of topics conveyed in the fragments, *DivM* always outperforms *Round-robin* and *SimM*.

6.4 Example of Document Results

To illustrate how *DivM* surpasses the other techniques, we consider an example from one of the conversation fragments of the ELEA Corpus. The manual transcript of this conversation fragment is given in the Appendix A. As described in Section 5, the conversation participants had to select a list of 12 items vital to survive in winter while waiting to be rescued. The keywords extracted from the manual transcript of this fragment by our method (Habibi and Popescu-Belis, 2013) are: *fire, lighter, cloth, shoe, cold, die, igloo, walking*. As our keyword extraction method was shown to be robust to ASR noise, we only use here the reference transcripts (Habibi and Popescu-Belis, submitted).

We display the topically-aware implicit queries prepared by our method from this keyword list along with their weights in Table 2. Then, in Table 3 we show the retrieval results (five highest-ranked Wikipedia pages) obtained by the four methods using the reference transcript of this fragment.

As shown in Table 2, each implicit query corresponds to one of the main topics of the fragment with a specific weight. In this example, the main topics spoken in the fragment are about making an igloo, lightening a fire, having warm clothes, and suitable shoes for walking.

As shown in Table 3, *DivS* provides two irrelevant documents likely because the single (collective) query does not separate the mixture of topics in the conversation fragment, and leads to some poor results (Wikipedia pages) such as "Cold Fire (Koontz novel)". *SimM* slightly improves the results by separating the discussed topics of the conversation fragment into multiple queries. However, it does not cover all the

Implicit queries	Weights
q_1 = {fire, cold, igloo, lighter}	w_1 = 0.110
q_2 = {shoe, lighter, walking}	w_2 = 0.097
q_3 = {cloth}	w_3 = 0.058
q_4 = {die}	w_4 = 0.040
q_5 = {igloo}	w_5 = 0.026

Table 2: Example of implicit queries built from the keyword list extracted from a sample fragment of the ELEA Corpus. Each query covers one of the main topics of the fragment and has a different weight.

DivS	***SimM***	***Round-robin***	***DivM***
Flint spark lighter	Igloo	Igloo	Igloo
Extended Cold Weather Clothing System	Flint spark lighter	Shoe	Shoe
Cold Fire (Koontz novel)	Lighter	Jersey (clothing)	Flint spark lighter
Igloo	Lighter (barge)	Die Hard	Jersey (clothing)
Walking	Worcester Cold Storage Warehouse fire	Flint spark lighter	Lighter

Table 3: Example of retrieved Wikipedia pages from the four different methods tested in this paper. Results of diverse merging (*DivM*) appear to cover more topics relevant to the conversation fragment than other methods. The average ranking (*DivM* > *Round-robin* > *SimM* > *DivS*) is also observed in this example.

topics mentioned in the fragment due to mostly focusing on the single topic represented by q_1. *Round-robin* further enhances the results by adding diversity, but as it gives the same level of importance to all topics, it provides a poor result like "Die Hard" from a topic of the conversation fragment with a small weight. The results of *DivM* appear to be the most useful ones, as they include other articles relevant to q_1, q_2, and q_3 before showing results relevant to the low weight queries q_4 and q_5. Therefore, in this example, *DivM* provides better ranking of documents by covering the largest number of main topics mentioned in the fragment.

7 Conclusion

We proposed a diverse merging technique for combining lists of documents from multiple topically-separated implicit queries, prepared using keyword lists obtained from the transcripts of conversation fragments. Our *diverse merging* method *DivM* provides a short, diverse, and relevant list of recommendations, which avoids distracting participants that would consider it during the conversation. We also compared *DivM* to existing merging techniques, in terms of comprehensiveness and relevance of the final recommended list of documents to the conversation fragment. The human judgments collected via Amazon Mechanical Turk showed that *DivM* outperforms all other methods.

Moreover, these results emphasized the benefit of splitting the keyword set into multiple topically-separated queries: the suggested lists of documents from *DivS* (which accounts for the diversity of results by re-ranking the documents of a single list) were indeed found less relevant than those from *SimM* and the other two methods, which merged results from multiple queries.

In the future, the diverse merging method *DivM* will be integrated in the ACLD just-in-time retrieval system for conversational environments, with implicit queries that are prepared from the ASR transcript of users' conversation. User-oriented evaluation experiments will be conducted. We will also enable the system to answer explicit queries asked by users, considering contextual factors to improve the relevance of the answers, which will complement the recommendation functionality based on implicit queries.

Acknowledgments

The authors are grateful to the Swiss National Science Foundation for its support through the IM2 NCCR on Interactive Multimodal Information Management (2002-2013, see `http://www.im2.ch`), and to the Hasler Foundation for its support through the REMUS project (Re-ranking Multiple Search Results for Just-in-Time Document Recommendation, 2014). The authors also thank the anonymous reviewers for their helpful suggestions.

References

Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 5–14. ACM.

Jagdev Bhogal, Andy Macfarlane, and Peter Smith. 2007. A review of ontology based query expansion. *Information and Processing Management*, 43:866–886.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jonathan Boyd-Graber, Jordan Chang, Sean Gerrish, Chong Wang, and David Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1–9.

Jay Budzik and Kristian J. Hammond. 2000. User interactions with everyday applications as context for just-in-time information access. In *Proceedings of the 5th International Conference on Intelligent User Interfaces (IUI)*, pages 44–51. ACM.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM.

Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)*, 44(1):1–56.

Ben Carterette and Praveen Chandar. 2009. Probabilistic models of ranking novel documents for faceted topic retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1287–1296.

Philip N. Garner, John Dines, Thomas Hain, Asmaa El Hannani, Martin Karafiat, Danil Korchagin, Mike Lincoln, Vincent Wan, and Le Zhang. 2009. Real-time ASR from meetings. In *Proceedings of Interspeech 2009 (10th Annual Conference of the International Speech Communication Association)*, pages 2119–2122.

Maryam Habibi and Andrei Popescu-Belis. 2012. Using crowdsourcing to compare document recommendation strategies for conversations. In *Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE 2011)*, pages 15–20.

Maryam Habibi and Andrei Popescu-Belis. 2013. Diverse keyword extraction from conversations. In *Proceedings of the ACL 2013 (51th Annual Meeting of the Association for Computational Linguistics)*, pages 651–657.

Maryam Habibi and Andrei Popescu-Belis. submitted. Keyword extraction and clustering for document recommendation in conversations. Manuscript submitted for publication.

Peter E. Hart and Jamey Graham. 1997. Query-free information retrieval. *International Journal of Intelligent Systems Technologies and Applications*, 12(5):32–37.

Matthew D. Hoffman, David M. Blei, and Francis Bach. 2010. Online learning for Latent Dirichlet Allocation. *Proceedings of 24th Annual Conference on Neural Information Processing Systems*, 23:856–864.

Gareth J.F. Jones and Peter J. Brown. 2004. Context-aware retrieval for ubiquitous computing environments. In *Mobile and ubiquitous information access*, pages 227–243. Springer.

Jingxuan Li, Lei Li, and Tao Li. 2012. Multi-document summarization via submodularity. *Applied Intelligence*, 37(3):420–430.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the ACL 2011 (49th Annual Meeting of the Association for Computational Linguistics)*, pages 510–520.

Andrew K. McCallum. 2002. MALLET: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Mehryar Mohri, Pedro Moreno, and Eugene Weinstein. 2010. Discriminative topic segmentation of text and speech. In *International Conference on Artificial Intelligence and Statistics*, pages 533–540.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming Journal*, 14(1):265–294.

Andrei Popescu-Belis, Erik Boertjes, Jonathan Kilgour, Peter Poller, Sandro Castronovo, Theresa Wilson, Alejandro Jaimes, and Jean Carletta. 2008. The AMIDA Automatic Content Linking Device: Just-in-time document retrieval in meetings. In *Proceedings of MLMI 2008 (Machine Learning for Multimodal Interaction)*, LNCS 5237, pages 272–283.

Andrei Popescu-Belis, Majid Yazdani, Alexandre Nanchen, and Philip N. Garner. 2011. A speech-based just-in-time retrieval system using semantic search. In *Proceedings of 49th Annual Meeting of the ACL*, pages 80–85.

Filip Radlinski and Susan Dumais. 2006. Improving personalized web search using result diversification. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 691–692. ACM.

Bradley J. Rhodes and Pattie Maes. 2000. Just-in-time information retrieval agents. *IBM Systems Journal*, 39(3.4):685–704.

Stephen E. Robertson. 1997. The probability ranking principle in IR. In Karen Sparck Jones and Peter Willett, editors, *Readings in information retrieval*, pages 281–286. Morgan Kaufmann Publishers Inc.

Dairazalia Sanchez-Cortes, Oya Aran, Marianne Schmid Mast, and Daniel Gatica-Perez. 2012. A nonverbal behavior approach to identify emergent leaders in small groups. *IEEE Trans. on Multimedia*, 14(3):816–832.

Rodrygo L.T. Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th Int. Conf. on the World Wide Web*, pages 881–890. ACM.

Saúl Vargas, Pablo Castells, and David Vallet. 2012. Explicit relevance models in intent-oriented information retrieval diversification. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 75–84. ACM.

Jun Wang and Jianhan Zhu. 2009. Portfolio theory of information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 115–122. ACM.

Shengli Wu and Sally McClean. 2007. Result merging methods in distributed information retrieval with overlapping databases. *Information Retrieval*, 10(3):297–319.

Cheng Xiang Zhai, William W. Cohen, and John Lafferty. 2003. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 10–17. ACM.

Appendix A. Transcript of a Conversation Fragment from the ELEA Corpus

The following transcript of a conversation fragment (speakers noted A through C) was submitted to the document recommender system and is exemplified in Section 6.4. The corresponding implicit queries and recommendations are respectively shown in Tables 2 and 3.

```
  A: okay I start.
  B: how  how do you want to proceed?
  A: I guess -
  C: yes what is the most important?
  A: I guess fire light.
  B: fire lighter?
  A: fire, yes.  I would say if we had something we can fire with -- I guess that
the lighter is useful in getting some sparks.
  B: hopefully.
  A: so we can use either newspaper or -- something like that.
  C: but again - first it is more important to have enough err clothes.
  A: and for me, more important to know where to go.  I would say that the compass.
  C: I mean -- if you don't have enough clothes so -- at one point you can --
  B: you can die.
  C: yes you can -- you will die.  so first issue, try to keep yourself alive and
then you can --
  A: but -- but you already have some --
  B: basics.  you everything.  you have enormous which is and so is no shoes here.
  C: okay that we have shoes so -- okay.
  B: because seventy kilometers will take you how many days?  err in the snow --
what do you think?
  A: two or three.
  B: it can be two or three days?
  C: yes, but okay you cannot always have fire with you -- but you need always
have clothes with you.  I mean it is the only thing that protects you when you are
walking.
  B: oh yes.  and erm you can make an igloo during the evening.  not that cold.
only about five degrees.  so lighting a fire is not so important.
  C: I guess fire is an extra.  I mean it is important but err for me first it is
important that when you keep walking you should be protected.
```

Identifying Important Features for Graph Retrieval

Zhuo Li and **Sandra Carberry** and **Hui Fang*** and **Kathleen F. McCoy**
ivanka@udel.edu carberry@udel.edu hui@udel.edu mccoy@udel.edu
Department Computer and Information Science,
*Department of Electrical and Computer Engineering
University of Delaware

Abstract

Infographics, such as bar charts and line graphs, occur often in popular media and are a rich knowledge source that should be accessible to users. Unfortunately, information retrieval research has focused on the retrieval of text documents and images, with almost no attention specifically directed toward the retrieval of information graphics. Our work is the first to directly tackle the retrieval of infographics and to design a system that takes into account their unique characteristics. Learning-to-rank algorithms are applied on a large set of features to develop several models for infographics retrieval. Evaluation of the models shows that features pertaining to the structure and the content of graphics should be taken into account when retrieving graphics and that doing so results in a model with better performance than a baseline model that relies on matching query words with words in the graphic.

1 Introduction

Infographics are non-pictorial graphics such as bar charts and line graphs. When such graphics appear in popular media, they generally have a high-level message that they are intended to convey. For example, the graphic in Figure 1 ostensibly conveys the message that Toyota has the highest profit among the automobile companies listed. Thus infographics are a form of language since, according to Clark (Clark and Curran, 2007), language is any deliberate signal that is intended to convey a message.

Although much research has addressed the retrieval of documents, very little attention has been given to the retrieval of infographics. Yet research has shown that the content of an infographic is often not included in the article's text (Carberry et al., 2006). Thus infographics are an important knowledge source that should be accessible to users of a digital library.

Techniques that have been effective for document or image retrieval are inadequate for the retrieval of infographics. Current search engines employ strategies similar to those used in document retrieval, relying primarily on the text surrounding a graphic and web link structures. But the text in the surrounding document generally does not refer explicitly to the infographic or even describe its content (Carberry et al., 2006). An obvious extension to using the article text would be to collect all the words in an infographic and use it as a bag of words. However, infographics have structure and often a high-level message, and bag of words approaches ignore this structure and message content.

This paper explores the features that should be taken into account when ranking graphics for retrieval in response to a user query. Using a learning-to-rank algorithm on a wide range of features (including structural and content features), we produce a model that performs significantly better than a model that ignores graph structure and content. Analysis of the model shows that features based on the structure and content of graphs are very important and should not be ignored. To our knowledge, our research is the first to take graph structure and content into account when retrieving infographics.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 600–609, Dublin, Ireland, August 23-29 2014.

Figure 1: An Example Infographic

2 Related Work

Information retrieval research has focused on the retrieval of text documents and images. Two popular approaches to text retrieval are the vector space method and probabilistic methods. The vector space method (Dubin, 2004) represents the document and the query each as a vector of weighted words and then uses a similarity function to measure the similarity of each document to the query. Most weighting mechanisms reward words that occur frequently in both the document and query but infrequently in the overall collection of documents. Probabilistic retrieval models instead estimate the probability that a document is relevant to a user query. In recent years, the language modeling approach has shown promise as a retrieval strategy with sound statistical underpinnings (Lv and Zhai, 2009; Manning et al., 2008). In all of the above approaches, query expansion techniques have been used to expand the query with synonyms and related words before ranking documents for retrieval. Work on short document and query expansion have shown improvements in retrieval performance (Arguello et al., 2008; Escalante et al., 2008; Metzler and Cai, 2011).

Work in Content Based Image Retrieval (CBIR) (Datta et al., 2008) has progressed from systems that retrieved images based solely on visual similarity, relying on low-level features such as color, texture and shape ((Flickner et al., 1995; Swain and Ballard, 1991; Smith and Chang, 1997; Gupta and Jain, 1997), among others), to systems which attempt to classify and reason about the semantics of the images being processed (Bradshaw, 2000; Smeulders et al., 2000; Datta et al., 2008). However, images are free-form with relatively little inherent structure; thus it is extremely difficult to determine what is conveyed by an image, other than to list the image's constituent pieces. Most systems that retrieve infographics, such as SpringerImages (http://www.springerimages.com) and Zanran (http://www.zanran.com), are based on textual annotations of the graphics as in image retrieval (Gao et al., 2011) or on matching the user's query against the text surrounding the graphic. However, the structure and content of the graph are not taken into consideration.

In this paper, we focus on natural language queries given that such queries allow users to express their specific information need more clearly than keywords (Phan et al., 2007; Bendersky and Croft, 2009). Previous work on verbose and natural language queries (Bendersky and Croft, 2008; Liu et al., 2013) used probabilistic models and natural language processing techniques to identify the key contents in such queries. Our query processing method not only extracts key entities but also further classifies the extracted key entities into different components using a learned decision tree model.

3 Problem Formulation

Our research is currently limited to two kinds of infographics: simple bar charts and single line graphs. We assume that our digital library contains an XML representation of each graphic that includes 1) the graphic's image, 2) its structural components: the set of independent axis (x-axis) labels[1], the entity being measured on the dependent axis (y-axis), and the text that appears in the graphic's caption, referred to as G_x, G_y, and G_c respectively, and 3) the graphic's intended message G_m and any entities G_f that the

[1]We will refer to the independent axis as the x-axis and the dependent axis as the y-axis throughout this paper.

message focuses on. This paper is not concerned with the computer vision problem of recognizing the bars, labels, colors, etc. in a graphic; other research efforts, such as the work in (Chester and Elzer, 2005; Futrelle and Nikolakis, 1995) are addressing the parsing of electronic images such as bar charts and line graphs.

Prior research on our project has addressed issues that arise in recognizing G_y, G_m, and G_f. The dependent axis of an infographic often does not explicitly label what is being measured, such as *net profit* in Figure 1, and these must be inferred from other text in the graphic. Our prior work (Demir et al., 2007) identified a hierarchy of graphic components in which pieces of the entity being measured might appear; a set of heuristics were constructed that extracted these pieces and melded them together to form what we refer to as a *measurement axis descriptor* and which is G_y. The project's prior work also identified a set of 17 categories of intended message, such as *Rank*, *Relative-difference*, *Maximum*, and *Rising-trend*, that might be conveyed by simple bar charts and line graphs; a Bayesian system (Elzer et al., 2011; Wu et al., 2010) was developed that utilizes communicative signals in a graphic (such as the coloring of one bar differently from the other bars) in order to recognize a graphic's intended message, including both the message category and the parameters of the message such as any focused entity. For example, the intended message of the bar chart in Figure 1 is ostensibly that Toyota has the highest net profit of any of the automobile manufacturers listed; thus its message falls into the *Maximum* message category and its focused entity is *Toyota*.

Our vision is that since graphics have structure and content, the users whose particular information needs could be satisfied by an infographic will formulate their queries to indicate the requisite structure of the desired graphics. Thus we assume the use of full-sentence queries so that the semantics of the query can be analyzed to identify characteristics of relevant graphics. For example, consider the following two queries that contain similar keywords but represent different information needs:

Q_1: *Which countries have the highest occurrence of rare diseases?*
Q_2: *Which rare diseases occur in the most countries?*

These two queries contain almost identical words but are asking for completely different graphics. Query Q_1 is asking for a comparison of countries (independent axis) according to their occurrence of rare diseases (dependent axis) while query Q_2 is asking for a comparison of different rare diseases (independent axis) according to the number of countries in which they occur (dependent axis). In addition, both queries are asking for a graphic with a *Rank* message that ranks countries (query Q_1) or rare diseases (query Q_2) as opposed to a graphic that shows the trend in rare diseases throughout the world.

4 Methodology

To retrieve relevant graphics in response to a user query, the query will first be analyzed to identify requisite characteristics of relevant infographics. We have developed learned decision trees (Li et al., 2013a; Li et al., 2013b) for analyzing a query and identifying the requisite structure of relevant infographics (the content of the independent axis or x-axis and dependent axis or y-axis, referred to as Q_x and Q_y), and the category of intended message and focused entity, if any, (referred to as Q_m and Q_f) that will best satisfy the user's information need.

Given a new user query, it is parsed and noun phrases are extracted. Each query-phrase pair, consisting of a query and an extracted noun phrase, is processed by a decision tree that determines whether the noun phrase represents x-axis content, y-axis content, or neither. Attributes used by this decision tree include whether the main verb of the query is a comparison verb (such as "differ" and "compare") or a trend verb (such as "change" and "decrease"), whether the noun phrase is preceded by a quantity phrase such as "the number of" suggesting that the noun phrase specifies y-axis content of relevant infographics, and whether the noun phrase describes a period of time.

Similarly, another decision tree is constructed to identify the category of graph intended message (such as *Trend* or *Rank*) that the query desires, using a subset of the attributes from the axes decision tree combined with the classification results of the axes decision tree. An example of the reused attributes is the class of the main verb in the user query; for example, a comparison main verb suggests that relevant infographics will convey a comparison-based intended message, such as a *Relative-difference* or *Rank*

intended message. Other attributes include the presence of a superlative or comparative in the query and attributes depending on the identified content of the x and y axes by the axes decision tree, such as the number of x-axis entities, their plurality, and whether an x-axis entity describes a time interval. A third decision tree is constructed for identifying whether a noun phrase describes a specific focused x-axis entity. Then the infographics in the digital library must be rank-ordered according to how well they satisfy the requirements of the user query.

This paper is concerned with identifying the most important features in a metric for rank-ordering the graphics in response to a user query. We experiment with two learning-to-rank algorithms and 56 features that include both general features such as bag of words comparisons and structural and content features. Our hypothesis is that structural and content-based features play an important role in graph retrieval and cannot be ignored. Section 5 discusses the features used in our experiments, Section 6 discusses the learning algorithms, Section 6.1 compares the resultant models with a baseline that uses just general features treating query and graphic each as one bag of words, and Section 6.2 discusses the features that appear most influential in the models.

5 Features

We consider three kinds of features: 1) general features that compare words in the query with words in the graphic, 2) structural features that compare the requisite structure hypothesized from the query with the structure of candidate infographics, and 3) content-based features that compare the requisite message hypothesized from the user query with the intended message of candidate graphics.

Query expansion is a commonly used strategy in information retrieval to bridge the vocabulary gap between terms in a query and those in documents. The basic idea is to expand the original query with terms that are semantically similar to the ones in the query. This addresses the problem encountered when the query uses the word *car* but the document uses the term *automobile*. But retrieval of information graphics presents an additional problem. Consider a query such as *"Which car manufacturer has the highest net profit?"* A graphic such as the one in Figure 1 displays a set of car manufacturers on the x-axis (Toyota, Nissan, etc.) but nowhere in the graphic does the word *car* or a synonym appear. Identifying the ontological category, such as *car* or *automobile*, of these labels is crucial since the user's query often generalizes the entities on the independent axis of relevant graphs rather than listing them.

To expand a given text string s, we use Wikimantic (Boston et al., 2013), a term expansion method that uses Wikipedia articles as topic concepts. A topic concept is a unigram distribution built from words in the Wikipedia article for that topic. A string *s* is interpreted by Wikimantic into a mixture concept that is a weighted vector of topic concepts that capture the semantic meaning of the words in s. Each topic concept is weighted by the likelihood that the concept (Wikipedia article) generates the text string s. The weighted concepts are then used to produce a unigram distribution of words that serve as the expansion of the terms in the string s. One issue in graph retrieval is correlating the requisite x-axis content specified in the user query with the x-axis labels in graphs. A query such as "Which car manufacturer has ... ?" is requesting a graph where "car manufacturers" are listed on the x-axis. Thus we need to recognize individual x-axis words which are often proper nouns (e.g., "Ford", "Nissan", "Honda") as instances of car manufacturers. In the case of labels on the independent axis (such as *Toyota*, *Nissan*, *Honda*, etc.), words such as *car* or *automobile* are part of the produced unigram distribution — that is, as a side effect, the ontological category of the individual entities becomes part of the term expansion.

We use Wikimantic to interpret and expand each of the graph components G_x, G_y, G_f, and G_c. The expansion of the graph components (as opposed to the typical expansion of the query) accomplishes two objectives: 1) it addresses the problem of sparse graphic text by adding semantically similar words and 2) it addresses the problem of terms in the query capturing general classes (such as *car* or *automobile*) when the graphic instead contains an enumeration of members of the general class. Expansion of the words in the graphics, unlike query expansion, has the added advantage that it is completed in advance and off-line.

5.1 General Features

Our general feature set includes 17 general features capturing a variety of different kinds of relevance scorings between two bags of words consisting respectively of words from the user query and words from the candidate infographic:

- GF_1: A modified version of Okapi-BM25 (Fang et al., 2004) calculated as:

$$\text{Okapi-BM25 Score} = \sum_{w \in Q} \log \frac{|D|+1}{df_w+1} \cdot \frac{tf_w \cdot (1+k_1)}{tf_w+k_1}$$

  where Q is a query, $|D|$ is the number of graphs in the digital library, w is a query word in Q, df_w is the frequency of graphs containing word w in the digital library, tf_w is the frequency of word w in the text expansion of the given graphic, and k_1 is a parameter that is typically set to 1.2. Okapi-BM25 is a bag-of-words ranking function used in many information retrieval systems. Our modified version of Okapi-BM25 addresses the problem of negative values that can occur with the original Okapi formula. In addition, our formula does not take text length or query term frequency into account since graphics have relatively similar amounts of text and most terms in a query occur only once.

- GF_2: The term frequency-inverse document frequency (tf-idf) value of query words that appear in the expanded graphic.

- GF_3: The maximum, minimum, and arithmetic mean of the term frequency (tf) of query words that appear in the expanded graphic.

- GF_4: The maximum, minimum, and arithmetic mean of inverse document (graphic) frequency (idf) of query words that appear in the expanded graphic.

5.2 Structural Features

Our structural feature set includes 35 features: 17 that address how well a graphic's x-axis (independent axis) relates to the requisite x-axis content hypothesized from the user's query and 18 that address how well a graphic's y-axis (dependent axis) content captures the requisite dependent axis content hypothesized from the query. The following are a few of the x-axis features:

- SFX_1: The Okapi-BM25 value using the same modified formula as for general features, given the query x-axis words and the text expansion of the x-axis labels in the graphic.

- SFX_2: The tf-idf of x-axis words hypothesized from the query that appear in the expansion of the x-axis labels in the graphic.

- SFX_3: The maximum, minimum, and arithmetic mean of tf of x-axis words hypothesized from the query that appear in the expansion of the x-axis labels in the graphic.

- SFX_4: The maximum, minimum, and arithmetic mean of idf of x-axis words hypothesized from the query that appear in the expansion of the x-axis labels in the graphic.

The y-axis features (SFY_1, SFY_2, SFY_3, and SFY_4) include the same relevance measurements as used for the x-axis features; for example, feature SFY_1 captures the Okapi-BM25 score for the y-axis content hypothesized from the query and the text expansion of the graphic y-axis words, and feature SFY_2 is the tf-idf score for the y-axis content hypothesized from the query and the expansion of the graphic y-axis words. One additional feature that is specific to the y-axis is:

- SFY_5: The posterior probability of the Wikimantic (Boston et al., 2013) mixture concept[2] for the y-axis words hypothesized from the query, given the Wikimantic mixture concept representing the y-axis words in the graph, referred to as $p(Q_y|G_y)$. Both query y-axis words and the graphic y-axis

[2] A Wikimantic mixture concept is a set of weighted concepts (Boston et al., 2013).

descriptor are each interpreted by Wikimantic into a mixture concept, M_{qy} and M_{gy} respectively. Recall from the introduction to Section 5 that a mixture concept is a weighted vector of topic concepts that defines the semantic meaning of a term or set of terms. For example, the mixture concept for the country China is represented by a vector of topic concepts such as *"China"*, *"People's Republic of China"*, *"Mainland China"*, and so on. Wikimantic estimates the probability of a concept given another concept by the amount of overlapping words between the two concepts. For example, the topic concept for the country *"United States"* is likely to contain similar words to the concept for *"China"*, such as the words *"country"*, *"nation"*, *"region"*, *"capital"*, *"GDP"*, etc. Therefore the probability of *United States* given *China* is likely to be higher than that of *United States* given the topic *"rugby"*.

5.3 Content Features

Our content feature set contains four features that address how well the intended message of a graphic captures the requisite message content hypothesized from the user's query. Ideally, a relevant graphic's intended message G_m will match the message category Q_m hypothesized from the user's query. When the two do not match exactly, we use a hierarchy of message categories and the concept of *relaxation* as the paradigm for estimating how much perceptual effort would be required to extract the message specified by the query from the graphic. For example, suppose that the query requests a *Rank* message; graphics with *Rank* messages will convey the rank of a specific entity by arranging the entities in order of value and highlighting in some way the entity whose rank is being conveyed. Graphics with a *Rank-all* intended message will convey the rank of a set of entities without highlighting any specific entity; the *Rank-all* message category appears as a parent of *Rank* in the message hierarchy since it is less specific than *Rank*. Although one can identify the rank of a specific entity from a graphic whose intended message is a *Rank-all* message, it is perceptually more difficult since one must search through the graph for the entity whose rank is desired. By moving up or down the message hierarchy from Q_m to G_m, Q_m is relaxed to match different G_m. The greater the degree of relaxation involved, the less message-relevant the infographic is to the user query. The four content-based features are:

- CF_1: Whether the message category Q_m hypothesized from the user's query matches exactly the intended message category G_m of the graphic.
- CF_2: The amount of relaxation needed to relax the message category Q_m hypothesized from the user's query so that it matches the intended message category G_m of the graphic.
- CF_3: The Okapi-BM25 value given the intended message focused entity Q_f (if any) hypothesized from the user's query and the focused entity G_f in the graphic, if any.
- CF_4: The Okapi-BM25 value given the intended message focused entity Q_f(if any) hypothesized from the user's query and the non-focused x-axis entities G_{nf} in the graphic.

6 Constructing a Ranking Model for Graph Retrieval

Learning-to-rank algorithms (Liu, 2009) construct a learned model that ranks objects based on partially ordered training data. Tree-based ensemble methods have been shown to be very effective (Chapelle and Chang, 2011). We experimented with two state-of-the-art tree-based learning-to-rank algorithms as implemented in the RankLib library (http://people.cs.umass.edu/vdang/ranklib.html): Multiple Additive Regression Trees abbreviated as MART (Friedman, 2001) and Random Forest (Breiman, 2001).

A human subject experiment was performed to collect a set of 152 full sentence user queries from five topics. The queries were collected from 5 different tasks and covered a variety of topics involving companies. Two sample queries are "What credit card company made the most money in 2008?" and "How does Avis rank compared to other car rental companies in revenue?". We used the collected queries to search on popular commercial image search engines to get more infographics from the same topics. These commercial search engines include Google Image, Microsoft Bing Image Search, and Picsearch. This produced a set of 257 infographics that are in the topics of the collected queries. Each

query-infographic pair was assigned a relevance score on a scale of 0-3 by an undergraduate researcher. A query-infographic pair was assigned 3 points if the infographic was considered highly relevant to the query and 0 points if it was irrelevant. Query-infographic pairs where the graphic was somewhat relevant to the query were assigned 1 or 2 points, depending on the judged degree of relevance of the graphic to the query. This produced a corpus for training and testing.

Using MART and Random Forest, we developed four models from all 56 features, including the structural and content features. Two of the models were built using our learned decision trees (Li et al., 2013b; Li et al., 2013a) to analyze the queries and hypothesize the requisite x-axis content, y-axis content, message category, and focused entity (if any); see the second row of Table 1. Since the learned decision trees are not perfect, the other two models were built from hand-labelled data; see the last row of Table 1. In addition, two baseline models were constructed using only the general features and omitting the structural and content-based features.

6.1 Evaluating the Models

Normalized Discounted Cumulative Gain (NDCG) (Järvelin and Kekäläinen, 2002) is used to evaluate the retrieval result. Table 1 displays the NDCG@10 results. In each case, we averaged together the NDCG results of 10 runs using the Bootstrapping Method (Tan et al., 2006) in which the query data set is sampled with replacement to select 152 queries; these 152 queries, and for each query the relevance judgements assigned to each of the graphics, comprised the training set, with the unselected queries and their relevance judgements comprising the testing set. The Bootstrapping method is a widely used evaluation method for small datasets. Typically, approximately 63% of the dataset is selected for the training set (with some items appearing more than once in the training set) and 37% for the testing set. The second row of Table 1 provides results when each query is processed by our learned decision trees to extract the structural content and message category that the query specifies. However, the decision trees are imperfect. To determine whether our system could do even better if the decision trees were improved, the third row of Table 1 reports results when each query was hand-labelled with the correctly extracted structural and message content.

The models using all 56 features produced significantly better results than the baseline model that used just the general features, indicating that structural and content-based features are very important and must be taken into account in graph retrieval. In addition, the models built from the hand-labelled data produced better results than the models where the structural and content features were automatically extracted from the queries using the learned decision trees; this suggests that improving the decision trees that process the queries would improve the accuracy of the learned graph retrieval models. In some cases, the Random Forest learned model performed better than the MART model, but the improvement was not significant. The experimental results show that both MART and Random Forest using all 56 features, either using the hand-labelled query data or decision tree query data, provide significantly better results than the baseline approach ($p<0.0005$).

Algorithm	MART	Random Forest
Baseline	0.4943	0.4935
Decision Tree Query Data	0.6239	0.6258
Hand-labelled Query Data	0.6723	0.6758

Table 1: NDCG@10 Results

Figure 2 displays the NDCG@k results for different values of k. The bottom solid line and the line composed of triangles depict the baseline results, the middle dashed line and the line composed of circles depict the results using the decision tree query data, and the top solid line and the line composed of triangles depict the results using the hand-labelled data. All of the models improve as k increases. Most important, both our MART and Random Forest models constructed from all 56 features perform much better than the baseline models for all values of k. Thus we conclude that the use of structural and content features helps in selecting the most relevant graphic as well as the most relevant sets of graphics.

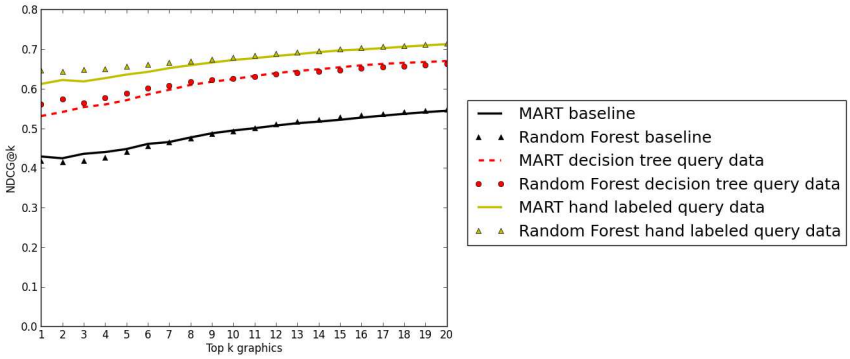


Figure 2: NDCG@k for Various Values of n

6.2 Analysis of Influential Features

In both MART and Random Forest, features that are used at the top levels of each tree are more important in ranking a graphic than features that appear lower in the tree. We analyzed the importance of each of the 56 features based on the level in each tree where the feature is first used. 70% of the top ten most important features in the trees produced by both MART and Random Forest were structural or content features. The most influential two features in trees produced by MART were SFY_5 which captures $p(Q_y \mid G_y)$ and SFX_2 which captures the tf-idf of x-axis words hypothesized from the query that appear in the expansion of the x-axis labels in the graphic. Although these two features were not the two most influential features in the trees produced by Random Forest, they did appear among the top 5 features. Two content-based features appeared among the top ten most important features: CF_3 which captures the relevance of the focused entity Q_f (if any) hypothesized from the query to the focused entity G_f (if any) in the graphic and CF_4 which captures the relevance of the focused entity Q_f (if any) hypothesized from the query to the non-focused entities G_{fx} in the graphic. The content features CF_1 and CF_2 that measure relevance of the message category hypothesized from the query to the intended message category in a candidate graphic appeared among the top 20 features but not among the top 10 features. Further inspection of the trees and analysis of the queries and graphics leads us to believe that message category relevance is influential in refining the ranking of graphics once graphics with appropriate structural content have been identified. Our future work will examine these two features more closely and determine whether modifications of them, or changes in how they are used, will improve results.

Based on these results, we conclude that structural and content-based features are important when ranking infographics for retrieval and must be taken into account in an effective graph retrieval system.

7 Conclusion and Future Work

To our knowledge, no other research effort has considered the use of structural and content-based features when ranking graphics for retrieval from a digital library. We developed learned models that take into account how well the structure and content of an infographic matches the requisite structure and content hypothesized from the user query, and showed that these models perform significantly better than baseline models that ignore graph structure and message content. In addition, an analysis of the learned models showed which structural and content features were most influential. In our future work, we will improve our methods for hypothesizing requisite features of relevant graphics and will analyze our relaxation metric to determine whether an improved metric will play a more influential role in ranking graphics for retrieval.

Acknowledgements

This work was supported by the National Science Foundation under grant III-1016916 and IIS-1017026.

References

Jaime Arguello, Jonathan L Elsas, Jamie Callan, and Jaime G Carbonell. 2008. Document representation and query expansion models for blog recommendation. *ICWSM*, 2008(0):1.

Michael Bendersky and W Bruce Croft. 2008. Discovering key concepts in verbose queries. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 491–498. ACM.

Michael Bendersky and W Bruce Croft. 2009. Analysis of long queries in a large scale search log. In *Proceedings of the 2009 workshop on Web Search Click Data*, pages 8–14. ACM.

Christopher Boston, Hui Fang, Sandra Carberry, Hao Wu, and Xitong Liu. 2013. Wikimantic: Toward effective disambiguation and expansion of queries. *Data & Knowledge Engineering*.

Ben Bradshaw. 2000. Semantic based image retrieval: a probabilistic approach. In *Proceedings of the eighth ACM international conference on Multimedia*, pages 167–176. ACM.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Sandra Carberry, Stephanie Elzer, and Seniz Demir. 2006. Information graphics: an untapped resource for digital libraries. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 581–588. ACM.

Olivier Chapelle and Yi Chang. 2011. Yahoo! learning to rank challenge overview. In *Yahoo! Learning to Rank Challenge*, pages 1–24.

Daniel Chester and Stephanie Elzer. 2005. Getting computers to see information graphics so users do not have to. In *Foundations of Intelligent Systems*, pages 660–668. Springer.

S. Clark and J.R. Curran. 2007. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552.

Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. 2008. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (CSUR)*, 40(2):5.

Seniz Demir, Sandra Carberry, and Stephanie Elzer. 2007. Effectively realizing the inferred message of an information graphic. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 150–156.

David Dubin. 2004. The most influential paper gerard salton never wrote.

Stephanie Elzer, Sandra Carberry, and Ingrid Zukerman. 2011. The automated understanding of simple bar charts. *Artificial Intelligence*, 175(2):526–555.

Hugo Jair Escalante, Carlos Hernández, Aurelio López, Heidy Marín, Manuel Montes, Eduardo Morales, Enrique Sucar, and Luis Villaseñor. 2008. Towards annotation-based query and document expansion for image retrieval. In *Advances in Multilingual and Multimodal Information Retrieval*, pages 546–553. Springer.

Hui Fang, Tao Tao, and ChengXiang Zhai. 2004. A formal study of information retrieval heuristics. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 49–56, New York, NY, USA. ACM.

Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, et al. 1995. Query by image and video content: The qbic system. *Computer*, 28(9):23–32.

Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 10.

Robert P Futrelle and Nikos Nikolakis. 1995. Efficient analysis of complex diagrams using constraint-based parsing. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 2, pages 782–790. IEEE.

Y. Gao, M. Wang, H. Luan, J. Shen, S. Yan, and D. Tao. 2011. Tag-based social image search with visual-text joint hypergraph learning. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1517–1520. ACM.

Amarnath Gupta and Ramesh Jain. 1997. Visual information retrieval. *Communications of the ACM*, 40(5):70–79.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, October.

Zhuo Li, Matthew Stagitis, Sandra Carberry, and Kathleen F. McCoy. 2013a. Towards retrieving relevant information graphics. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 789–792, New York, NY, USA. ACM.

Zhuo Li, Matthew Stagitis, Kathleen McCoy, and Sandra Carberry. 2013b. Towards finding relevant information graphics: Identifying the independent and dependent axis from user-written queries.

Jingjing Liu, Panupong Pasupat, Yining Wang, Scott Cyphers, and Jim Glass. 2013. Query understanding enhanced by hierarchical parsing structures. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 72–77. IEEE.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331.

Yuanhua Lv and ChengXiang Zhai. 2009. Positional language models for information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306. ACM.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.

Donald Metzler and Congxing Cai. 2011. Usc/isi at trec 2011: Microblog track. In *TREC*.

Nina Phan, Peter Bailey, and Ross Wilkinson. 2007. Understanding the relationship of information need specificity to search query length. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 709–710. ACM.

Arnold WM Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. 2000. Content-based image retrieval at the end of the early years. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(12):1349–1380.

John R Smith and Shih-fu Chang. 1997. Querying by color regions using the visualseek content-based visual query system. *Intelligent multimedia information retrieval*, 7(3):23–41.

Michael J Swain and Dana H Ballard. 1991. Color indexing. *International journal of computer vision*, 7(1):11–32.

Pang-Ning Tan, Michael Steinbach, Vipin Kumar, et al. 2006. Introduction to data mining. *WP Co*.

Peng Wu, Sandra Carberry, Stephanie Elzer, and Daniel Chester. 2010. Recognizing the intended message of line graphs. In *Diagrammatic Representation and Inference*, pages 220–234. Springer.

Inducing Discourse Connectives from Parallel Texts

Majid Laali and Leila Kosseim
Department of Computer Science and Software Engineering,
Concordia University, Montreal, Quebec, Canada
{m_laali, kosseim}@cse.concordia.ca

Abstract

Discourse connectives (e.g. *however, because*) are terms that explicitly express discourse relations in a coherent text. While a list of discourse connectives is useful for both theoretical and empirical research on discourse relations, few languages currently possess such a resource. In this article, we propose a new method that exploits parallel corpora and collocation extraction techniques to automatically induce discourse connectives. Our approach is based on identifying candidates and ranking them using Log-Likelihood Ratio. Then, it relies on several filters to filter the list of candidates, namely: Word-Alignment, POS patterns, and Syntax. Our experiment to induce French discourse connectives from an English-French parallel text shows that Syntactic filter achieves a much higher MAP value (0.39) than the other filters, when compared with LEXCONN resource.

1 Introduction

Discourse relations are often categorized as being *implicit* or *explicit* depending on how they are marked linguistically (Prasad et al., 2008). *Implicit* relations between two text spans are inferred by the reader even if they are not explicitly connected through lexical cues. On the other hand, *explicit* relations are explicitly identified with syntactically well-defined terms, so called *discourse markers* or *discourse connectives* (DCs). A list of DCs is a valuable resource to help the automatic detection of discourse relations in a text. Discourse parsers (e.g. (Lin et al., 2010)) often use DCs as a powerful distinguishing feature to tag discourse relations (Pitler and Nenkova, 2009). A list of DCs is also instrumental in generating annotated training data which, in turn, is critical for training data-driven parsers (Prasad et al., 2010).

In this article, we propose an automatic method to induce a list of DCs for one language from a parallel corpus. We present an experiment in inducing a French DC list from an English-French parallel text. Our approach is based on the hypothesis that discourse relations are retained during the translation process. Therefore, if a reliable discourse tagger exists in a language, we can produce a corpus with discourse annotation labels in any language that has a parallel text with that language. Fortunately, according to (Versley, 2011), in English, the discourse usage of DCs can be automatically identified and labeled with their relation with 84% precision; a result that is close to the reported inter-annotator agreement. Moreover, with the advancement of statistical machine translation, today English parallel corpora for several languages are publicly available.

Although we can expect little variability in the usage of discourse relations in parallel texts, this is not the case for DCs. In other words, translated texts may not always reproduce DCs of the source texts. Since discourse relations can be conveyed either explicitly with a DC or implicitly, a translator may choose to remove explicit DCs in the source text and express the relation in the translated text implicitly. In fact, Meyer and Webber (2013) has shown that DCs drop out up to 18% of the times in human reference translations.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 610–619, Dublin, Ireland, August 23-29 2014.

To alleviate noisy data (i.e. sentences whose DCs are dropped during the translation), we have been inspired by work in collocation extraction (e.g. (Seretan, 2010)). As such, our approach consists of two main steps: candidate identification and candidate ranking and filtering. We have used several types of information to filter out incorrect DC candidates and used Log-Likelihood ratio to rank them. These filters include Part-of-speech tags, syntactic tree and word-alignment. Our results show that syntactic information outperforms the other filtering methods for the DC identification task.

This paper is organized as follow. Section 2 reviews related work. Section 3 describes our approach to extract DCs from a parallel text. Section 4 reports detailed experimental results, and finally Section 5 presents our conclusion and future work.

2 Related Work

Currently, publicly available lists of DCs already exist for English (Knott, 1996), Spanish (Alonso Alemany et al., 2002), German (Stede and Umbach, 1998), and French (Roze et al., 2012). Typically, these lists have been manually constructed by applying systematic linguistic tests to a list of potential DCs. For example, (Roze et al., 2012) gathered a potential list of DCs (about 600 expressions) from English DC translations and various lists of subordinate conjunctions and prepositions. Then, they applied syntactic, semantic, and discourse tests to filter this initial list and identify DCs and their associated relations.

A list of DCs can also be created automatically by analyzing lexically-grounded discourse annotated corpora. The Penn Discourse Tree Bank (PDTB) (Prasad et al., 2008) is the largest resource to date that provides a discourse annotated corpus in English. In this corpus, discourse relations between two text spans are labeled with a DC. If a discourse relation is expressed without any explicit DC, an inferred DC which conveys the same discourse relation has been inserted between the text spans. This approach has been widely adopted to create discourse tree banks in several other languages such as Turkish (Zeyrek et al., 2010), Chinese (Zhou and Xue, 2012), Arabic (Al-Saif and Markert, 2010), Czech (Mladová et al., 2008), and Hindi (Oza et al., 2009).

Several work have already investigated the use of discourse relations in machine translation (e.g. (Meyer and Webber, 2013; Meyer, 2011)). Others have attempted to generate discourse annotated corpora from parallel corpora (e.g. (Cartoni, 2013; Meyer, 2011; Popescu-Belis et al., 2012; Versley, 2010; Zhou et al., 2012)). Among these, the most similar approach to ours is Versley (2010) who has projected English DCs to their counterparts in German in a parallel corpus. Doing this, he produced a corpus where discourse vs. non-discourse usage of German DCs were annotated and built a discourse parser from the corpus. Although Versley (2010) used a list of DCs in generating the dataset, he also tried to automatically induce the DCs from his corpus. However, Versley (2010) did not explicitly evaluate his list of DCs, but rather focused on his parser. The main difference between our work and Versley (2010) is that he has solely employed word alignment to find DCs, which as mentioned in his paper, is not sufficient to align discourse connectives. In contrast, we have used and compared three approaches for inducing a DC list: word-alignment, POS patterns and syntactic information.

3 Method

Our approach to the extraction of DCs consists of two steps. The first step is the preparation of the parallel corpus with discourse annotations; the next step is the mining of the parallel corpus to identify DCs.

3.1 Preparing the Parallel Corpus

Our experiment has focused on building a French list of DCs from English. In order to build the English-French parallel corpus with discourse annotations, we used the Europarl corpus (Koehn, 2005). The Europarl corpus contains sentence-aligned texts in 21 European languages that have been extracted from the proceeding of the European parliament. For our study, we have only considered the English-French part of this corpus.

To label discourse relations in the parallel text, we have automatically parsed the English side of the parallel text and assumed that the same relation existed in the French translation. Although this

assumption is not directly addressed in previous work, it has been implicitly used by many (e.g. (Cartoni, 2013; Meyer et al., 2011; Popescu-Belis et al., 2012; Versley, 2010; Prasad et al., 2010)). In particular, Prasad et al. (2010) have suggested to the use of the back-translation technique (translating a text from language A to language B, then back translate the same text from language B to language A again) to discover new DCs. In this work, the authors have implicitly assumed that the discourse relations of the initial text are maintained in the back-translation. We argue that since discourse relations are semantic and rhetorical in nature, they usually transfer from source language to target language. We have used the PDTB-style End-To-End Discourse parser (Lin et al., 2010) to parse the English text. This parser has been trained on Section 02-22 of the PDTB corpus (Prasad et al., 2008) and can identify and label a DC with its relation with 81.19% precision when tested on Section 23 of the PDTB.

After tagging the English text, we have only kept parallel sentences whose English translation had exactly one discourse relation. This was done to ensure that no ambiguity would exist in the discourse relation of the French sentences, once we transfer the discourse relation from English to French. In other words, we can label each French sentence with a single discourse relation, that of its English translation. In addition, we have also removed sentences whose discourse relations were expressed implicitly. Although the (Lin et al., 2010) parser is able to identify both implicit and explicit discourse relations, we have only considered relations expressed with a DC. This has been done, since not only the precision of the parser in detecting discourse relation in the absence of DC is very low (24.54%), but also we would not expect implicit relations to help us to identify DCs in French. In other words, a translator only occasionally inserts DCs in a translation and therefore we would not expect that too many DCs would exist in the translation of sentences with an implicit discourse relation.

Table 1 provides statistics on the original English-French Parallel Corpus and the corpus extracted with exactly one explicit discourse relation per sentence. Initially, the Europarl corpus contained 2,054K sentences (57 million and 63 million words in the English and the French sides respectively). However, after removing the sentences with more than one discourse relation, the corpus was reduced to 543K sentences automatically annotated with discourse relations. The English part of these sentences contains 14 million words, while the French part contains 15 million words.

	# Parallel Sentences	**# English Words**	**# French Words**
Original Europarl Corpus	2,054K	57M	63M
Extracted Corpus	543K	14M	15M

Table 1: Statistics on the Parallel Corpora

Although this new annotated corpus represents only 26% of the original French Europarl, the corpus still represents a large annotated corpus with respect to existing discourse-annotated corpora. For example, the corpus is almost 30 times bigger than PDTB. Therefore, due to the large size of the corpus, it can be expected that eventual errors in the corpus (e.g. sentences whose discourse relations have been changed during the translation) should not affect the results significantly.

3.2 Mining the Parallel Corpus

Once the aligned corpus has been built, we have mined the French side to identify DCs. To do this, we have produced an initial list of DC candidates from the corpus; then we have ranked the list based on the Log-Likelihood Ratio (LLR). Finally, we have applied several filters to refine the final list.

To produce the initial DC candidates, we have extracted n-grams (unigrams, bigrams, ..., and six-grams) from all French sentences as a potential candidate for a DC. Then, we have stored each potential candidate with its discourse relation as a pair. For example, in sentence (1) below, the English sentence contains an ALTERNATIVE relation signaled with the "*So*" English DC. We have therefore produced the pairs "{ALTERNATIVE, *Donc*}", "{ALTERNATIVE, *Donc d*}", "{ALTERNATIVE, *Done d un*}", etc. from its corresponding French sentence.

(1) So, judicially, something needs to be done./ALTERNATIVE
Donc, d'un point de vue judiciaire, il convient de prendre des mesures.

Once the initial list of DC candidates has been extracted, we have used the LLR to rank the DCs[1]. LLR evaluates association strength between a pair of events based on their frequency. This measure, for example, has been largely used in collocation extraction (e.g. (Seretan, 2010)). According to Evert (2004), LLR is equivalent to the average mutual information that one event conveys about the other. For the sake of completeness, Figure 1 shows the formula used to calculate LLR for two binary random variables X and Y. Note that in Figure 1, O refers to the observed frequencies, E refers to the expected frequencies and N refers to the total number of observations.

$$LLR(X,Y) = 2 \times \sum_{i=1}^{2} \sum_{j=1}^{2} O_{ij} \times log(\frac{O_{ij}}{E_{ij}})$$

$$E_{ij} = \frac{\sum_{k=1}^{2} O_{ik} \times \sum_{k=1}^{2} O_{kj}}{N}, \; N = \sum_{i=1}^{2} \sum_{j=1}^{2} O_{ij}$$

	$Y = v$	$Y = \neg v$
$X = u$	O_{11}	O_{12}
$X = \neg u$	O_{21}	O_{22}

Figure 1: The formula used to calculate LLR.

In our configuration, our pairs of events consist of the observation of a discourse relation and a DC candidate. We have computed contingency tables of frequencies of these pairs from the French corpus and then used the NSP package (Pedersen et al., 2011) to calculate the LLR for each candidate to rank them. Once the initial list of DCs has been ranked, we have experimented with several types of filters to refine it.

Frequency Filter: This simple filter tries to account for the fact that low frequent events may affect the reliability of the LLR measure. Therefore, as a simple baseline filter, we have removed DC candidates that appear less than a certain number of times in the French corpus.

Word-Alignment Filter: This filter removes any DC candidate that does not align with any part of an English DC. In other words, this filter keeps any consecutive words in the French text if at least one of its composing words aligns to at least one word of an English DC when using a word-alignment model. A word-alignment model maps each word in the target text to its translation in the source text (creating an n-to-one mapping). Therefore, two word-alignment models can be produced (i.e. when the target text is French (En2Fr) or when the target text is English (Fr2En)). In addition, Och and Ney (2003) have also presented another word-alignment model called Intersect word-alignment that uses a heuristic to combine En2Fr and Fr2En word alignments. Figure 2 presents the later alignment for two parallel sentences. An alignment between two words is shown by a line connecting them. For example, in these sentences, the connective "*therefore*" is aligned to the three French words "*raison pour laquelle*". We have used MGIZA++ (Gao and Vogel, 2008) to generate En2Fr and Fr2En word-alignments; then used Moses (Koehn et al., 2007) to compute the Intersect word alignment. In this article, we only consider Intersect word-alignment, as it is able to map n-to-m mapping[2].

Syntactic Filters: DCs are defined as syntactically well-defined terms (Prasad et al., 2008). The syntactic filters exploit this property and remove any constituent that is not categorized as a DC. In other words, these filters keep only Prepositional Phrases (PP), Coordinate Phrases (CP) or Adverbial Phrases (ADVP). We have implemented two types of Syntactic Filters. The first one (called **POS Filter**) uses predefined POS patterns to filter out incorrect candidates. We have manually defined POS patterns based on an analysis of the French DCs in the LEXCONN resource (Roze et al., 2012). Table 2 shows the POS patterns we have used along with an example. The second approach (called **Syntax Tree Filter**) makes use of Syntax Trees to filter unlikely syntactic combinations. Therefore, after parsing all the

[1]We have also used other association measures, such as PMI, t-score test, and Chi-square test, but LLR achieved the best results in terms of mean average precision.

[2]We have also experimented with other word-alignments but their performances were not better. The Intersect model outperformed the Fr2En word-alignment model and acheived similar results as the En2Fr word-alignment model.

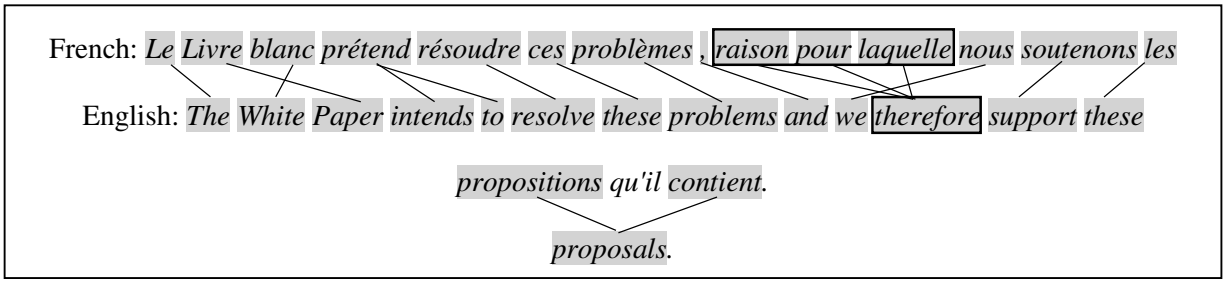


Figure 2: Example of Word-Alignments between English and French Texts.[3]

French sentences, the Syntax Tree Filter only kept PPs, CPs and ADVPs. We have used the Stanford POS Tagger (Toutanova et al., 2003) and the Stanford PCFG Parser (Green et al., 2011) for POS tagging and parsing the French text, respectively.

POS Pattern	Example	POS Pattern	Example
ADV	alors	P ADV	après tout
C	et	P N	par exemple
P	comme	P P	avant de
ADV C	encore que	V C	considérant que
ADV P	en outre	N D P	de ce fait
C C	parce que	P N P	de manière à
N P	histoire de	P D N	dans ce cas

Table 2: POS Patterns Used in the POS Filter.

3.3 Gold Dataset

To evaluate our final ranked list of French DCs candidates and compare the four filters, we have used the LEXCONN dataset (Roze et al., 2012). This manually constructed dataset includes 467 French discourse connectives with their syntactic categories and the discourse relations that they express[4]. Table 3 provides some statistics about LEXCONN. We also provide statistics about the DCs in PDTB for comparative purposes. Each row of Table 3 indicates the number of DCs and the average number of relations per DC in parenthesis. For example, in LEXCONN, 70 DCs are unigrams and on average they indicate 1.66 different discourse relations. Table 3 also shows statistics on the length of DCs (in number of words). It is interesting to note that French tends to have longer DCs than English. Indeed LEXCONN contains 69 DCs that contain four words (e.g."*au même titre que*", "*dans l'espoir de*", etc.) while there are only 4 four-gram DCs in English (e.g. "*as it turns out*" or "*on the other hand''*).

Although there are fewer relations in PDTB, English DCs tend to be more ambiguous. As Table 3 shows, each English DC conveys 3.05 relations on average, while this number is 1.29 for French DCs. We also notice that the longer the DC, the less ambiguous it is in terms of discourse relations it can convey. For example, unigram DCs in French convey on average 1.66 relations, however the number of relations decreases when the length of the DC increases, so that for a trigram DC, on average, there are 1.22 relations.

3.4 Evaluation Metric

Since our task is very similar to a collocation extraction task, we have used a similar evaluation methodology to evaluate our results. We have modeled the task of inducing DCs as a binary classification and tried to evaluate it using precision and recall. In other words, by choosing a threshold for LLR, we can

[3]The examples in this figure are taken from the Europarl corpus.

[4]LEXCONN has 431 DCs, however if we consider different spelling of each DC (e.g. "*alors que''* and "*alors qu'*"), the number increases to 467.

[5]As the parser labels relations at the second level of the PDTB hierarchy, we here report only the number of second level relations.

	LEXCONN (French)	PDTB DCs (English)
# Discourse relation	29	16[5]
# Total number of DCs	467 (1.29)	133 (3.05)
# Unigram DCs	70 (1.66)	76 (3.50)
# Bigram DCs	169 (1.25)	33 (2.70)
# Trigram DCs	139 (1.22)	18 (2.11)
# Four-gram DCs	69 (1.17)	4 (2.50)
# Five-gram DCs	14 (1.07)	1 (1.00)
# Six-gram DCs	5 (1.20)	0 (-)
# Seven-gram DCs	1 (2.00)	1 (1.00)

Table 3: Statistics on Discourse Connectives in LEXCONN and PDTB v.2.

label each potential DC candidate as "*DC*" if its LLR is above the threshold or "*non-DC*" otherwise. However, choosing the LLR threshold depends on the application and there is no principled way to determine an ideal value for the threshold. Therefore, we measured the performance of the ranked list of DCs with 11-point interpolated average precision curve (Manning et al., 2008). This curve shows highest precision at the 11 recall levels of 0.0, 0.1, 0.2, ..., 1.0. Using this methodology, we can evaluate the ranked list without considering any threshold.

In addition to the 11-point interpolated average precision, we also used Mean Average Precision (MAP) (Manning et al., 2008). As Pecina (2010) noted for the evaluation of collocation extraction, since the precision is not reliable at low recall levels and changes frequently at high recall levels, we only consider average precision in the interval of $<0.1, 0.9>$when we are calculating MAP.

Another consideration when evaluating our final ranked lists is how to evaluate DC fragments. For example, when evaluating the candidate "*à ce point*", we have to label it as a wrong DC because it is not repertoried in LEXCONN. However, it is a segment of the French DC "*à ce point que*" and only one word is missing in the expression. This issue has been also addressed in the field of collocation extraction; in particular, Kilgarriff et al. (2010) suggested to consider a partial collocation as a true positive, since it signals the presence of the longer collocation. However, this "was not a decision that human evaluators were comfortable with" (Kilgarriff et al., 2010). In our evaluation, we have used two approaches to evaluate fragment DCs. In the first approach, the Exact Match approach, we have considered fragment DCs as an incorrect DC. In the other approach, the Exclude-From-The-List approach, we have removed them from our list, so that when we analyzed the find list, they do not appear as an incorrect DC.

4 Results

To evaluate the DC extraction approach, we first analyzed the candidate generation step without any filtering. Table 4 provides the frequency distribution of LEXCONN's DCs in the annotated corpus. This table shows that the longer the DCs, the less frequent they are in our corpus. For example, all one-word DCs of LEXCONN appear in the corpus, while 21% of LEXCONN's five-gram and 60% of LEXCONN's six-gram DCs never occur in the corpus. Overall, 14% of all LEXCONN DCs do not appear in the corpus.

Recall that the Frequency filter removes DCs that do not appear enough times in order to use LLR to rank candidates. In our experiment, we used a minimum threshold of 10 for this filter. Therefore, the filter removed additional 20% DCs, so that overall only 66% of LEXCONN's DCs are considered in the corpus. Most of these removed DCs are not common or rather formal expressions in French such as "*conséquemment*", "*hormis que*", "*tout bien considéré*". However, several more informal DCs commonly used in French were also removed, especially in the trigram and more groups of DCs (e.g. "*à part ça*").

Once we calculated the number of available DCs in the corpus, we evaluated the ranked list of DCs after applying each filter. Table 5 shows the MAP values of each filter using both the Exact Match

	freq > 10	**$10 \geq$ freq > 0**	**freq $= 0$**
# Unigram DCs	93%	7%	0%
# Bigram DCs	76%	16%	8%
# Trigram DCs	60%	24%	16%
# Four-gram DCs	36%	31%	33%
# Five-gram DCs	50%	29%	21%
# Six-gram DCs	20%	20%	60%
Overall	66%	20%	14%

Table 4: Distribution of LEXCONN DCs in the Extracted Corpus.

Filter	**MAP with Exact Match**	**MAP with Exclude-From-The-List**
LLR only	0.06	0.07
LLR + Word-Alignment Filter	0.10	0.12
LLR + POS Pattern Filter	0.12	0.14
LLR + Syntax Tree Filter	0.39	0.44

Table 5: MAP of Each Filter.

and Exclude-From-The-List approaches to judge fragment DCs[6] (see Section 3.4). With all four filters, we first used the Frequency Filter and then ranked the candidates using LLR. Our results show that using the POS Pattern Filters outperforms the Word-Alignment filter. For example, if we consider the Exact Match metric, the MAP value of the Word-Alignment is 0.10 while it is 0.12 for the POS-Pattern Filter. As Table 5 shows, the best MAP values are achieved using the Syntax Tree Filter. For the rest of document, we only consider the Exclude-From-The-List approach to judge fragment DCs, since we would like to focus on other sources of errors in the ranked list of DCs in addition to the fragment DCs.

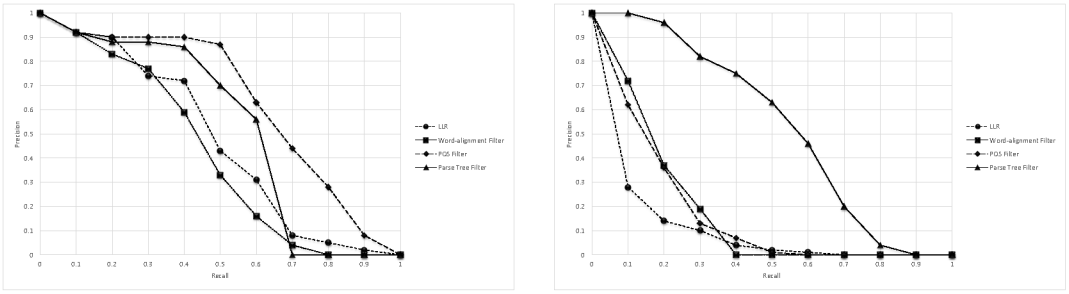
After analyzing the list of DCs generated by all approaches, we noted that the size of a DC affects the performance of our approach. Figure 3 shows the performance of each filter in detecting unigram (Figure 3a) and bigram (Figure 3b) DCs. These figures shows that except for the Syntax Tree filter, the performance of the identification of bigram DCs drops rapidly when compared with the identification of unigram DCs. To better understand why longer DCs are more difficult to identify, we manually analyzed the errors of each filters. The most significant proportion of errors with bigram DCs is generated from a unigram DC and a noisy word. For example, "*mais je*" is composed of the French DC "*mais*" and a noisy word "*je*". As these errors usually do not create a syntactic well-defined constituent, they can only be filtered out by the Syntax Tree Filter.

The POS pattern filter cannot detect noisy syntactic components since detecting such components needs contextual syntactic information. When we analyzed negative examples of this filter, we noticed that most of bigram errors are comprised of two words that belong to two different chunks. For example, in sentence (2) below, the POS pattern "ADV C" extracts "*donc que*", but these two words belong to two different syntactic constituents (i.e *ADV* and *Ssub*).

(2) VN [Je demande] ADV [donc] Ssub[que l'on soutienne l'Irlande dans ce cas particulier].

It is interesting to note that the ranked list created with the Syntax Tree Filter includes several DCs that do not appear in the LEXCONN lexicon but are nevertheless correct DCs in French. Among the top 100 candidates labeled as an incorrect DC, we have found 31 correct DCs which are not listed in LEXCONN, such as"*toutefois*", "*certes*" and "*au lieu de cela*". The work of (Roze et al., 2012) (or any manually curated list of DCs) constitutes an invaluable resource. However, as Prasad et al. (2010) mentioned, DCs are open-class terms. Therefore, our approach to induce DCs from parallel texts can be

[6]When calculating recall points, we only considered the available DCs in the dataset after applying the Frequency Filter (i.e. 66% of DCs).



(a) Unigram DCs (b) Bigram DCs

Figure 3: 11-Point Interpolated Average Precision Curve for the Extraction of Unigram and Bigram DCs

used to improve the coverage of such a list.

The results of the Word-alignment show that the intersect word-alignment model cannot align DCs in English to French. Indeed, our analysis shows that only 176 LEXCONN DCs (38%) were aligned to English DCs. We believe that since a discourse relation can be conveyed with different DCs and human translators can choose between them during the translation, aligning DCs is much harder for alignment models. Moreover, DCs can be also placed at the beginning or at the end of discourse units, therefore the word-alignment needs to tolerate long-distance alignment to align them.

Finally, since LEXCONN uses a different set of relations than PDTB's relations, we cannot evaluate relations that are assigned to French DCs in the ranked list of candidates. However, a preliminary analysis of 20 randomly selected pairs of <DC, relation>among the top 100 pairs in the ranked list of the Syntax Tree Filter showed that it achieves 75% precision in labeling DCs.

5 Conclusion and Future Work

In this paper, we have presented an approach to induce discourse connectives from a parallel text. In our approach, we have extracted a list of DC candidates and ranked them with the Log-Likelihood Ratio. We have also used several filters to prune the final list of DCs: Word-Alignment, POS Patterns and Syntax Tree Filter. We have achieved the best result in term of MAP with the Syntax Tree Filter. Our analysis shows that the size of discourse connectives affects the quality of the filters. We also found that some candidates that labeled as wrong discourse connective, are indeed correct discourse connectives, yet are not covered in the LEXCONN lexicon.

There are several ways in which this work can be extended. Firstly, although we used the French language to do our experiment, the same methodology can be applied to other languages. It is worth mentioning that French has pervasive multi-word expressions and our approach suffers from such components since they are usually long expressions. We believe that our approach would achieve a better result in languages with shorter discourse connectives. However, for languages that mark discourse relations through other means such as morphology (e.g. Arabic), the approach would certainly have to be reviewed. Secondly, we have produced a huge number of sentences and automatically tagged them with their discourse relation, however, the discourse relations have not been evaluated. As a future work, we would like to evaluate the discourse relations in these sentences and use the same corpus to implement a French discourse connective classifier that labels discourse connectives with their relations.

Acknowledgement

The authors would like to thank the anonymous referees for their insightful comments on an earlier version of the paper. Many thanks also to Laurence Danlos and Félix-Hervé Bachand for fruitful discussions on this work. This work was financially supported by an NSERC grant.

References

Amal Al-Saif and Katja Markert. 2010. The Leeds Arabic Discourse Treebank: Annotating Discourse Connectives for Arabic. In *LREC*, pages 2046–2053, Valletta, Malta.

Laura Alonso Alemany, Irene Castellón Masalles, and Lluìs Padró Cirera. 2002. Lexicón computacional de marcadores del discurso. *Procesamiento del Lenguaje Natural*, 29.

Bruno Cartoni. 2013. Annotating the meaning of discourse connectives by looking at their translation: The translation-spotting technique. *Dialogue & Discourse*, 4(2):65–86.

Stefan Evert. 2004. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. PhD dissertation, Institut fr Maschinelle Sprachverarbeitung, University of Stuttgart.

Qin Gao and Stephan Vogel. 2008. Parallel Implementations of Word Alignment Tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57, Columbus, OH, USA.

Spence Green, Marie-Catherine de Marneffe, John Bauer, and Christopher D. Manning. 2011. Multiword Expression Identification with Tree Substitution Grammars: A Parsing tour de force with French. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 725–735, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Adam Kilgarriff, Vojtch Kov, Simon Krek, Irena Srdanovi, and Carole Tiberius. 2010. A Quantitative Evaluation of Word Sketches. In *Proceedings of the 14th EURALEX International Congress*, Leeuwarden, The Netherlands.

Alistair Knott. 1996. *A data-driven methodology for motivating a set of coherence relations*. PhD dissertation, University of Edinburgh.

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

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2010. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, 20(02):151–184.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*, volume 1. Cambridge University Press.

Thomas Meyer and Bonnie Webber. 2013. Implicitation of Discourse Connectives in (machine) Translation. In *Proceedings of the 1st DiscoMT Workshop at ACL 2013 (51st Annual Meeting of the Association for Computational Linguistics)*, pages 19–26, Sofia, Bulgaria.

Thomas Meyer, Charlotte Roze, Bruno Cartoni, L. Danlos, and A. Popescu-Belis. 2011. Disambiguating discourse connectives using parallel corpora: senses vs. translations. In *Proceedings of Corpus Linguistics*.

Thomas Meyer. 2011. Disambiguating Temporal-Contrastive Discourse Connectives for Machine Translation. In *Proceedings of ACL-HLT*, pages 46–51, Portland, OR, USA.

Lucie Mladová, Sarka Zikanova, and Eva Hajicová. 2008. From Sentence to Discourse: Building an Annotation Scheme for Discourse Based on Prague Dependency Treebank. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, pages 28–30, Morocco, Marrakech.

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

Umangi Oza, Rashmi Prasad, Sudheer Kolachina, Dipti Misra Sharma, and Aravind Joshi. 2009. The Hindi Discourse Relation Bank. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 158–161, Suntec, Singapore.

P. Pecina. 2010. Lexical association measures and collocation extraction. *Language Resources and Evaluation*, 44(1):137–158.

T. Pedersen, S. Banerjee, B. T. McInnes, S. Kohli, M. Joshi, and Y. Liu. 2011. The Ngram Statistics Package (text:: NSP)-A Flexible Tool for Identifying Ngrams, Collocations, and Word Associations. In *Workshop on Multiword Expression: from Parsing and Generation to the Real World (MWE 2011)*, pages 131–133, Portland, OR, USA.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers , pages*, pages 13–16, Suntec, Singapore.

Andrei Popescu-Belis, Thomas Meyer, Jeevanthi Liyanapathirana, Bruno Cartoni, and Sandrine Zufferey. 2012. Discourse-level Annotation over Europarl for Machine Translation: Connectives and Pronouns. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 23–25, Istanbul, Turkey.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K. Joshi, and Bonnie L. Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, pages 28–30, Marrakech, Morocco.

Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2010. Realization of Discourse Relations by Other Means: Alternative Lexicalizations. In *COLING '10*, pages 1023–1031, Beijing, China.

Charlotte Roze, Laurence Danlos, and Philippe Muller. 2012. LEXCONN: a French lexicon of discourse connectives. *Discours [En ligne]*, (10).

V. Seretan. 2010. *Syntax-Based Collocation Extraction*, volume 44. Springer-Verlag.

Manfred Stede and Carla Umbach. 1998. DiMLex: A lexicon of discourse markers for text generation and understanding. In *Proceeding of the 17th international conference on Computational Linguistics (COLING-98)*, pages 1238–1242, Montreal, Canada. Association for Computational Linguistics.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003*, pages 173–180, Edmonton. Association for Computational Linguistics.

Yannick Versley. 2010. Discovery of ambiguous and unambiguous discourse connectives via annotation projection. In *Proceedings of Workshop on Annotation and Exploitation of Parallel Corpora (AEPC)*, pages 83–82, Tartu, Estonia. Northern European Association for Language Technology (NEALT).

Yannick Versley. 2011. Towards Finer-grained Tagging of Discourse Connectives. In *Beyond Semantics: Corpus-based investigations of pragmatic and discourse phenomena*.

Deniz Zeyrek, In Demirahin, Ay Sevdik-all, Hale gel Balaban, hsan Yalnkaya, and mit Deniz Turan. 2010. The annotation scheme of the Turkish Discourse Bank and an evaluation of inconsistent annotations. In *Proceedings of the Fourth Linguistic Annotation Workshop*, pages 282–289, Uppsala, Sweden. Association for Computational Linguistics.

Yuping Zhou and Nianwen Xue. 2012. PDTB-style Discourse Annotation of Chinese Text. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 69–77, Jeju, Republic of Korea. Association for Computational Linguistics.

Lanjun Zhou, Wei Gao, Binyang Li, Zhongyu Wei, and Kam-Fai Wong. 2012. Cross-lingual identification of ambiguous discourse connectives for resource-poor language. In *Proceedings of COLING 2012*.

Lyrics-based Analysis and Classification of Music

Michael Fell
Computational Linguistics
Saarland University
D-66123 Saarbrücken
mic.fell@gmail.com

Caroline Sporleder
Computational Linguistics & Digital Humanities
Trier University
D-54286 Trier
sporledc@uni-trier.de

Abstract

We present a novel approach for analysing and classifying lyrics, experimenting both with n-gram models and more sophisticated features that model different dimensions of a song text, such as *vocabulary*, *style*, *semantics*, *orientation towards the world*, and *song structure*. We show that these can be combined with n-gram features to obtain performance gains on three different classification tasks: genre detection, distinguishing the best and the worst songs, and determining the approximate publication time of a song.

1 Introduction

The ever growing amount of music available on the internet calls for intelligent tools for browsing and searching music databases. Music recommendation and retrieval systems can aid users in finding music that is relevant to them. This typically requires automatic music analysis, e.g., classification according to genre, content or artist and song similarity. In addition, automatic music (and lyrics) analysis also offers potential benefits for musicology research, for instance, in the field of Sociomusicology where lyrics analysis is used to place a piece of music in its sociocultural context (Frith, 1988).

In principle, both the audio signal and the lyrics (if any exist) can be used to analyse a music piece (as well as additional data such as album reviews (Baumann et al., 2004)). In this paper, we focus on the contribution of the lyrics. Songwriters deploy unique stylistic devices to build their lyrics. Some of those can be measured automatically and we hypothesise that these are distinctive enough to identify song classes such as genre, song quality and publication time. There is, in fact, strong empirical evidence that it is worthwhile to look deeper into lyrical properties when analysing and classifying music. For example, it has been shown that classifiers that incorporate textual features outperform audio-only classifiers on most classification tasks (Mayer et al., 2008a; Mayer and Rauber, 2011; Li and Ogihara, 2004). Lyrics are also often easier to obtain and process than audio data, and non-musicians, in particular, often rely strongly on lyrics when interacting with a music retrieval system (Baumann and Klüter, 2002; Bainbridge et al., 2003). Moreover, lyrics do not only add semantic content, they can serve as an (easily observable) proxy for the melodic, structural and rhythmic properties of the audio signal. Melody and rhythm, for example, can often be traced in the stress pattern of the text (Nichols et al., 2009), while a song's overall structure is reflected in the order of textual elements such as chorus, verse and bridge. Psychological research also provides evidence the audio and textual content are indeed processed independently in the brain and hence are complementary for our appreciation of a song (Besson et al., 1998).

We extend previous research on lyrics-based song classification in two important ways: First, while earlier approaches mostly used fairly shallow textual features, such as bags-of-words, we designed features that model semantic and stylistic properties of lyrics at a much deeper level and show that these features can indeed be beneficial. Second, we address two novel classification tasks beyond genre detection, namely distinguishing 'best' and 'worst' songs[1] and determining the approximate publication time, and show that these can also be tackled by lyrics analysis.

[1]There is a growing body of work on automatic hit prediction but we would argue that this is a different task as hits are not necessarily qualitatively good songs and vice versa.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 620–631, Dublin, Ireland, August 23-29 2014.

2 Related Work

This study draws on earlier work on text classification, including genre detection (Lustrek, 2007) and authorship attribution (Stamatatos, 2009; Holmes, 1994), but also more specifically on poetry analysis (Simonton, 1990) and, in particular, lyrics-based music classification. Generally, shallow features, such as average word and sentence length, part-of-speech and function word distribution tend to work well for authorship and genre classification, while content word distribution is more indicative of the topic. Recent work on text classification has also employed deeper features, such as distributions of syntactic constructions (see e.g., Kim et al. (2010)). However, not all features that work well for prose carry over to song lyrics. Syntax, for example, is strongly constrained by meter. On the other hand, additional features like meter and rhyme properties might be useful. So far, most studies on lyrics classification have used rather simple features, for example (tf-idf weighted) bags-of-words (Neumayer and Rauber, 2007; Mahedero et al., 2005; Logan et al., 2004), sometimes enriched by synonymy and hypernymy information (Scott and Matwin, 1998). Mayer et al. (2008a; 2008b) also include POS tag distributions, simple text statistics (avg. word length, proportion of hapax legomena per document/line, distribution of punctuation marks and digits, words per minute) and simple (end-of-line) rhyme features. Li and Ogihara (2004) use a similar feature set but also include function word distributions. Finally, Hirjee and Brown (2010) analyse Rap lyrics and focus exclusively on rhyme features, providing a sophisticated statistical rhyme detector which can also identify in-line and slant rhymes. We build on this work but extend the feature space with more explicit modelling of abstract stylistic and linguistic dimensions such as vocabulary, style, semantics, orientation of the song content with respect to the world and overall song structure.

3 Material

Since no large lyrics dataset was publicly available (cf. Mayer and Rauber (2011)), we had to collect our own.[2] Song lyrics are widely available across the internet in the form of user-generated content. We chose *Lyricsmode*[3] because of its large coverage and subjectively high consistency. Even so, a certain amount of inconsistency and noise remains. We employed heuristics to clean the data, e.g., to remove duplicate song texts and normalise the notation style of different users.[4] Only English lyrics were included; songs in other languages were filtered out using language detection.[5] Furthermore, to minimise data sparseness, songs were only included if more than 20 song texts were available for the corresponding artist. The final corpus consists of roughly 400k English song texts of 7.2k artists.[6] For the experiments, the lyrics were POS tagged[7] and chunked.[8]

In addition to the lyrics themselves, we need three types of metadata for our experiments: genre information, quality ratings, and publication time. In all experiments, we classify songs rather than artists or albums. However, to avoid artist effects on our results, we control for the artist, i.e., we make sure that the test set does not contain (songs of) an artist if the training set already contains (songs of) the same artist; test and training set are completely disjunct with respect to artists.[9] Because of this, we need to ensure a sufficient number of artists for each output class in the three experiments.

[2]The Million Song Database (Bertin-Mahieux et al., 2011), a large publicly available data set for music classification, does not contain lyrics and the only available data set that does contain lyrics, SLAC (McKay et al., 2010), only contains lyrics for 160 songs, which is too small to train and test on.

[3]`http://www.lyricsmode.com`

[4]See Fell (2014) for more details on the heuristics used in the present study and Knees et al. (2005) for an overview of the types of noise typically encountered and general methods for cleaning.

[5]A freely available Java library for language detection (Shuyo, 2010) was used.

[6]Note that there is no guarantee that the artist also wrote the lyrics. The corpus might contain covers and lyrics/songs that were written 'on request'. However, performers do not choose their songs randomly but try to stick to songs that fit in with their preferred genre and style.

[7]`http://nlp.stanford.edu/software/tagger.shtml`

[8]`https://opennlp.apache.org`

[9]Most previous studies did not explicitly control for this. However, we noticed in a preliminary experiment that the results can be notably inflated if training and test set overlap in artists. For genre classification, we saw an increase in F-Score of up to 7%, while for publication time classification, the F-Score increased by up to 12% (Fell, 2014). This indicates that lyrics may provide a stronger signal for the artist than for other classes such as genre or publication time.

Genre information was obtained from *Allmusic*,[10] which classifies artists and bands according to 21 coarse-grained genres and numerous subgenres. We excluded artists who experimented with several genres like Peter Gabriel (Pop/Rock, International) or Prince (R&B, Pop/Rock, Electronic) because in that case it is not clear which genre a particular song belongs to. As most genres occur only sparsely in our corpus, we focused on the nine most common genres,[11] resulting in a data set of 4,712 artists from the nine major genres with the following numbers of artists per genre: Pop/Rock: 2602, Metal: 1140, Rap: 390, Country: 225, R&B: 153, Religious: 118, Reggae: 38, Blues: 26, Folk: 20.

Besides genre, we retrieved album ratings and publication years from *Rateyourmusic*.[12] Album ratings range from 0 stars (worst) to 5 stars (best) and are typically averaged over hundreds to thousands user ratings. To exclude "one-hit-wonders", only artists with at least two rated albums were considered. Theoretically, it would be possible to assign all songs a rating by transferring an album rating to all songs in the album. However, in practice this is difficult to do robustly because album ratings and lyrics come from different sites and are not trivial to align. Song listings for an album are sometimes incomplete and song titles noisy, making it difficult to map album ratings directly to songs. As a way around this we map album ratings to artists (which are much more robustly identifiable from the metadata) and then compute an overall artist rating as the median over all album ratings for the artist. Each song by the artist is then assigned this rating. Basically, we hypothesise that a good artist consistently writes good songs, which is, obviously, a simplifying assumption.

4 Features

We designed 13 feature classes, consisting of one or more related features each, and grouped them into five abstract sets, reflecting different stylistic and linguistic dimensions (see Table 1).[13]

Model	Dimension	Feature Classes
topK	vocabulary:	output class specific top 100 n-grams ($n \leq 3$)
extended	vocabulary:	type-token ratio, non-standard words
	style:	POS/chunk tags, length, echoisms, rhyme features
	semantics:	imagery
	orientation:	pronouns, past tense
	song structure:	chorus, title, repetitive structures

Table 1: Overview of features

As a baseline (*topK*), we implemented an n-gram model, which captures words and collocations that are most specific to an output class. This model can be considered 'uninformed' in that it does not attempt to represent abstract stylistic or structural properties. We rank n-grams according to the tf-idf for the class (i.e., the genres are considered 'documents' and the frequency of an n-gram is incremented by 1 for each song in which it occurs). To reduce the impact of vocabulary preferences of individual artists, we then re-rank by discounting n-grams which are too artist-specific. The top 100 n-grams (for $n \leq 3$) are represented in the feature vector.[14]

The remaining features (*extended*) attempt to model the following five dimensions of the lyrics:

VOCABULARY: These features estimate the vocabulary richness (**type-token ratio** for n-grams up to $n = 3$) and the use of **non-standard words**, i.e., uncommon and slang words. Uncommon words are defined as words not found in Wiktionary.[15] Slang words are defined as words contained in the Urban Dictionary,[16] but not in Wiktionary. We encode the (normalised) logarithmic frequency of slang words and the ratio of uncommon words to all words.

[10] `http://www.allmusic.com`

[11] We excluded the, also fairly frequent, genre Electronic as it is mainly musically defined (Logan et al., 2004).

[12] `http://www.rateyourmusic.com`

[13] Note that features are normalised by the length of the lyrics where necessary.

[14] The total number of encoded n-gram features is maximally 300 per output class but can be less, since n-grams common to multiple classes are encoded only once.

[15] `http://en.wiktionary.org`

[16] `http://www.urbandictionary.com`

STYLE: We employed the **POS and chunk tag** distributions as proxies for syntactic structure. To reduce data sparseness, all tags are mapped to supertags such as V, N, ADV. We also implemented various **length** features (lines per song, tokens per song, tokens per line). **Rhyme** structure is modelled by encoding the output of the rhyme detection tool by Hirjee and Brown (2010), which detects perfect and imperfect in-line and line final rhymes. Repetitions of letters ("riiiiise") or words ("money, money") are common in lyrics and often caused by a mismatch between number of syllables and line meter but they can also be employed as a means for emphasis and indicating emotion. We collectively dub such repetitions **echoisms**. We also group in-line (slant) rhymes ("burning turning", "where were we") under 'echoisms'. Echoisms are computed by looking for words with letter repetitions or word sequences with a relatively high similarity (according to an edit distance measure). Frequencies per type (letter reduplication, word repetition) and sequence length (less or more than 3 words) are encoded.

SEMANTICS: Lyrics can vary widely with respect to the topics they mention and the images they evoke. Instead of using a linguistic model of semantic fields, we opted to build on work in psychology and use the Regressive Imagery Dictionary (RID) (Martindale, 1975; Martindale, 1990) to identify predominant concepts ("imageries") in a text. RID classifies words as belonging to the separate fields "conceptual thought" (abstract, logical, reality-oriented), "primordial thought" (associative, concrete, fantasy), and "emotion". For example, the imagery 'Moral' (conceptual) contains words such as "should", "right", and "virtue". Whereas the imagery 'sensation' (primordial) contains "delicious", "perceive", and "glamour". We chose this resource because, intuitively, it is not only important *what* is said but also *how* it is said and the RID seemed to capture both aspects well. We computed the dominant imageries for each text and encoded this information in the feature vector.

ORIENTATION: This dimension models how the song narrative (entities, events) is oriented with respect to the world. We encode a temporal dimension, i.e., whether the song mainly recounts past experiences or present/future ones, by representing the fraction of **past tense verb forms** to all verb forms as a feature. We also model how "egocentric" a song is. We compute **pronoun** frequencies for 1st, 2nd, 3rd singular and plural person. As derived features, we also encode the proportion of self-referencing pronouns (first person singular/plural) to non-self-referencing ones and the ratio of first person singular pronouns to second person. The former feature measures the degree of talking about oneself as opposed to talking about other people, the latter measures whether the "I" or the "you" carries more weight in an interpersonal relationship.

SONG STRUCTURE: Structural repetitions are characteristic of song texts. We search for **repetitive structures**, i.e., identical or similar multi-line blocks that re-occur, typically but not always representing the chorus. We use heuristics to align such structures, allowing for fuzzy matches. An example of a song text[17] with a repeated structure is provided in Figure 1, where lines 56-60 are aligned to lines 61-65. It can be seen that corresponding lines are not lexically identical but only structurally and lexically similar. To be able to recognise such cases, we compute the overall similarity between two lines as a weighted sum of their lexical and structural similarities which are modelled in terms of word and POS tag bigram overlaps, respectively. Using this information and a set of heuristics, it is then determined whether a song contains a **chorus** and whether the **title** appears in the song text.

[56] 'Cause now I see right through you	**[61]** But I see right through you
[57] Look into my eyes	**[62]** I look into your eyes
[58] Tell me what you see	**[63]** Tell you what I see
[59] I see a man who thought you loved me	**[64]** I see a girl who ran game on me
[60] You played me like a fool	**[65]** You thought you had me fooled

Figure 1: Alignment of two blocks in the same song text

[17] *See right through you* by 'NSync.

	genre	best vs. worst			approx. publication time
training set	2,520	1,008 (Rap)	1,680 (Metal)	3,360 (Pop/Rock)	315 (Pop/Rock)
test set	840	294 (Rap)	546 (Metal)	1,092 (Pop/Rock)	105 (Pop/Rock)

Table 2: Average data set sizes (number of songs) for each experiment

5 Experiments

We carried out three experiments: classifying songs by (i) their genre, (ii) their quality (best vs. worst), and (iii) their approximate publication time. There is empirical evidence that lyrics may indeed play a crucial role in all three classification tasks. Musical genre is often defined as a cultural category, rather than a purely musical one (Fabbri, 1981). What topics artists sing about and how they sing about them clearly belongs to this cultural dimension. Lyrics also contribute to whether a song is viewed as 'good' or 'bad'. A study by Cunningham et al. (2005) also indicates that the lyrics are an important factor for *disliking* a song and Salley (2011) provides examples of how (text-)stylistic devices such as alliteration can make a song more engaging and therefore more successful. Finally, Hirjee and Brown (2010) show, for Rap, that the dominant style of song texts can change over time.

In all experiments, we compared our baseline feature set (*topK*) against the extended set (*extended*) and a combined set (*combined*). As the class distribution in our data is severely skewed, we performed random undersampling to create balanced sets for all experiments and thus avoid problems commonly associated with learning from imbalanced data (He and Garcia, 2009). The sampled data sets were split into 75% for training and 25% for testing. The exact numbers depend on the experiment (see Table 2). We repeated the sampling, training and testing procedure between 100 and 1000 times (depending on the experiment) and report the average. The Weka (Hall et al., 2009) implementation of SVMs with the default setting was used for classification.

5.1 Experiment 1: Genre Classification

We focused on the following eight genres: Blues, Rap, Metal, Folk, R&B, Reggae, Country, and Religious. Pop/Rock was excluded because it is the most heterogeneous genre and comprises many subgenres. Table 3 shows the results per genre and averaged over all genres, as well as the standard deviations. The n-gram model (*topK*) outperforms the *extended* model on all genres except Country but a combination of both models consistently yields even better results with an overall average F-Score of 52.5%. All F-Score differences between the three models are statistically significant at $p < 0.01$.[18] The fact that the combined model performs best indicates that the two basic models are at least partially complementary. The n-gram model hones in on the topic of a text, while the extended model captures more abstract structural and stylistic properties. However, both perform similarly on individual genres, i.e., they both in themselves capture important aspects of 'genre'. Looking at the individual genres, Rap seems to be most easily detectable on the basis of the lyrics alone (77.6% F-Score, *combined*). This is not surprising, since Rap lyrics have properties that are quite unique, such as complex rhyme structures, long lyrics and a fairly distinctive vocabulary. Folk seems to be the most difficult genre (29.6% F-Score, *combined*). A look at the confusion matrix revealed that Folk was frequently confused with Blues or Country. All share similar topics (e.g., love, traveling) and are also structurally and stylistically similar. They are mainly distinguished by musical properties (instrumentation, rhythm etc.). Lyrical similarities and differences are also revealed by looking at the top 100 unigrams for each genre (Figures 3 to 10). It can be seen that some genres stand out lexically, for example Rap (dominant slang use), Reggae (Jamaican slang, Rastafarian terms), Religious (religious terms) and Metal (death, violence). Some genres, however, are lexically quite similar such as Folk, Blues and Country.

Figure 2 shows the contributions of different feature groups to the overall performance of the combined model.[19] It can be seen that *length* contributes most, followed by *slang use*, *type-toke ratio*, *POS/Chunk*

[18] We performed a non-exhaustive permutation test by sampling 10^7 permutations and computed the Wilson-Score Interval for the estimated p-value with probability 99.9999%.

[19] The feature contribution is measured by correlating the features with the output class labels by computing the Symmetric

F-score [%]	Blues	Rap	Metal	Folk	R&B	Reggae	Country	Religious	Average
topK	51.2	76.0	49.0	28.3	48.7	44.4	41.3	53.3	49.0 ($\pm$ 2.5)
extended	46.6	75.1	47.3	24.5	47.7	35.8	53.8	37.3	46.0 ($\pm$ 2.4)
combined	**54.1**	**77.6**	**52.0**	**29.6**	**52.6**	**45.4**	**54.6**	**53.8**	**52.5** ($\pm$ 2.7)
human optimistic	40.9	66.7	42.4	18.2	34.8	12.1	28.8	53.0	37.1 ($\pm$ 8.4)
human pessimistic	37.6	53.8	38.3	18.6	29.4	15.3	27.7	47.8	33.6 ($\pm$ 7.5)

Table 3: F-Scores[%] for genre classification (1000 runs, averages)

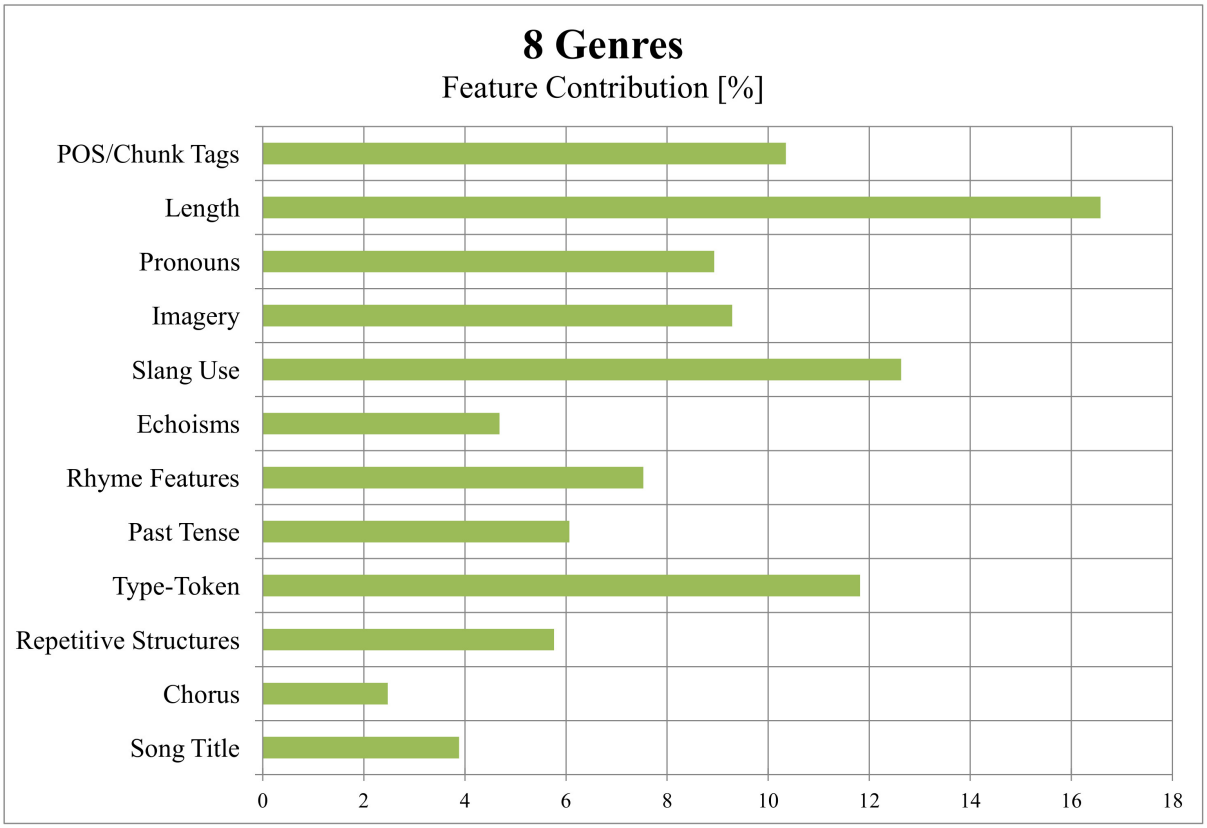


Figure 2: Feature Contributions for Experiment 1 (combined model)

tags and the more semantic features *imagery* and *pronouns*. Rap, which tends to have long lyrics with many slang words, is the genre that is identified most reliably by the classifiers and it is therefore not surprising that the two most contributing feature groups are particularly well suited for distinguishing Rap from the remaining genres.

While the performance of the combined model is promising,[20] there is still room for improvement. In order to determine whether this is a limitation inherent to the model or whether lyrics alone simply do not provide a strong enough signal for music genre classification, we performed a human annotation experiment. Participants ($n = 11$) had to classify randomly selected song texts into the 8 genres. They were allowed to assign up to two genres to each song. We report two performance measures (see Table 3): *human optimistic* counts an instance as correct if the correct genre was in the set of genres assigned, *human pessimistic* only counts unique genre assignments which correspond to the gold standard as correct. It can be seen that the human performance is actually worse than the automatic classification.[21]

Uncertainty (SU) (Witten and Frank, 2005) for each feature and class label. By accumulating the SUs for all features in a feature group we estimate to which proportion on average a group of features helps in identifying the correct class.

[20]The random baseline for this experiment is 12.5% F-Score.

[21]While this is unusual, the same observation has been made for some other stylometric tasks, in particular translation detection (Baroni and Bernardini, 2006).

Apparently, humans had difficulty picking up on subtle stylistic properties, especially since they were not 'trained' in any way, i.e., they had to rely on their own conception of what is typical for a genre. Hence, they (self-reportedly) relied mostly on the topic of a text.[22] Comparing results on individual genres, however, humans behave similar to the automatic classifier: Folk is the most difficult genre, Rap the easiest. An exception is Reggae which was more difficult for our participants than for the models. We performed a more detailed (statistical) comparison of the confusion matrices for humans and classifiers, which indicated that genres are indeed similarly confused by both. This could suggest that some genres are inherently more difficult to detect than others (based on lyrics alone).

Figure 3: Blues top 100 words

Figure 4: Rap top 100 words

Figure 5: Metal top 100 words

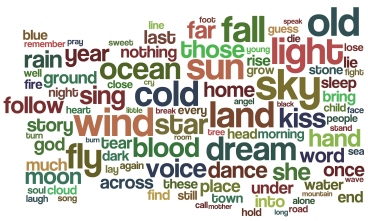


Figure 6: Folk top 100 words

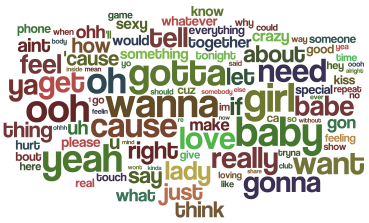


Figure 7: R&B top 100 words

Figure 8: Reggae top 100 words

Figure 9: Country top 100 words

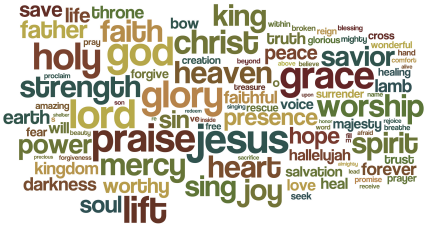


Figure 10: Religious top 100 words

5.2 Experiment 2: Best vs. Worst Music

In our second experiment, we tested whether the 'best' songs can be distinguished from the 'worst' solely on the basis of their lyrics. Having obtained average artist ratings (see Section 3), we defined the best (worst) artists as top (bottom) percentiles of all ratings. We also made sure that the distance between best and worst ratings was at least 1 point to ensure there was still a large enough gap. We assume that the quality of a song is genre-dependent, i.e., properties that make a good rap song are not necessarily desirable for a good blues song. Hence, our classifiers were trained and tested within genres. Only three of the original genres had enough material to satisfy the constraints: Pop/Rock, Metal, and Rap. For Pop/Rock and Metal, where more material was available, the 'best' ('worst') was defined as the top (bottom) 5% of artists, while for Rap the top (bottom) 10% percentiles were considered.

[22] The standard deviation is quite high for humans. This may be due to the relatively small number of participants or due to the fact that some participants had more previous exposure to different genres.

Model	Pop/Rock			Metal			Rap		
	Best	Worst	Average	Best	Worst	Average	Best	Worst	Average
topK	69.4	72.0	70.8 ($\pm$ 2.2)	71.3	72.0	71.7 ($\pm$ 3.7)	85.9	85.1	85.6 ($\pm$ 4.1)
extended	72.6	73.9	73.3 ($\pm$ 2.0)	76.5	76.2	76.4 ($\pm$ 4.0)	81.1	82.4	81.8 ($\pm$ 4.2)
combined	**74.7**	**76.2**	**75.5** ($\pm$ 2.3)	**76.7**	**76.2**	**76.5** ($\pm$ 4.3)	**86.4**	**86.3**	**86.4** ($\pm$ 4.1)

Table 4: F-Score[%] for Best vs. Worst (100 runs, averages)

Table 4 shows the results, which are encouragingly high, ranging from 75.5% to 86.4% F-Score (compared to a random baseline of 50% F-Score). It seems that the quality of a song does indeed at least partially depend on the quality of its lyrics and that the latter can to some extend be determined automatically. As in the previous experiment the *combined* model outperforms the other two models. However, unlike in the previous experiment, the *extended* model now outperforms the *topK* on two genres (Pop/Rock and Metal). This suggests that, at least for these two genres, the simple word n-grams are not sufficient to distinguish good and not so good songs; other features, contribute as well. Rap is the odd-one-out here: For this genre, the quality of a song seems to lie largely in the words and phrases used. All differences in F-Scores between the three models are significant with $p < 0.01$, except for the difference *extended* vs. *combined* for Metal, which is not significant ($p > 0.3$).

Figure 11 shows the feature contributions in the combined model. It can be seen that Rap behaves differently than the other two genres. For Rap the features *length* and *slang* contribute most, followed by *type-token ratio*, *POS/Chunk tags*, *pronouns* and *rhyme features*. The latter are noticeably more important for Rap than for the other two genres. For Pop/Rock and Metal, *type-token ratio* is by far the most important, closely followed by *length*. Orientation features (*pronouns*, *past tense*), song structure (*repetitive structures*, *chorus*) and *POS/Chunk tags* also contribute quite a lot. Generally, it seems to hold for all three genres that the best songs are characterised by a higher type-token ratio, fewer interjections and nonsense words ("lalala"), and lower ratio of first person pronouns.

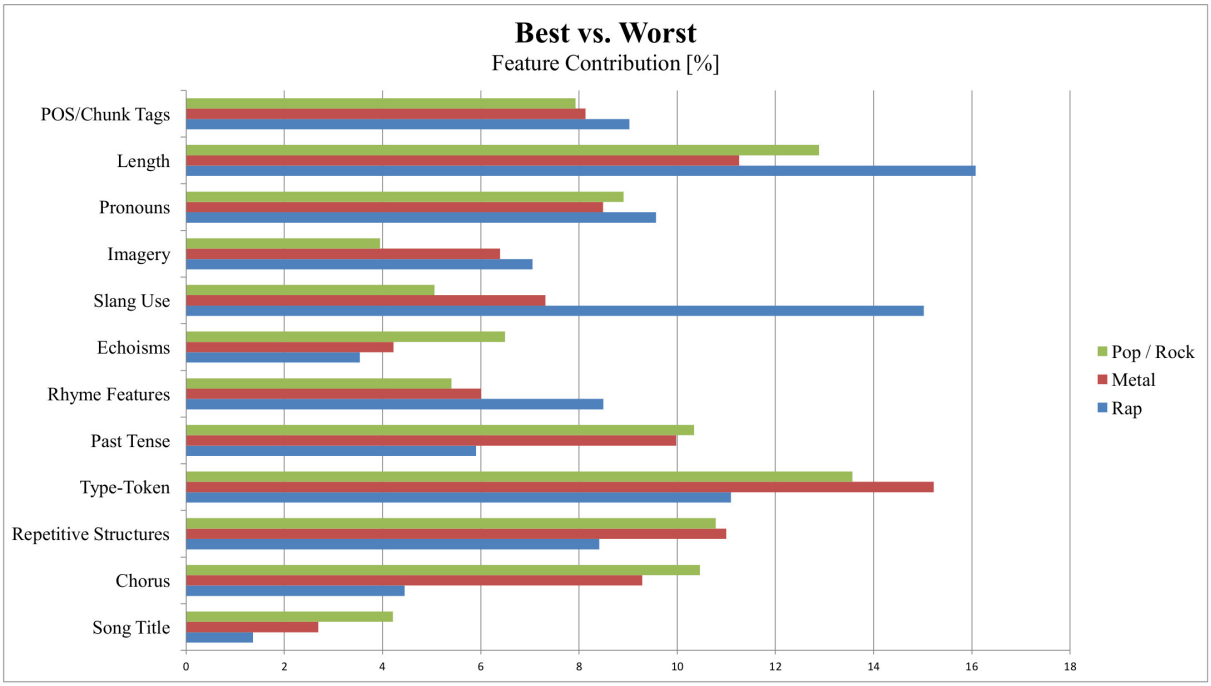


Figure 11: Feature Contributions for Experiment 2 (combined model)

We also look at n-grams distinguishing good from bad songs. Generally, it can be said that the best songs are much less concerned with sex and violence and more with story-telling. For example, the best Rap songs deal with the cosmic battle of man, good vs. evil, and rapping - while the worst Rap seems to be more about sex, violence, and money (see Figures 12 and 13).

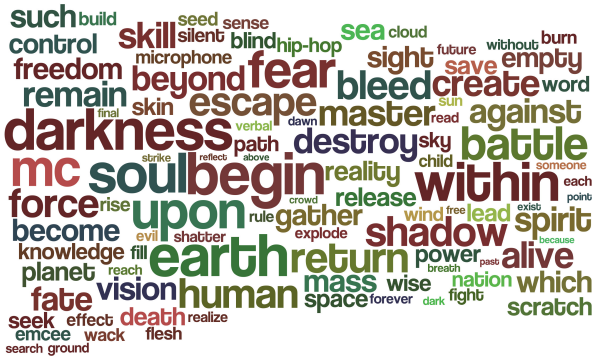


Figure 12: Best Rap top 100 words

Figure 13: Worst Rap top 100 words

5.3 Experiment 3: Approximate Publication Time

In the final experiment, we investigate whether one can automatically predict the approximate publication time of a song given its lyrics. We controlled both for genre and musical quality and trained classifiers to separate the song texts according to their age. We chose Pop/Rock since we have the most material for this genre and defined three time groups in such a way that we have sufficient data for each of them: *New* contains songs published since 2008, *mid-age* songs published between 1998 and 2001, and *old* songs published before 1988. We intentionally left gaps between the three classes since the boundaries are somewhat arbitrary and we assume that a song published in, say, 2008 will not *per se* differ substantially from one published in 2007.

Table 5 shows the results. Regarding the performance of the different models, we see a similar trend as in the previous experiments: the combined model performs best, the n-gram model performs slightly better than the extended model but there is not much between them. The differences in F-Scores between the models are statistically significant with $p < 0.01$. Overall, while all models beat the random baseline (33% F-Score), the results are still relatively low. By inspecting the confusion tables, we discovered that *old* and *new* are separated well from each other, with *new* being classified as *old* in only 23% of the cases and the opposite happening in 17% of the cases. On the other hand, *mid-age* shows the lowest F-Score and is misclassified in an almost symmetrical way (*new*: 27%, *mid-age*: 41%, *old*: 32%). It seems that, while the publication date of a song can at least partially be determined on the basis of the lyrics, only songs which are published 20 years and more apart can be distinguished relatively well. In a way this is not so surprising; we would expect the musical style to change over shorter times spans reflecting for example changes in taste regarding instrumentation and recording technology (live drums vs. drum computers, use of auto-tune effects etc.), properties of the lyrics, however, tend to change over much longer time periods.

Model	New	Mid-Age	Old	Avg.
topK	47.2	39.2	48.7	45.0 ($\pm$ 4.4)
extended	44.1	39.1	43.2	42.2 ($\pm$ 4.9)
combined	48.9	41 .0	50.5	46.8 ($\pm$ 5.4)

Table 5: F-Scores [%] for Old vs. Mid-Age vs. New Pop/Rock (100 runs, averages)

Nonetheless, our models learnt some interesting properties that distinguish old and new songs. Predictably, it was found that newer song texts tend to be longer. This is in line with the fact that the duration of songs increased over time. Older songs also contain more repetitive structures, more exact matches between text blocks and have a higher chance of containing a chorus. With respect to overall song structure it thus seems that the songwriters deviated more and more from traditional structure templates over time. Interestingly (and more surprisingly), newer songs also contain less possessive pronouns and more terms referring to food and drink. Figure 14 shows the feature contributions, confirming the importance of length and repetitive structures. Figures 15 to 17 show the word clouds.

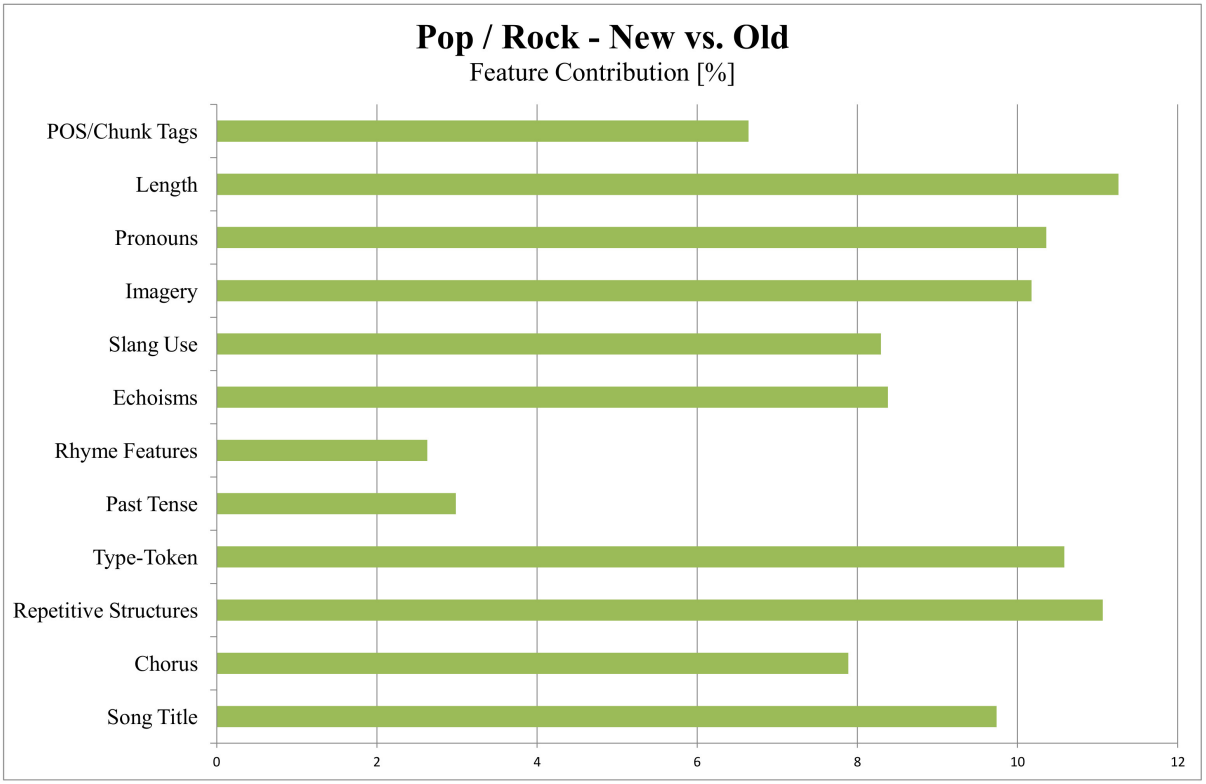


Figure 14: Feature Contributions for Experiment 3

Figure 15: Pop/Rock *old*, top 100 words

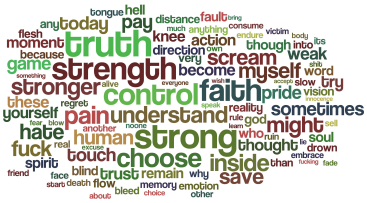


Figure 16: Pop/Rock *mid-age*, top 100 words

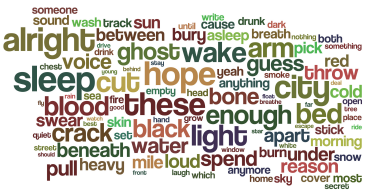


Figure 17: Pop/Rock *new*, top 100 words

6 Conclusion

We showed that lyrics-based statistical models can be employed to perform different music classification tasks: genre detection, distinguishing the best from the worst songs, and predicting the approximate publication time. The latter two are novel, as far as we know. Our study was partly exploratory and we experimented with different feature types, comparing simple n-gram models to more sophisticated approaches. The latter modelled vocabulary, style, semantics, how the writers positions themselves and the story told with respect to the outside world, and overall song structure. Both models were tested in isolation and combined on all three tasks. We found that an n-gram model is often a good first approximation for all of the tasks, however extending the feature space with more sophisticated features nearly always significantly improves the results. We believe that lyrics-based song classification has potential benefits not only for applications such as music retrieval and recommendation but also for basic musicology research by enabling researchers to mine lyrics corpora for interesting trends. Lyrics-based music mining is still in its infancy and would benefit from the development of more sophisticated methods for cleaning, processing and analysing song texts. This applies both to the adaptation of standard NLP tools to this domain and to the further development of stylometric techniques dedicated to analysing lyrics.

References

David Bainbridge, Sally Jo Cunningham, and J. Stephen Downie. 2003. How people describe their music information needs: A grounded theory analysis of music queries. In *Proc. of the International Symposium on Music Information Retrieval (ISMIR 2003)*, pages 221–222.

Marco Baroni and Silvia Bernardini. 2006. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274.

Stephan Baumann and Andreas Klüter. 2002. Super-convenience for non-musicians: Querying mp3 and the semantic web. In *Proc. of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*.

Stephan Baumann, Tim Pohle, and Vembu Shankar. 2004. Towards a socio-cultural compatibility of MIR systems. In *In Proc. of the 5th International Conference on Music Information Retrieval (ISMIR '04)*, pages 460–465.

Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. 2011. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.

M. Besson, F. Fata, I. Peretz, A.-M. Bonnel, and J. Requin. 1998. Singing in the brain: Independence of lyrics and tunes. *Psychological Science*, 9(6):494–498.

Sally Jo Cunningham, J. Stephen Downie, and David Bainbridge. 2005. "The pain, the pain": Modelling music information behavior and the songs we hate. In *Proc. of the International Conference on Music Information Retrieval (ISMIR '05)*, pages 474–477.

Franco Fabbri. 1981. A theory of musical genres: Two applications. In D. Horn and P. Tagg, editors, *Popular Music Perspectives*, pages 52–81. International Association for the Study of Popular Music, Göteborg and Exeter.

Michael Fell. 2014. Lyrics classification. Master's thesis, Saarland University.

Simon Frith. 1988. *Music for Pleasure: Essays in the Sociology of Pop*. Routledge, New York.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November.

Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.

Hussein Hirjee and Daniel G. Brown. 2010. Using automated rhyme detection to characterize rhyming style in Rap music. *Empirical Musicology Review*, 5(4):121–145.

David I. Holmes. 1994. Authorship attribution. *Computers and the Humanities*, 28(2):pp. 87–106.

Sangkyum Kim, Hyungsul Kim, Tim Weninger, and Jiawei Han. 2010. Authorship classification: A syntactic tree mining approach. In *Proceedings of the ACM SIGKDD Workshop on Useful Patterns*, pages 65–73.

Peter Knees, Markus Schedl, and Gerhard Widmer. 2005. Multiple lyrics alignment: Automatic retrieval of song lyrics. In *Proc. of the 6th International Conference on Music Information Retrieval (ISMIR'05)*, pages 564–569.

Tao Li and Mitsunori Ogihara. 2004. Music artist style identification by semi-supervised learning from both lyrics and content. In *Proc. of the 12th Annual ACM International Conference on Multimedia*, pages 364–367.

Beth Logan, Andrew Kositsky, and Pedro Moreno. 2004. Semantic analysis of song lyrics. In *Proc. IEEE International Conference on Multimedia and Expo (ICME '04)*, pages 827–830.

Mitja Lustrek. 2007. Overview of automatic genre identification. Technical Report IJS-DP-9735, Jozef Stefan Institute, Department of Intelligent Systems, Jamova 39, 1000 Ljubljana, Slovenia, January.

Jose P. G. Mahedero, Álvaro Martínez, Pedro Cano, Markus Koppenberger, and Fabien Gouyon. 2005. Natural language processing of lyrics. In *Proceedings of the 13th Annual ACM International Conference on Multimedia*, pages 475–478.

C. Martindale. 1975. Romantic progression: The psychology of literary history. *Hemisphere*.

C. Martindale. 1990. The clockwork muse: The predictability of artistic change. *Basic Books*.

Rudolf Mayer and Andreas Rauber. 2011. Music genre classification by ensembles of audio and lyrics features. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, pages 675–680.

Rudolf Mayer, Robert Neumayer, and Andreas Rauber. 2008a. Combination of audio and lyrics features for genre classification in digital audio collections. In *Proceedings of the 16th ACM international conference on Multimedia*, MM '08, pages 159–168.

Rudolf Mayer, Robert Neumayer, and Andreas Rauber. 2008b. Rhyme and style features for musical genre classification by song lyrics. In *Proc. of the 9th International Conference on Music Information Retrieval (ISMIR08)*.

C. McKay, J. A. Burgoyne, J. Hockman, J. B. L. Smith, G. Vigliensoni, and I. Fujinaga. 2010. Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 213–8.

Robert Neumayer and Andreas Rauber. 2007. Integration of text and audio features for genre classification in music information retrieval. In *Proc. of the 29th European Conference on Information Retrieval (ECIR07)*, pages 724–727.

Eric Nichols, Dan Morris, Sumit Basu, and Christopher Raphael. 2009. Relationships between lyrics and melody in popular music. In *Proc. of the International Symposium on Music Information Retrieval (ISMIR 2009)*.

Keith Salley. 2011. On the interaction of alliteration with rhythm and metre in popular music. *Popular Music*, 30:409–432.

Sam Scott and Stan Matwin. 1998. Text classification using WordNet hypernyms. In *Proc. the Coling-ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, pages 45–51.

Nakatani Shuyo. 2010. Language detection library for Java.

Dean Keith Simonton. 1990. Lexical choices and aesthetic success: A computer content analysis of 154 Shakespeare sonnets. *Computers and the Humanities*, 24(4):pp. 251–264.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2nd edition.

Interpretation of Chinese Discourse Connectives for Explicit Discourse Relation Recognition

Hen-Hsen Huang, Tai-Wei Chang, Huan-Yuan Chen, and Hsin-Hsi Chen
Department of Computer Science and Information Engineering
National Taiwan University
No. 1, Sec. 4, Roosevelt Road, Taipei, 10617 Taiwan
{hhhuang, twchang}@nlg.csie.ntu.edu.tw;
{b00902057, hhchen}@ntu.edu.tw

Abstract

This paper addresses the specific features of Chinese discourse connectives, including types (word-pair and single-word), linking directions (forward and backward linking), positions and ambiguous degrees, and discusses how they affect the discourse relation recognition. A semi-supervised learning method is proposed to learn the probability distributions of discourse functions of connectives from a small labeled dataset and a big unlabeled dataset. The statistics learned from the dataset demonstrates some interesting linguistic phenomena such as connective synonyms sharing similar distributions, multiple discourse functions of connectives, and couple-linking elements providing strong clues for discourse relation resolution.

1 Introduction

Discourse relation labeling determines how two discourse units cohere to each other. A discourse unit may be a clause, a sentence, or a group of sentences. The labeled relation has many potential applications. Coherence is considered as a metric to evaluate the essay writing by essay scorer (Lin et al., 2011). Discourse relations are used to order sentences in an event in a summarization system (Derczynski and Gaizauskas, 2013). Sentiment transition of two clausal arguments is identified based on their discourse relation in sentiment analysis (Hutchinson, 2004; Zhou et al., 2011; Wang et al., 2012; Huang et al., 2013).

The pioneer research of discourse has been established by Hobbs (1985), Polanyi (1988), Hovy and Maier (1992), and Asher and Lascarides (1995). Various discourse relation types have been defined in the frameworks such as Sanders et al. (1992), Hovy and Maier (1992), RST-DT (Carlson et al., 2002), Wolf and Gibson (2005), and PDTB (Prasad et al., 2008). *Temporal*, *Contingency*, *Comparison*, and *Expansion*, the four classes on the top level of PDTB sense hierarchy, are common used in the discourse relation labeling tasks. When two arguments are temporally related, they form a *Temporal* relation. The *Contingency* relation talks about the situation that the event in one argument casually affects the event in the other argument. *Comparison* is used to show the difference between two arguments. The last one relation, *Expansion*, is the most common. An *Expansion* relation either expands the information for one argument in the other one or continues the narrative flow.

In the recent years, discourse relation recognition has been studied for different languages (Afantenos et al., 2012, Cartoni et al., 2013). In explicit English discourse relation labeling tasks, the accuracy of the approach using just the connectives is already quite high, 93.67%, and incorporating the syntactic features raises performance to 94.15% (Pitler and Nenkova, 2009). In our previous work, we investigate Chinese intra-sentential relation detection and show an accuracy of 81.63% and an F-score of 71.11% in the two-way classification (*Contingency* vs. *Comparison* relations) when connectives are

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 632–643, Dublin, Ireland, August 23-29 2014.

introduced as features (Huang and Chen, 2012a). We also report an accuracy of 27.10% and an F-score of 24.27% in the four-way inter-sentential relation classification when only connectives are used (Huang and Chen, 2011). Sporleder and Lascarides (2008) point out some English connectives are often ambiguous between multiple discourse relations or between discourse and non-discourse usage, and Roze et al. (2010) report the ambiguity of French connectives. This issue also occurs in Chinese. Zhou et al. (2012) propose a framework to identify the ambiguous Chinese discourse connectives, and report an F-score of 74.81% in the four-way classification at the intra-sentential level.

The above discourse relation labeling tasks are done on the datasets of different size for different languages at the intra-/inter-sentential levels, thus the results cannot be compared directly. However, these works show a tendency: discourse connectives are useful clues for explicit discourse relation recognition, and the uses of Chinese connectives in discourse relation labeling are more challenging than those of English connectives. In comparison with English, the connectives in Chinese are more and their parts of speech are diverse. There are 100 English explicit connectives annotated in the PDTB 2.0. In Chinese, the linguists report a list of 808 discourse connectives (Cheng and Tian, 1989; Cheng, 2006). In addition, the Chinese discourse connectives have a variety of parts-of-speech. For example, 假設 (jiǎ shè, suppose) is a verb and listed as a discourse connective of the *Contingency* relation.

The following examples address some specific features of Chinese discourse connectives. On the one hand, the two words, "雖然" (suī rán, although) and "但是" (dàn shì, but), which form a word-pair connective, appear in the two discourse units shown in (S1), respectively. These two units demonstrate a *Comparison* relation. On the other hand, "雖然" (suī rán, although) and "但是" (dàn shì, but) can appear individually as single-word connectives shown in (S2)-(S6). The two discourse units have different discourse relations when the single-word connectives appear at different positions, i.e., (S2): *Comparison*, (S3): *Comparison*, (S4): *Expansion*, (S5): *Comparison*, and (S6): *Expansion*. Furthermore, the short word "而" (ér) can be an individual connective, which is interpreted as "而且" (and), "然而" (but), or "因而" (thus), and serves as functions of *Expansion*, *Comparison*, and *Contingency*, respectively. In addition, it can be linked with "雖然" (suī rán, although) and "因為" (yīn wèi, because) to be word-pair connectives, which are interpreted as *Comparison* and *Contingency* functions in (S7) and (S8), respectively. These examples demonstrate word-pair connectives composed of a same word and other words may have different discourse functions, so does the same single-word connective at different positions.

(S1) 雖然湯姆很聰明，但是他並不用功。(Although Tom is smart, he doesn't study hard.)
(S2) 雖然湯姆很聰明，他並不用功。(Although Tom is smart, he doesn't study hard.)
(S3) 他流很多汗，雖然才走幾哩路。(He sweated a lot, although he went only a few miles.)
(S4) 我會好好閱讀，雖然我真的覺得蜘蛛好可怕。(I'll read, even if I really feel spider terrible.)
(S5) 湯姆很聰明，但是他並不用功。(Tom is smart, but he doesn't study hard.)
(S6) 但是在巴黎，他放棄了學醫。(But in Paris, he gave up studying medicine.)
(S7) 雖然你不說，而我一聞就知道。(Although you did not say, I knew that smell.)
(S8) 他因為晚回家，而被媽媽罵了。(Because he came home late, he was scolded by his mother.)

In this paper, we investigate special features of Chinese discourse connectives and apply the results to discourse relation labeling. A semi-supervised learning algorithm is proposed to estimate the probability distribution of the discourse functions of each connective. We address the issue of ambiguity between multiple discourse relations of Chinese connectives. The ambiguity between discourse and non-discourse usages is not our focus in this paper. This paper is organized as follows. Section 2 analyses the types of Chinese connectives and their forward/backward linking properties. Section 3 presents a semi-supervised method to deal with the probability distributions of discourse functions of Chinese connectives and discourse relation labeling. The experimental results are shown and discussed. In Section 4, we further introduce the discourse relation labeler to annotate 302,293 unlabeled sentences and analyze the linguistic phenomena of discourse connectives. We conclude this work in Section 5.

2 Types of Discourse Connectives

From the surface form, there are three kinds of linking elements in Chinese (Li and Thompson, 1981): forward-linking elements, backward-linking elements, and couple-linking elements. Discourse connectives are such kinds of linking elements. A discourse unit containing a forward-linking (backward-linking) element is linked with its next (previous) discourse unit. A couple-linking element is a pair of words that exist in two discourse units (Chen, 1994).

Figure 1 shows connectives and their linking direction. The word-pair connective "雖然...但是" (suī rán…dàn shì, although…but) in (S1) is a couple-linking element. A single-word connective may function as a forward-linking element and/or a backward-linking element. It may be a word appearing in a word-pair connective, e.g., "雖然" (suī rán, although), or a word existing individually, e.g., "以及" (yǐ jí, and). A single-word connective which is the first (the second) word of a word-pair connective may function as a forward-linking (backward-linking) element. The single-word connective "雖然" (suī rán, although) in (S2) is a typical example. It keeps the major discourse function, i.e., *Comparison*, of the word-pair connective that it belongs when it appears in the first discourse unit. In contrast, it may become ambiguous when its position is reversed from the first to the second (i.e., S3 and S4). It may link to the previous or the next discourse units. S5 and S6 have the similar behaviors. The single-word "但是" (dàn shì, but) in (S5) shows a backward-linking. In (S6), it is shifted to the first position and becomes ambiguous. It may be linked to the previous, or to the next discourse units. The correct interpretation depends on the context. These phenomena show a single-word connective may have different senses when it is not at its original position.

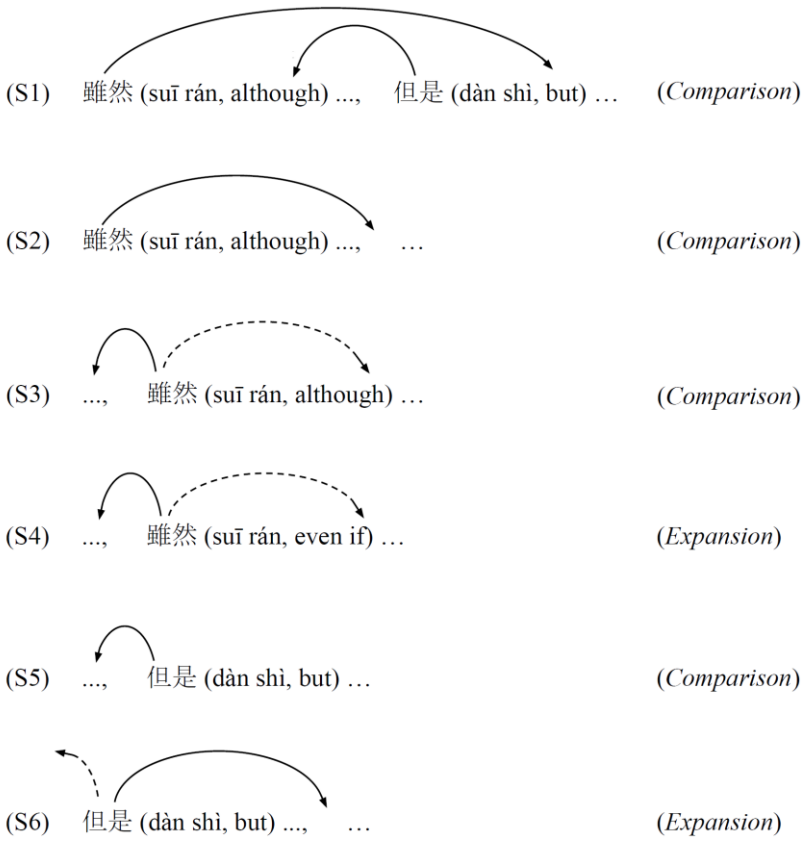


Figure 1: Examples for forward linkging and backward linking.

In this study, we collect 808 discourse connectives based on Cheng and Tian (1989), Cheng (2006), and Lu (2007). The discourse connective lexicon contains 319 single-word and 489 word-pair connectives. Initially, each connective is associated with only one discourse function manually by linguists.

For example, the word-pair connective, "雖然...但是" (suī rán…dàn shì, although…but), is assigned a *Comparison* function. The assignment is one-to-one mapping, thus it cannot capture the complete discourse functions of Chinese connectives. Table 1 shows an overview of the discourse connective lexicon. In this lexicon, *Expansion* is the majority, and *Comparison* is the minority. The percentages of *Contingency* and *Expansion* are close. *Temporal* is the third largest discourse function. Intuitively, the discourse connective lexicon cannot cover all their senses. To learn the probability distribution of the discourse functions of a connective needs a large-scale discourse corpus. Compared with RST-DT (Carlson et al., 2002) and PDTB (Prasad et al., 2008), Chinese discourse corpora are not publicly available (Zhou and Xue, 2012; Huang and Chen, 2012b).

Discourse Function	Number of Connectives	Examples of Single-Word and Word-Pair Discourse Connectives
Temporal	151 (18.69%)	接著 (jiē zhe, then), 最初...現在 (zuì chū…xiàn zài, first...now)
Contingency	261 (32.30%)	因為 (yīn wèi, because), 如...則 (rú…zé, if ... then)
Comparison	87 (10.77%)	即使 (jí shǐ, even if), 儘管…但 (jǐn guǎn…dàn, although…but)
Expansion	309 (38.24%)	另外 (lìng wài, besides), 不僅…而且 (bù jǐn…ér qiě, not only…but also)

Table 1: A Chinese discourse connective lexicon.

3 Learning Discourse Functions of Connectives

This section proposes a semi-supervised learning method to learn the interpretation of discourse connectives from an incomplete and sparse dataset.

3.1 A Semi-Supervised Learning Algorithm

Given a pair of discourse units ds_1 and ds_2 containing an explicit connective *c*, a discourse relation classifier *drc* aims at selecting a relation *r* from the set {*Temporal*, *Contingency*, *Comparison*, *Expansion*} to illustrate how ds_1 and ds_2 cohere to each other. The connective *c* may be a word-pair $c_1 \ldots c_2$, where c_1 and c_2 appear in ds_1 and ds_2, respectively. It may be a single word appearing in ds_1 or ds_2. Each discourse unit is mapped into a representation. Various features from different linguistic levels have been explored in the related work (Huang and Chen, 2011; Huang and Chen, 2012a; Zhou et al, 2011; Zhou et al., 2012). We adopt some of their features shown as follows. Here we focus in particular on the probability distributions of the discourse functions and the positions of connectives.

Length. This feature includes the word counts of ds_1 and ds_2.

Punctuation. The punctuation at the end of ds_2 is regarded as a feature. The possible punctuation includes a full stop, a question mark, or an exclamation mark. The punctuation at the end of ds_1 is dropped from the features because it is always a comma.

Words. The bags of words in ds_1 and ds_2 are considered.

Hypernym. The bags of hypernyms of the words in ds_1 and ds_2 are considered. A Chinese thesaurus, Tongyici Cilin[1], is consulted. The categorization scheme at the fourth level is adopted.

Shared Word. The number of words shared in ds_1 and ds_2 is considered as a feature.

Collocated Word. Collocated words are word pairs mined from the training set. The first and the second words of a pair come from ds_1 and ds_2, respectively.

POS. The bags of parts of speech in ds_1 and ds_2 are considered.

Polarity. Polarity and discourse relation may be related (Huang et al., 2013; Zhou, et al., 2011). For example, a *Comparison* relation implies its two discourse units are contrasting, and some contrasts are presented with different polarities. We estimate the polarity of ds_1 and ds_2 by a lexicon-based approach. The polarity score and the existence of negation are taken as features.

Discourse Connective. A discourse connective *c* is represented as a probability distribution of discourse functions denoted by a quadruple ($P_{(c,temporal)}$, $P_{(c,contingency)}$, $P_{(c,comparison)}$, $P_{(c,expansion)}$), where $P_{(c,temporal)}$, $P_{(c,contingency)}$, $P_{(c,comparison)}$, and $P_{(c,expansion)}$ indicate the probabilities of the four discourse functions of *c*, such that $P_{(c,temporal)}+P_{(c,contingency)}+P_{(c,comparison)}+P_{(c,expansion)}=1$. Section 3.3 shows how we assign the probabilities to each connective in different experimental settings.

Position. The linguistic phenomena discussed in Section 2 show a single-word connective at different position may play different discourse function. Thus, the position of *c* is considered as a feature.

[1] http://ir.hit.edu.cn/

Because the number of Chinese connectives is large (e.g., 808 Chinese connectives in our lexicon) and the large-scale labeled Chinese discourse corpus is not available, how to learn the probability distribution is a challenging issue. This paper proposes a semi-supervised learning method as follows. Its pseudo code is shown in Algorithm 1.

(1) Train a 4-way discourse relation classifier *drc* with the training set and LIBSVM (Chang and Lin, 2011).
(2) Initialize probability distributions of unknown connectives in the test set (see experiments).
(3) Use *drc* to label all the instances in the test set.
(4) Compute the new probability distribution of discourse functions of each connective based on the labeled results in the current run. Maximum likelihood estimation is adopted.
(5) Repeat (3) and (4) until the number of label changes between two successive runs is below 1%.

Algorithm 1. Probability Estimation for the Discourse Functions of Connectives

Input:

D={*Temporal*, *Contingency*, *Comparison*, *Expansion*}: a set of discourse relations and discourse functions for argument pairs and discourse connectives,

$C=\{c_1, c_2, \ldots, c_n\}$: a set of n discourse connectives,

$S=\{s_1, s_2, \ldots, s_p\}$: a set of p labeled argument-pairs $[sa_1, sa_2]$ containing connective $c \in CS \subseteq C$, each with a label $d \in D$, where CS is a set of connectives appearing in S,

$T=\{t_1, t_2, \ldots, t_q\}$: a set of q unlabeled argument-pairs $[ta_1, ta_2]$ containing connective $c \in CT \subseteq C$, where CT is a set of connectives appearing in T.

Output:

$Q=\{q_1, q_2, \ldots, q_n\}$: a probability distribution q_i for connective $c_i \in C$.

Method:

1. Initialization
   1) Train a classifier *drc* using S.
   2) Initialize the probability distribution with equal weight, (0.25, 0.25, 0.25, 0.25), for connective $c \in CT\text{-}CS$, and build $Q^{(0)}$.
   3) $i \leftarrow 0$

2. Relation labeling
   For each $t \in T$, estimate the probabilities of four discourse relations, $P_{(t,temporal)}$, $P_{(t,contingency)}$, $P_{(t.comparison)}$, and $P_{(t.expansion)}$, using the classifier *drc* with $Q^{(i)}$.

3. Updating the probability distribution
   1) For each $c \in C$, compute the average probability of each discourse relation among the argument-pairs containing c in T:
      $P_{(c,tempora)l} \leftarrow$ Average of $P_{(t,temporal)}$ for all t containing c in T.
      $P_{(c,contingency)} \leftarrow$ Average of $P_{(t,contingency)}$ for all t containing c in T.
      $P_{(c,comparison)} \leftarrow$ Average of $P_{(t,comparison)}$ for all t containing c in T.
      $P_{(c,expansion)} \leftarrow$ Average of $P_{(t,expansion)}$ for all t containing c in T.
   2) Form a new $Q^{(i+1)}$
   3) $i \leftarrow i+1$

4. Repeat steps 2-3 until the ratio of the number of label changes by previous and current runs is less than 1%.

5. $Q \leftarrow Q^{(i)}$

3.2 Experimental Setup

For the corpus study of discourse connectives and discourse relations, we refer to a public available Chinese Web POS tagged corpus (Yu et al., 2012). This Chinese POS-tagged corpus is developed based on the ClueWeb09 dataset (CMU, 2009), where Chinese material is the second largest. To capture the discourse functions of individual connectives more accurately, the following three criteria are used to sample sentences:

1. A sentence should contain only two clauses.
2. A sentence should contain exact one discourse connective.

3. The lengths of both clauses in a sentence are no more than 20 Chinese characters.

Total 7,601 sentences composed of two discourse units linked by a connective are sampled from a public available Chinese Web POS tagged corpus (Yu et al., 2012). Each sentence is annotated with a most likely discourse relation selected from {*Comparison*, *Contingency*, *Comparison*, *Expansion*} by three annotators guided by an instruction manual. The majority is taken as the ground truth. A mentor is involved to make a final decision for the tie conditions. The inter-agreement among the annotators is 0.41 in Fleiss' Kappa values, which is a moderate agreement. The discourse category with the lowest inter-annotation agreement is *Temporal*, which annotators usually confuse with *Expansion*. It shows the difficulty to distinguish *Temporal* and *Expansion* even by human. Table 2 shows the statistics of the corpus. More than 50% of pairs are annotated with *Expansion* relation. The second largest group is *Contingency* relation. The percentages of *Temporal* and *Comparison* relations are near. Only 359 connectives appear in the corpus. That reflects the incompleteness issue.

Discourse Relation	# Instances	Percentage
Temporal	846	11.13%
Contingency	1,594	20.97%
Comparison	926	12.18%
Expansion	4,235	55.72%

Table 2: Statistics of the experimental discourse corpus.

This Chinese discourse corpus is used for training and testing. We set up the experiments to simulate the scenario of estimating the probability distributions of discourse functions of the unknown connectives based on the information in the training set. We evaluate the experimental results by 5-fold cross-validation. To ensure the discourse connectives appearing in the test set are mutual exclusive of those connectives in the training set, we split the discourse connectives into 5 mutual exclusive sets and split all the 7,601 sentences into 5 folds according to the 5 sets of discourse connectives.

The kernel of our SVM classifier is the radial basis function. The two parameters, cost *c* and gamma *g*, are optimized by the grid-search algorithm within the range $c \in \{2^{-5}, 2^{-3}, 2^{-1}, \ldots, 2^{15}\}$ and $g \in \{2^{-15}, 2^{-13}, 2^{-11}, \ldots, 2^{3}\}$.

3.3 Results and Discussions

To demonstrate the performance of our proposed semi-supervised learning methods, the following five models are experimented and compared.

M0: Label the relation between two discourse units linked by a connective *c* based on the *c*'s discourse function defined in the connective lexicon. M0 is considered as a baseline model.

M1: Train a 4-way discourse relation classifier *drc* with the training set, then initialize the function probability distributions of the unknown connectives to (0.25, 0.25, 0.25, 0.25), and finally label all the pairs of discourse units by the classifier *drc*. M1 is a supervised-learning method.

M2: M2 model is similar to M1 model except that the probability distribution ($p_{(c,temporal)}$, $p_{(c,contingency)}$, $p_{(c,comparison)}$, $p_{(c,expansion)}$) of an unknown connective is initialized based on its setting in the connective lexicon. The probability of the unique function is set to 1, and the others are set to 0.

M3: M3 is a semi-supervised learning method. In testing, the function probability distributions of the unknown connectives are initialized to (0.25, 0.25, 0.25, 0.25). Discourse relation labeling and probability distribution updating are done iteratively. Finally, all the test instances are labeled, and probability distributions of discourse functions are learned for all test connectives.

M4: M4 is similar to M3 except that the initial probability distributions are set based on the connective lexicon.

Table 3 compares the performances of these five models. The average tendency is M4>M3>M2>M1>M0. It shows the proposed two semi-supervised learning methods are significantly better than the baseline model M0 and the two supervised-learning methods M1 and M2 at p=0.001. The best model is M4, but the performance differences between M3 and M4 are not significant. It demonstrates that both the two initial assignments, i.e., equal-weight assignment and lexicon-based

assignment, are effective. If a connective is not listed in the lexicon due to its coverage, we can still derive its probability distribution starting from the equal-weight approach.

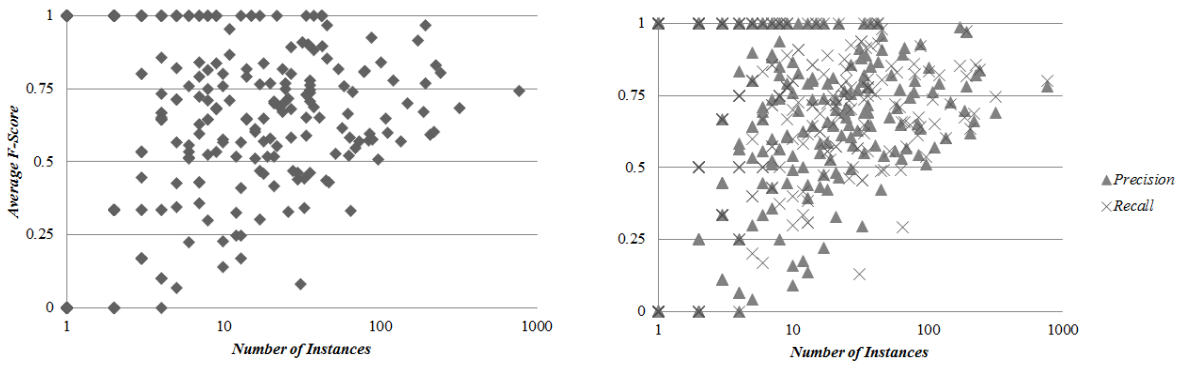
We further examine the individual performance of each discourse relation. Comparing M1 and M3, the semi-supervised classifier (M3) outperforms the supervised classifier (M1) in all three metrics in all the four relations except recall and F-score in the *Temporal* relation. Because more than one half of the pairs of discourse units annotated with *Temporal* relation whose discourse connectives have *Expansion* function in the connective lexicon, some discourse-units of *Temporal* relation are misclassified as *Expansion* relation. That is why the recall is dropped by 8.22% in M3. The precisions of all the four relations are increased. In particular, the precisions of *Temporal*, *Contingency*, and *Comparison* gain more than 10%. The overall F-score is increased 6.61%.

Moreover, M4 is better than M2 in F-score for all the relations. In particular, the precisions of *Temporal*, *Contingency*, and *Comparison* recognition by M4 are greatly increased. In other words, the boosting algorithm tends to correct those instances that are originally misclassified into the *Expansion* relation. The t-test also confirms M4 has a significant improvement over M2 at p=0.001.

The semi-supervised algorithm learns the probability distributions of discourse functions of the unknown connectives from the test instances, so that their size may affect the performance. Figure 2 analyzes how the number of test instances of a connective affects the performance. Each point (x, y) in this figure denote a connective, where x is its total occurrences in the test set, and y is its F-score in Figure 2(a) and its precision/recall in Figure 2(b). We can find (1) many connectives have good performance, (2) connectives containing more test instances demonstrate better performance, and (3) connectives containing fewer instances are sensitive to the evaluation. We treat the probability distribution of discourse functions of each connective as a vector of four real numbers and compute the cosine similarity among the distributions of connectives derived by the connective lexicon, human annotators, and our best model M4. When the 114 connectives containing more than 10 instances are counted, the average cosine similarity between our model and human is 0.940, and the average cosine similarity between the connective lexicon and human is 0.767.

Metric	Model	Temporal	Contingency	Comparison	Expansion	Average
Precision	M0	0.3933	0.7124	0.5092	0.7364	0.6656
	M1	0.5618	0.6005	0.5982	0.7147	0.6595
	M2	0.5024	0.7038	0.5332	0.7529	0.6879
	M3	0.6682	0.7652	0.7749	0.7254	0.7334
	M4	**0.6708**	**0.7773**	**0.7869**	**0.7373**	**0.7344**
Recall	M0	0.3757	**0.6014**	**0.6588**	0.7389	0.6600
	M1	**0.5371**	0.5098	0.4154	0.8114	0.6694
	M2	0.4808	0.5808	0.6207	0.7578	0.6731
	M3	0.4549	0.5387	0.5065	**0.9015**	0.7276
	M4	0.4480	0.5803	0.5821	0.8985	**0.7299**
F-score	M0	0.3843	0.6522	0.5744	0.7376	0.6606
	M1	**0.5492**	0.5515	0.4903	0.7600	0.6644
	M2	0.4913	0.6364	0.5736	0.7553	0.6805
	M3	0.5413	0.6323	0.6126	0.8039	0.7305
	M4	0.5372	**0.6645**	**0.6691**	**0.8099**	**0.7322**

Table 3: Performance comparisons among models.



(a) F-Score (b) Precision/Recall

Figure 2: Effects of the number of test instances for each connective on relation labeling.

4 Further Analyses on a Big Dataset

We further apply the best model (M4) to predict the probability distributions of discourse functions of connectives on a big dataset. For each discourse connective *c*, up to 500 sentences composed of two discourse units linked by *c* are randomly selected from the Chinese Web POS tagged corpus (Yu et al., 2012). The limitation of 500 is set to reduce the imbalance among the discourse connectives. Some connectives appear quite often in the dataset, e.g., the connective "也" (yě, also). Some connectives appear less than 500 times, e.g., "千萬…不然" (qiān wàn…bù rán, must...otherwise) occurs only 212 times. Finally, total 302,293 sentences are extracted and predicted. Because the dataset is very large, it is not easy to evaluate each pair of discourse units. We examine the linguistic phenomena instead. A lexicon of the probability distributions of connectives estimated by M4 is available at http://nlg.csie.ntu.edu.tw/ntu-discourse/.

We sort the discourse connectives by the ratios of their largest relations. In this way, the top connectives in this order almost contain one relation. They can be considered to be less ambiguous. The top ten connectives which appear 500 times are shown in Table 4. Note the bracket notation [ds_1, ds_2] denotes the discourse units where connectives appear. The discourse function defined in the discourse connective lexicon specified in Section 2 is marked in bold. The probabilities of the major discourse function of these connectives are larger than 0.89. The distribution is consistent with the human assignment except the last connective "除非...不然" (chú fēi...bù rán, unless...otherwise), which is assigned to *Contingency* in the lexicon. This connective denotes a negated cause-effect relation between *ds1* and *ds2* in which *ds2* is the effect when *ds1* is not satisfied. In such a case, *ds1* and *ds2* show clear contrast, so that it is reasonable to label this connective with a higher probability of the *Comparison* relation. There are two groups of synonyms in the list: (1) "雖然...不過" (suī rán…bú guò, although…but) and "雖然...可是" (suī rán…kě shì, although…but), and (2) "簡言之" (jiǎn yán zhī, in short) and "簡而言之" (jiǎn ér yán zhī, in short). Table 4 shows that synonyms share similar distributions. The cosine similarities of their probability distributions are 0.99996 and 0.99952, respectively.

The probability of each discourse function of each connective *c* is the average of the probabilities estimated by the classifier, thus the distributions reported by our model is not completely identical to the empirical distribution. For example, all the instances containing the connective "雖然...不過" (suī rán…bú guò, although…but) are labeled with the major discourse function *Expansion*, but the estimated probability of *Expansion* of this connective is 93.47%.

We also sort the discourse connectives by the ratio of their second largest relations. In this manner, the top connectives in this order may have two major discourse functions. In other words, they are ambiguous. Table 5 shows the top ten estimated ambiguous discourse connectives. It is interesting that *Expansion* is one of the two major discourse functions, and the other one shown in bold is the discourse function defined in the connective lexicon. The discourse connectives "緊接著" (jǐn jiē zhe, then), "現在" (xiàn zài, now), "未來" (wèi lái, in the future), and "終於" (zhōng yú, finally), which are defined to have *Temporal* function in the lexicon, frequently occur in the discourse units with *Expansion* relation. The estimated distribution of the connective "而" (ér, and; but; thus) is consistent with the human interpretation, i.e., it has multiple discourse functions.

Chinese single-word connectives are usually put together with other words to form word-pair connectives. Tables 6 and 7 show examples for "雖然" (suī rán, although) and "所以" (suǒ yǐ, so),

Discourse Connectives [ds1, ds2]	Temporal (%)	Contingency (%)	Comparison (%)	Expansion (%)
[簡言之, …] ([in short, …])	2.78	2.08	1.67	**93.47**
[雖然, 不過] ([although, but])	0.77	1.80	**92.70**	4.74
[換言之, …] ([in other words, …])	3.63	2.82	1.53	**92.02**
[雖然, 可是] ([although, but])	0.93	2.11	**91.58**	5.37
[由於, 因此] ([since, therefore])	1.41	**91.07**	0.97	6.55
[說到底,] ([after all, …])	3.17	3.95	2.97	**89.91**
[…, 說到底] ([…, after all])	3.13	4.34	2.84	**89.69**
[簡而言之,] ([in short, …])	5.07	3.20	2.25	**89.48**
[或是, 或是] ([or, or])	3.94	4.51	2.16	**89.39**
[除非, 不然] ([unless, otherwise])	1.04	**3.71**	89.33	5.93

Table 4: Top 10 less-ambiguous connectives estimated by using a big dataset.

Discourse Connectives [ds1, ds2]	Temporal (%)	Contingency (%)	Comparison (%)	Expansion (%)
[緊接著, …] ([then, …])	**48.71**	5.12	1.70	44.46
[…, 即使] ([…, even though])	3.93	5.23	**46.48**	44.36
[現在, …] ([now, …])	**44.31**	7.42	3.42	44.85
[…, 雖然] ([…, although])	3.60	3.68	**44.17**	48.55
[以便, …] ([so that, …])	3.96	**49.83**	2.05	44.16
[目前, 未來] ([now, in the future])	**47.05**	6.41	3.10	43.44
[只有, 才] ([only, then])	4.34	**43.30**	9.33	43.03
[未來, …] ([in the future, …])	**48.21**	6.15	2.85	42.79
[…, 而] ([…, and; but; thus])	3.72	6.13	**42.78**	47.37
[…, 終於] ([…, finally])	**42.39**	6.13	2.99	48.49

Table 5: Some ambiguous connectives estimated by using a big dataset.

respectively. The former is often connected with a word in the second discourse unit to form a couple-linking, while the latter is connected with a word in the first one. We can find word-pair connectives are less ambiguous than single-word connectives in different probabilities. The former ("雖然", suī rán, although) tends to have *Comparison* function. When the word-pair connectives are shorten to single-word connectives, the probability to have *Comparison* function becomes lower. The connective "雖然" (suī rán, although) in the first argument still has probability 0.7639 to have *Comparison* function. When "雖然" (suī rán, although) is moved to the second argument, the probability to serve as *Comparison* function is decreased to 0.4417, which is even lower than that of *Expansion* function. It shows that couple-linking elements provide strong clue to determine discourse relation. Besides, a single-word connective has some tendency to function as either forward linking or backward linking. For example, "雖然" (suī rán, although) is a forward-linking element. Normally, it will link the first discourse unit containing it with the second one. When it appears in the second discourse unit, it becomes ambiguous. The connectives containing "所以" (suǒ yǐ, so) have the similar effects. It tends to be a backward linking element, so its companion appears in the first discourse unit. Its probability to have *Contingency* function decreases from a word-pair connective to a single-word connective. When it appears in the first discourse unit, it may link to the previous sentence at the inter-sentential level.

Some Chinese short words like "而" (ér) is often a part of word-pair connectives. Table 8 shows 10 words which are often connected with "而" (ér) to form word-pair connectives. The word-pair connectives tend to have one major function. When the word-pair connective is "abbreviated" to a single-

Discourse Connectives [ds1, ds2]	Temporal (%)	Contingency (%)	Comparison (%)	Expansion (%)
[雖然,不過] ([although, but])	0.77	1.80	**92.70**	4.74
[雖然,可是] ([although, but])	0.93	2.11	**91.58**	5.37
[雖然,然而] ([while, however])	1.04	2.03	**90.76**	6.17
[雖然,但是] ([although, but])	1.14	2.62	**88.49**	7.74
[雖然,但] ([although, but])	1.48	2.89	**87.54**	8.09
[雖然,還] ([although, still])	2.70	3.43	**85.20**	8.68
[雖然,仍] ([although, still])	3.06	4.10	**81.03**	11.81
[雖然,而] ([although, while])	2.86	5.09	**79.23**	12.82
[雖然,仍然] ([although, still])	3.68	5.70	**77.23**	13.39
[雖然,還是] ([although, still])	3.51	8.54	**75.26**	12.69
[雖然,卻] ([although, still])	4.24	3.71	**74.58**	17.47
[雖然, …] ([although, …])	3.46	5.28	**76.39**	14.87
[…, 雖然] ([…, although])	3.60	3.68	**44.17**	**48.55**

Table 6: Effects of single-word and word-pair connectives containing "雖然" (suī rán, although).

Discourse Connectives [ds1, ds2]	Temporal (%)	Contingency (%)	Comparison (%)	Expansion (%)
[由於, 所以] ([because, so])	1.64	**85.25**	1.77	11.35
[因, 所以] ([because, so])	2.26	**83.20**	1.82	12.72
[因為, 所以] ([because, so])	2.69	**78.03**	2.35	16.93
[既然, 所以] ([since, so])	1.68	**67.32**	6.37	24.63
[…, 所以] ([…, so])	2.82	**50.67**	5.29	**41.22**
[所以, …] ([so, …])	5.71	**50.61**	2.50	**41.18**

Table 7: Effects of single-word and word-pair connectives containing "所以" (so).

word connective, it becomes ambiguous. The discourse function depends on which word-pair connective it is mapped. The determination relies on contextual information.

Table 9 further shows the effects of positions of single-word connectives. The major discourse function of the first 7 sets of connectives is changed when the connectives are shifted from the first discourse unit to the second one. In contrast, the last 3 sets of connectives keep their major discourse function no matter whether they are placed in the first or the second discourse unit. The only difference is the probability to serve as the major discourse function is changed. For example, the probability of the connective "只不過" (zhǐ bú guò, only; just; merely) to have *Comparison* function is increased from 0.6920 to 0.8501 when it is shifted from the first discourse unit to the second one.

Discourse Connectives [ds1, ds2]	Temporal (%)	Contingency (%)	Comparison (%)	Expansion (%)
[不只, 而] ([not only, but])	2.19	4.13	4.92	**88.76**
[不僅, 而] ([not only, but])	2.41	4.56	10.13	**82.89**
[不但, 而] ([not only, but])	3.20	5.14	10.55	**81.11**
[既然, 而] ([since, but])	3.99	13.87	13.42	**68.72**
[固然, 而] ([of course, while])	1.16	2.76	**80.82**	15.24
[雖然, 而] ([although, while])	2.86	5.09	**79.23**	12.82
[儘管, 而] ([although, while])	2.76	43.61	**79.16**	13.71
[由於, 而] ([because, so])	2.02	**79.01**	2.16	16.81
[因, 而] ([because, so])	3.21	**71.03**	2.28	23.49
[因為, 而] ([because, so])	3.11	**49.12**	7.52	40.26
[…, 而] ([…, and; but; thus])	3.71	6.13	**42.78**	**47.37**
[而, …] ([and; but; thus, …])	5.47	8.55	17.00	**68.98**

Table 8: Effects of single-word and word-pair connectives containing "而" (and, but, so).

Discourse Connectives [ds1, ds2]	Temporal (%)	Contingency (%)	Comparison (%)	Expansion (%)
[因而, …] ([therefore, …])	6.26	**64.30**	1.66	27.77
[…, 因而] ([…, therefore])	3.54	28.32	5.15	**62.99**
[只要, …] ([as long as, …])	2.68	**66.02**	5.33	25.98
[…, 只要] ([…, as long as])	2.57	5.49	4.23	**87.71**
[假如, …] ([if, …])	3.51	**57.15**	7.47	31.87
[…, 假如] ([…, if])	3.31	5.21	5.33	**86.16**
[不過, …] ([however, …])	8.17	9.20	23.12	**59.51**
[…, 不過] ([…, however])	2.26	2.39	**80.97**	14.38
[但是, …] ([but, …])	8.56	7.72	20.87	**62.86**
[…, 但是] ([…, but])	2.32	2.90	**75.76**	19.02
[即使, …] ([even though, …])	3.55	5.04	**75.65**	15.75
[…, 即使] ([…, even though])	3.93	5.23	**46.48**	**44.36**
[現在, …] ([now, …])	**44.31**	7.42	3.42	**44.85**
[…, 現在] ([…, now])	8.03	2.88	3.60	**85.49**
[且, …] ([and, …])	7.14	8.43	3.14	**81.29**
[…, 且] ([…, and])	4.62	3.79	2.38	**89.22**
[以及, …] ([as well as, …])	4.83	9.88	2.69	**82.60**
[…, 以及] ([…, as well as])	4.20	4.29	2.33	**89.18**
[只不過, …] ([merely, …])	3.54	4.76	**69.20**	22.50
[…, 只不過] ([…, merely])	1.48	2.00	**85.01**	11.50

Table 9: Effects of positions of single-word connectives.

5 Conclusion

In this paper, we address the issue of the ambiguous discourse functions of Chinese connectives in discourse relation labeling and propose a semi-supervised learning method to estimate the probability distribution of discourse functions of connectives. We examine the constructions of Chinese connectives and their effects on the discourse relation recognition. The proposed approach learns the probability distributions of discourse functions of Chinese connectives from a small labeled dataset and a big unlabeled dataset. The results reflect many interesting linguistic phenomena. We compare the ambiguity degrees of single-word and word-pair connectives, and show the effects of the positions of single-word connectives on the discourse functions. The discourse relation recognizer integrating the

probability distributions and contextual information significantly outperforms the approaches without the knowledge.

This methodology can be extended to estimate the probability distribution of discourse functions of connectives on much finer relation categories. In the current experiments, we focus on explicit discourse relation recognition. The 302,293 labeled sentences in Section 4 can be regarded as a training corpus for implicit discourse relation recognition. Those labeled sentences composed of unambiguous connectives will be sampled from the reference corpus for training an implicit discourse relation recognition system. Furthermore, how to employ the learned probability distributions to deal with discourse units containing multiple connectives will be investigated. In the future, we will tell out the discourse connective and non-discourse connective uses of words and explore their interpretations on the discourse relation recognition. Besides, we will make use of the probability distributions to the relation labeling on more than two clauses and further extend the methodology to experiments at the inter-sentence level.

Acknowledgements

This research was partially supported by Ministry of Science and Technology, Taiwan, under the grants 101-2221-E-002-195-MY3 and 102-2221-E-002-103-MY3, and 2012 Google Research Award. We are also very thankful to the anonymous reviewers for their helpful comments to revise this paper.

References

Stergos Afantenos, Nicholas Asher, Farah Benamara, Myriam Bras, Cécile Fabre, Mai Ho-dac, Anne Le Draoulec, Philippe Muller, Marie-Paule Péry-Woodley, Laurent Prévot, Josette Rebeyrolle, Ludovic Tanguy, Marianne Vergez-Couret, and Laure Vieu. 2012. An Empirical Resource for Discovering Cognitive Principles of Discourse Organisation: the ANNODIS Corpus. In *Proceedings of the18th International Conference on Language Resources and Evaluation* (*LREC 2012*), pages 2727-2734, Istanbul, Turkey.

Nicholas Asher and Alex Lascarides. 1995. Lexical Disambiguation in a Discourse Context. *Journal of Semantics*, 12(1):69-108, Oxford University Press.

Lynn Carlson, Daniel Marcu, and Mary E. Okurowski. 2002. *RST Discourse Treebank*. Linguistic Data Consortium, Philadelphia.

Bruno Cartoni, Sandrine Zufferey, and Thomas Meyer. 2013. Annotating the Meaning of Discourse Connectives by Looking at their Translation: The Translation Spotting Technique. *Dialogue and Discourse*, 4(2):65-86.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1-27:27.

Hsin-Hsi Chen. 1994. The Contextual Analysis of Chinese Sentences with Punctuation Marks. *Literal and Linguistic Computing*, 9(4):281-289.

Shou-Yi Cheng. 2006. *Corpus-Based Coherence Relation Tagging in Chinese Discourse*. Master Thesis, National Chiao Tung University, Hsinchu, Taiwan.

Xianghui Cheng and Xiaolin Tian. 1989. *Xian dai Han yu* (現代漢語), San lian shu dian (三聯書店), Hong Kong.

CMU 2009. ClueWeb09, http://lemurproject.org/clue-web09.php/

Leon Derczynski and Robert Gaizauskas. 2013. Temporal Signals Help Label Temporal Relations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Volume 2: Short Papers, pages 645-650, Sofia, Bulgaria.

Jerry R. Hobbs. 1985. On the Coherence and Structure of Discourse, Report No. CSLI-85-37, Center for the Study of Language and Information, Stanford University. http://www.isi.edu/~hobbs/ocsd.pdf

Eduard H. Hovy and Elisabeth Maier. 1992. Parsimonious or Profligate: How Many and Which Discourse Structure Relations? No. ISI/RR-93-373. Information Sciences Institute, University of Southern California, Marina del Rey.

Hen-Hsen Huang and Hsin-Hsi Chen. 2011. Chinese Discourse Relation Recognition. In *Proceedings of the 5th International Joint Conference on Natural Language Processing* (*IJCNLP 2011*), pages 1442-1446, Chiang Mai, Thailand.

Hen-Hsen Huang and Hsin-Hsi Chen. 2012a. Contingency and Comparison Relation Labeling and Structure Prediction in Chinese Sentences. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue* (*SIGDIAL 2012*), pages 261-269, Seoul, South Korea.

Hen-Hsen Huang and Hsin-Hsi Chen. 2012b. An Annotation System for Development of Chinese Discourse Corpus. In *Proceedings of the 24th International Conference on Computational Linguistics* (*COLING 2012*) Demonstration Papers, pages 223-230, Mumbai, India.

Hen-Hsen Huang, Chi-Hsin Yu, Tai-Wei Chang, Cong-Kai Lin, and Hsin-Hsi Chen. 2013. Analyses of the Association between Discourse Relation and Sentiment Polarity with a Chinese Human-Annotated Corpus. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 70-78, Sofia, Bulgaria.

Ben Hutchinson. 2004. Acquiring the Meaning of Discourse Markers. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics* (*ACL 2004*), pages 684-691, Barcelona, Spain.

Charles N. Li, Sandra A. Thompson. 1981. *Mandarin Chinese: A Functional Reference Grammar*. University of California Press.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically Evaluating Text Coherence Using Discourse Relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics* (*ACL 2011*), pages 997-1006, Portland, Oregon, USA.

Shuxiang Lu. 2007. *Eight Hundred Words of The Contemporary Chinese* (Xian dai Han yu Ba bai Ci). China Social Sciences Press.

Emily Pitler and Ani Nenkova. 2009. Using Syntax to Disambiguate Explicit Discourse Connectives in Text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 13-16, Suntec, Singapore.

Livia Polanyi. 1988. A Formal Model of the Structure of Discourse. *Journal of Pragmatics*, 12(5-6):601-638.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the 6th Language Resources and Evaluation Conference* (*LREC 2008*), pages 2961-2968, Marrakech, Morocco.

Charlotte Roze, Laurence Danlos, and Philippe Muller. 2010. LEXCONN: a French Lexicon of Discourse Connectives. In *Proceedings of the 8th International Workshop on Multidisciplinary Approaches to Discourse* (*MAD 2010*), Moissac.

Ted J. M. Sanders, Wilbert P. M. Spooren, and Leo G. M. Noordman. 1992. Toward a Taxonomy of Coherence Relations. *Discourse Processes*, 15(1):1-35.

Caroline Sporleder and Alex Lascarides. 2008. Using Automatically Labelled Examples to Classify Rhetorical Relations: A Critical Assessment. *Natural Language Engineering*, 14(3):369-416, Cambridge University Press.

Fei Wang, Yunfang Wu, and Likun Qiu. 2012. Exploiting Discourse Relations for Sentiment Analysis. In *Proceedings of the 24th International Conference on Computational Linguistics* (*COLING 2012*), Posters, pages 1311-1320, Mumbai, India.

Florian Wolf and Edward Gibson. 2005. Representing Discourse Coherence: A Corpus-Based Study. *Computational Linguistics*, 31(2):249-287.

Chi-Hsin Yu, Yi-jie Tang and Hsin-Hsi Chen. 2012. Development of a Web-scale Chinese Word N-gram Corpus with Parts of Speech Information. In *Proceedings the 8th International Conference on Language Resources and Evaluation* (*LREC 2012*), pages 320-324, Istanbul, Turkey.

Lanjun Zhou, Binyang Li, Wei Gao, Zhongyu Wei and Kam-Fai Wong. 2011. Unsupervised Discovery of Discourse Relations for Eliminating Intra-sentence Polarity Ambiguities. In *Proceedings of Conference on Empirical Methods in Natural Language Processing* (*EMNLP 2011*), pages 162-171, Edinburgh, UK.

Lanjun Zhou, Wei Gao, Binyang Li, Zhongyu Wei and Kam-Fai Wong. 2012. Cross-lingual Identification of Ambiguous Discourse Connectives for Resource-Poor Language. In *Proceedings of the 24th International Conference on Computational Linguistics* (COLING 2012), pages 1409-1418, Mumbai, India.

Yuping Zhou and Nianwen Xue. 2012. PDTB-style Discourse Annotation of Chinese Text. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics* (*ACL 2012*), pages 67-77, Jeju Island, Korea.

Unsupervised Coreference Resolution by Utilizing the Most Informative Relations

Nafise Sadat Moosavi and Michael Strube
Heidelberg Institute for Theoretical Studies gGmbH
Schloss-Wolfsbrunnenweg 35
69118 Heidelberg, Germany
{nafise.moosavi|michael.strube}@h-its.org

Abstract

In this paper we present a novel method for unsupervised coreference resolution. We introduce a precision-oriented inference method that scores a candidate entity of a mention based on the most informative mention pair relation between the given mention entity pair. We introduce an informativeness score for determining the most precise relation of a mention entity pair regarding the coreference decisions. The informativeness score is learned robustly during few iterations of the expectation maximization algorithm. The proposed unsupervised system outperforms existing unsupervised methods on all benchmark data sets.

1 Introduction

Due to the advent of the internet, the world wide web, social media, the electronic distribution of information and new means of communication, the amount of text available in many different languages is rising. Natural language processing (NLP) is in charge of automatic processing this growing data. NLP research has mainly focused on English and very few other languages. Therefore there is a rich set of annotated corpora for linguistic analysis tasks for these languages. However, there are no such corpora for thousands of other languages. Since unsupervised methods do not require annotated data for learning a model, employing unsupervised methods has become a popular and important area of research in NLP.

In this paper, we propose a new precision oriented method for unsupervised coreference resolution. Our method evaluates the candidate entities of mentions based on the most precise relation of each mention and its candidate entity. Though we develop and evaluate our method for the English language, we intend to apply it to low resource languages in the future.

Common coreference resolution approaches rely on a combination of different features for each decision (for an overview over such approaches, see Ng (2010)). However, a few approaches break down this combination having precision in mind (Baldwin, 1997; Zhou and Su, 2004; Haghighi and Klein, 2009; Lee et al., 2013). The idea of starting with high precision knowledge is used in various NLP tasks including parsing (Borghesi and Favareto, 1982), word alignment (Brown et al., 1993), and named entity classification (Collins and Singer, 1999) with different names like "islands of reliability", "stepping stones", and "cautiousness". Lee et al. (2013) is a successful recent work that implements this idea as "sieve architecture". Lee et al. (2013) first decide on the basis of more precise features, and then they extend these decisions by using less precise features in later sieves. In this system less precise knowledge is used for extending the decisions made by high precision knowledge.

Our proposed inference method goes in the same direction but in a different way. The probability of each coreference decision is computed based on a single relation of a mention-entity. This single relation is the most precise relation that exist between the mention-entity. In contrast to Lee et al. (2013), our inference method will never take into account less precise relations if more precise ones are present. The relative precision of relations can be determined based on our linguistic intuition. If we would rely on linguistic intuition, our system would look much like Lee et al.'s (2013)'s system, except that it processes

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 644–655, Dublin, Ireland, August 23-29 2014.

all mentions in a single sieve, instead of iterating over all mentions for each input relation. However, it is not a trivial task to determine the relative importance of relations for each new relation, new domain, or new language. In this regard, we propose an informativeness score for automatically determining the relative precision of relations.

The informativeness score is computed based on the distinguishing power of relations among coreferring and non-coreferring mentions. We learn the informativeness score in an unsupervised way via few iterations of the Expectation Maximization (EM) algorithm. Overall, our inference method first finds the most precise relation that a mention has with its candidate entity based on the computed informativeness scores. It then computes the probability of joining the mention to the entity based on this best relation and its distribution among all candidate entities.

We empirically validate our approach on the OntoNotes and ACE data sets, showing that despite being entirely unsupervised, our system performs well on all benchmark data sets.

2 Related Work

Early coreference resolution systems were mainly rule-based systems (Lappin and Leass, 1994; Baldwin, 1997). The success of statistical approaches in different NLP tasks together with the availability of coreference annotated corpora (like MUC-6 (Chinchor and Sundheim, 2003) and MUC-7 (Chinchor, 2001)) facilitated a shift from deploying rule-based methods to machine learning approaches in coreference research in the 1990s.

The increasing importance of multilingual processing, brought the deployment of semi-supervised and unsupervised methods into attention for automatic processing of limited resource languages. There are several works which treat coreference resolution as an unsupervised problem (Cardie and Wagstaff, 1999; Angheluta et al., 2004; Haghighi and Klein, 2007; Ng, 2008; Poon and Domingos, 2008; Haghighi and Klein, 2009; Haghighi and Klein, 2010; Kobdani et al., 2011). We compare our results with the unsupervised systems of Haghighi and Klein (2007), Poon and Domingos (2008), Haghighi and Klein (2009), and Kobdani et al. (2011). The Haghighi and Klein (2010) approach is an almost unsupervised approach, and we do not include this system in our comparisons.

We use the expectation maximization algorithm for unsupervised learning. EM has been previously used for coreference resolution (Cherry and Bergsma, 2005; Ng, 2008; Charniak and Elsner, 2009). Cherry and Bergsma (2005) and Charniak and Elsner (2009) use EM for pronoun resolution, and Ng (2008) models coreference resolution as EM clustering. The model parameters of Ng (2008) are of the form $P(f_1, \ldots, f_k|C_{ij})$, where f_i is a feature, and C_{ij} corresponds to the coreference decision of two mentions m_i and m_j. These parameters along with the entity set, are two sets of unknown variables in Ng (2008). He computes the posterior probabilities of entities in the E-step, and determines the parameters from the N-best clustering (i.e. estimated entities) in the M-step. Ng (2008) starts from an initial guess about the entities and determines the parameters based on this initial guess (M-step). In order to compute the N-best clustering, Ng (2008) uses the Bell tree approach of Luo et al. (2004).

The informativeness scores of mention pair relations (Section 3.2.1) are our unknown parameters. Our inference method only requires the ranking of the informativeness scores (and not their exact values). Therefore, it is much easier to estimate the ranking of these parameters than parameters like $P(f_1, \ldots, f_k|C_{ij})$, and our search space for finding an optimized ranking of the informativeness scores is very small. Since it is easier to have an initial guess about the ranking of informativeness scores (rather than guessing an initial entity set), we start from an E-step with a random ranking.

In our experiments, EM converges very fast regardless of the initial state. Indeed, in the M-step, we use our new inference method for computing an estimation of entities. The use of the EM algorithm in our approach is discussed in more detail in Section 3.3.

3 Method Description

Our coreference resolution method is a mention-entity approach which works at mention-mention granularity for processing candidate entities. It estimates entities incrementally while processing the mentions.

For resolving each mention, our inference method scores all candidate entities. For scoring each candidate entity, it first finds the most informative mention-mention relation that exists between the mention and the candidate entity. It then computes the probability of joining the mention to the entity (i.e. the score of the candidate entity) based on the distribution of this relation among all candidate entities of the mention.

In order to find the best mention-mention relation of a mention and an entity, we introduce an informativeness score that scores mention pair relations based on their association with coreference links. This measure is a global measure, and it is computed based on the association analysis of the mention pair relations and coreference links on a whole entity set of all input documents.

We learn the informativeness score in an unsupervised way by using the EM algorithm. Inference is performed at each E-step of the EM iterations. At each E-step, the whole set of entities is constructed from scratch. The informativeness score of the input relations is computed in the M-step based on the estimated entities of the E-step.

3.1 Notations

Assume that M is a mention set of the input document, and each document consists of a set of entities E in which each entity contains one or more mentions of M. $R = \{r_1, \ldots, r_K\}$ is a set of input relations with the following property:

$$\forall r \in R: \; r(m,n) \in \{0,1\} \tag{1}$$

where m and n are two mentions and r can be any arbitrary relation between two mentions like having a specific feature-value (in which the feature can be a combinational feature), or a linguistic rule.

In order to capture the natural left-to-right ordering of mentions, $r(m,n)$ is zero when n is positioned after m in the input document.

3.2 Inference Method

The inference method processes mentions in the text from the beginning of a document to its end. Initially, each mention is in its own entity. For each mention $m \in M$, all partial entities that have been estimated so far (i.e. entities constructed while processing mentions which are positioned before m) are considered as candidate entities of m (i.e. E_m).

For each candidate entity u, the inference method first determines the best relation among all existing mention pair relations between m and u that can indicate a coreference link based on the informativeness score. We call this relation r_u:

$$r_u = \underset{r \in R}{\operatorname{argmax}} \; (IS(r) \times \max_{n \in u} r(m,n)) \tag{2}$$

where $IS(r)$ is the informativeness score of the r relation.

Apparently, when $IS(r) \times \max_{n \in u} r(m,n)$ is equal to zero, u will be removed from E_m.

After finding the most informative relation that exists between m and u (i.e. r_u), we compute the probability of joining m to u based on r_u as follows:

$$Pr[m \rightarrow u] = \frac{\sum_{n \in u} r_u(m,n)}{\sum_{v \in E_m} \sum_{x \in v} r_u(m,x)} \tag{3}$$

Equation 3 computes the local distribution of r_u among all entities belonging to E_m. After computing the probability of Equation 3 for all candidate entities, m will be joined to the $\hat{u}$ that has the highest probability:

$$\hat{u} = \underset{u \in E_m}{\operatorname{argmax}} \, Pr[m \rightarrow u] \tag{4}$$

In case of a tie condition ($\forall_{u,v \in E_m} \; Pr[m \rightarrow u] = Pr[m \rightarrow v]$), $\hat{u}$ will be the entity whose most informative relation is more precise than the most informative relation of the other candidates:

$$\hat{u} = \underset{u \in E}{\operatorname{argmax}} \, [\max_{r \in R} (IS(r) \times \max_{n \in u} r(n,m))] \tag{5}$$

After finding the best candidate entity of m, the method proceeds to find the best entity of the next mention, based on the new updated E.

A mention m will be left in its own entity in two cases: 1) when E_m is empty, and 2) when the value of $Pr[m \rightarrow \hat{u}]$ is below a predefined threshold. We consider this threshold equal to 0.5 in our experiments. This threshold indicates situations in which less than half of the occurrences of $r_{\hat{u}}$ exist between m and $\hat{u}$, and the others are spread among other entities. This entity can be extended while processing later mentions or it may remain as a singleton.

Please note that the inference method does not care about the exact values of $\{IS(r)\}$, and it only needs to have a ranking of the informativeness scores for the given relations in order to select the most informative one.

3.2.1 Informativeness Score

We want to score a set of given relations based on their discriminative power in making coreference decisions. From a statistical point of view, this can be expressed as to determine whether the existence of a relation indicates a coreference link or is due to chance. In this regard, we can examine the following two hypotheses:

$$\textbf{Hypothesis 0: } P(C = 1|r = 1) = p = P(C = 1|r = 0) \tag{6}$$

$$\textbf{Hypothesis 1: } P(C = 1|r = 1) = p_1 \neq p_2 = P(C = 1|r = 0) \tag{7}$$

where $C \in \{0, 1\}$ is a random variable for coreference decisions.

Hypothesis 0 (null hypothesis) formalizes independence (the coreference decisions are independent of relation r). Hypothesis 1 formalizes dependence, which in case $p_1 \gg p_2$ indicates a strong positive association between r and C. This is the pattern that we are interested in.

We use the G^2 log-likelihood ratio statistics for testing these hypotheses. The statistics was introduced to the NLP community by Dunning (1993), and is defined as follows:

$$-2 \log \lambda = 2 \cdot \log \frac{L(H1)}{L(H0)} \tag{8}$$

where $L(H)$ is the likelihood of a hypothesis based on observed data assuming a binomial probability distribution for the existence of r between coreferring mentions. Asymptotically, $-2 \log \lambda$ is χ^2 distributed with one degree of freedom.

Assuming that we have the whole set of entities of input documents, we can use the maximum likelihood estimator to compute p_1, p_2, and p as follows:

$$\begin{aligned}
p_1 &= \frac{\sum_{u \in E} \sum_{m \in u} \sum_{\substack{n \in u \\ n \neq m}} r(m, n)}{\sum_{x \in M} \sum_{\substack{y \in M \\ y \neq x}} r(x, y)} \\
p_2 &= \frac{\sum_{u \in E} \sum_{m \in u} \sum_{\substack{n \in u \\ n \neq m}} (1 - r(m, n))}{\sum_{x \in M} \sum_{\substack{y \in M \\ y \neq x}} (1 - r(x, y))} \\
p &= \frac{\sum_{u \in E} \sum_{m \in u} \sum_{\substack{n \in u \\ n \neq m}} 1}{\sum_{x \in M} \sum_{\substack{y \in M \\ y \neq x}} 1}
\end{aligned} \tag{9}$$

The log-likelihood ratio statistics can be used both for filtering out non-informative relations and for scoring the remaining relations. The filtering is done by comparing the value of $-2 \log \lambda$ to the desired threshold value obtained from the χ^2 table (15.0 in our experiments) and removing the relations that are not significant at the desired level.

Similar to Dunning (1993), the test statistics can be used as a measure for scoring. In our formulation, the test statistics scores given mention pair relations based on their association with coreference links in a way that more precise relations (relations that indicate a coreference link more strongly) will get a

higher score, and less precise relations (relations that are randomly spread among coreferring and non-coreferring mentions) will get a lower score.

The formulation of the log-likelihood ratio in Dunning (1993) is a two-tailed statistical test that if p_1 and p_2 significantly diverge from each other, the $-2\log\lambda$ would get a high value. However, as mentioned above, we are just interested in the cases that p_1 is much higher than p_2, because, otherwise, coreference links among the mentions which have the relation r in common are less frequent than expected.

Therefore, we use the one-sidedness condition as discussed by Kiss and Strunk (2006) for the log-likelihood test. In this case, a relation r is selected as an informative relation for coreference resolution when the $-2\log\lambda$ is larger than the desired threshold, and also $p_1 > p_2$:

$$IS(r) = \begin{cases} -2\log\lambda & \text{if } p_1 > p_2 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

We compute the values of $\{IS(r)\}$ based on entities of the whole set of input documents in order to have a global estimation of the associations in the input data. In order to have a domain- or genre-specific model, one should learn different $\{IS(r)\}$ for each different domain/genre. The domain/genre adaptation is discussed in more detail in the discussion part.

3.3 Learning Method

From what we have discussed so far, $\{IS(r)\}$ values and document entities (E) are two unknown sets of variables that we want to find. When $\{IS(r)\}$ is known, we can estimate entities by using the inference method described in Section 3.2. When the entities are known, we can compute the $\{IS(r)\}$ as described in Section 3.2.1. We can see that these two steps (i.e. determining entities and the informativeness scores), correspond to the E- and M-steps of the expectation maximization algorithm, respectively.

Expectation maximization is an iterative procedure for computing the maximum likelihood estimator of a parameter set when only a subset of data is available. The EM model involves some hidden variables (Z), observed data (X) and a set of unknown parameters (θ). In our modeling, the informativeness scores are the unknown parameters, the observed data is a set of relations corresponding to R, and entities are hidden variables.

In the M-step, the model estimates $\{IS(r)\}$ by using the association analysis of mention pair relations and coreference links over the entire entity set of the input documents. In the E-step, the algorithm performs the inference method of Section 3.2 and reconstructs the whole set of entities based on the given $\{IS(r)\}$ values. As mentioned before, the inference method only needs the ranking of the informativeness scores, and therefore different values of $\{IS(r)\}$ with similar ordering will lead to the same result. Our model starts from an initial E-step, in which the values of $\{IS(r)\}$ are ranked randomly. The iteration between the E- and M-steps continues until $\{IS(r)\}$ converges to steady values. The convergence and the initial state of the EM algorithm are discussed in more detail in the discussion part.

4 Experiments

4.1 Mention Pair Relations

Here is the list of pairwise relations that we use for common and proper nouns:

- *String match:* Two mentions have the same string after removing their post-modifiers.
- *Compatible head match:* Two mentions have the same head, and the pre-modifiers of the anaphor are a subset of the pre-modifiers of the antecedent.
- *Proper head match:* Two proper names have the same head, and they do not contain numeric or location pre-modifiers.
- *Substring:* All words of the anaphor appear in the antecedent (possibly in different order).
- *Acronym:* One mention is an acronym of the other.

For the ACE data, we use additionally the following relations:

- *Apposition:* Two mentions are in an apposition structure.
- *Demonym:* One mention is a name for a resident of a place that derives from the name of the place, and the other mention is the place name itself.
- *Predicate nominative:* The anaphor follows a linking verb and renames or describes the subject mention.
- *Role apposition:* The antecedent (with a noun head) is a modifier of a noun phrase whose head is the anaphor.

For the OntoNotes data sets, *Same speaker* (Lee et al., 2013) is the only feature for resolving pronouns. For the ACE data *Relative pronoun* (i.e. the anaphor is a relative pronoun that modifies the head of the antecedent) is also used. Pronouns, for which we do not have any feature, are linked to the nearest antecedent (based on the Hobbs distance) that currently belongs to a partial entity which is compatible with the pronoun. The compatibility is measured in terms of number, gender, person, animacy, and named entity label. This approach corresponds to the pronoun resolution strategy of the Stanford system.

The differences between the relations of the OntoNotes and ACE corpora is due to the fact that these two corpora have different annotation schemes. Some of the relations mentioned (e.g. *Apposition*) are considered as coreference relations only in the ACE data.

4.2 Data

We evaluate our method on the following data sets:

- **OntoNotes-Dev**: Development set of the OntoNotes data provided by the CoNLL2012 shared task (Pradhan et al., 2012). This data set consists of 303 documents.
- **OntoNotes-Test**: Test set of the OntoNotes data provided by the CoNLL2012 shared task (Pradhan et al., 2012). This data set consists of 322 documents.
- **ACE2004-nwire**: Newswire subset of the ACE 2004 data set consisting of 128 documents. This split of ACE2004 has been utilized in previous work (Poon and Domingos, 2008; Finkel and Manning, 2008; Haghighi and Klein, 2009; Lee et al., 2013).
- **ACE2004-Culotta-Test**: One of the test splits of the ACE 2004 data set that has been used in previous work (Culotta et al., 2007; Bengtson and Roth, 2008; Haghighi and Klein, 2009; Lee et al., 2013). This data set consists of 107 documents.
- **ACE2003-BNEWS**: BNEWS subset of the ACE 2003 data set utilized in Ng (2008) and Kobdani et al. (2011) consisting of 51 documents.
- **ACE2003-NWIRE**: NWIRE subset of the ACE 2003 data set utilized in Ng (2008) and Kobdani et al. (2011) consisting of 29 documents.

4.3 Preprocessing

The mention detection of the Stanford coreference system (Lee et al., 2013) is used for the OntoNotes data sets. We use the predicted information in the OntoNotes data sets for named entity labels, and syntactic roles. For experiments on the ACE data sets, gold mentions are used, so that comparison with previous work is possible. For preprocessing, the Stanford parser (Klein and Manning, 2003) and named entity recognizer (Finkel et al., 2005) are deployed.

We also use the singleton detection of the Stanford system (Recasens et al., 2013) for the OntoNotes data sets. When both mentions are detected as a singleton by the singleton detection module, the value of all their corresponding relations will be set to zero. In other words, $r(m, n)$ is set to zero when both n and m have been detected as a singleton. For examining the effect of the singleton detection module

		MUC			B^3			$CEAF_e$			Avg.
System		R	P	F1	R	P	F1	R	P	F1	F1
OntoNotes-Test											
Supervised	Berkeley	67.48	72.97	70.12	54.4	61.94	57.92	53.84	55.48	54.65	60.90
	IMS	65.23	70.10	76.58	49.41	60.69	54.47	51.34	49.14	50.21	57.42
Rule-based	Stanford	63.95	65.43	64.68	48.65	56.66	52.35	51.04	46.77	48.81	55.28
Unsupervised	This Work	65	64.27	64.64	49.96	55.35	52.52	51.82	46.66	49.11	55.42
OntoNotes-Dev											
Unsupervised	This Work	65.05	65.69	65.37	51.78	58.31	54.85	54.26	48.72	51.34	57.19
	− Singleton	65.44	63.83	64.62	52.26	56.29	54.2	54.63	46.45	50.21	56.34
	& Genre	65.09	65.7	65.39	51.84	58.31	54.89	54.26	48.75	51.36	57.21

Table 1: Experimental results on OntoNotes data sets.

in our inference method, we evaluate our system without this module. The result is shown in Table 1 (specified as "− Singleton"). The results of the Stanford system are also reported using the singleton detection module of Recasens et al. (2013).

	MUC			B^3		
System	R	P	F1	R	P	F1
ACE2003-NWIRE						
This Work	72.92	86.13	78.98	74.68	90.05	81.65
Haghighi07	44.7	55.5	49.5	-	-	-
Ng08	47.0	68.3	55.7	-	-	-
Kobdani11 (UNSEL)	68.6	64.8	66.6	73.6	61.5	67.0
ACE2003-BNEWS						
This Work	67.36	84.72	75.05	70.35	89.56	78.80
Haghighi07	56.8	68.3	62.0	-	-	-
Ng08	56.1	71.4	62.8	-	-	-
Kobdani11 (UNSEL)	65.0	69.5	67.1	65.9	70.2	68.0
ACE2004-nwire						
This Work	74.77	84.53	79.35	74.21	87.50	80.31
Haghighi07	62.3	66.7	64.2	-	-	-
Poon08	71.3	70.5	70.9	-	-	-
Haghighi09	75.09	77.0	76.5	74.5	79.4	76.9
ACE2004-Culotta-Test						
This Work	68.88	82.42	75.04	73.62	88.87	80.53
Haghighi09	77.7	74.8	79.6	78.5	79.6	79.0

Table 2: Comparison with other unsupervised systems on ACE data sets.

4.4 Results

We evaluate our proposed model with the most commonly used metrics for coreference resolution: for the OntoNotes data sets MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), CEAF (Luo, 2005) and their average F1 as used in the CoNLL 2011 and 2012 shared tasks; for the ACE data sets MUC and B^3. The experimental results for the OntoNotes and ACE data sets are presented in Tables 1 and 2, respectively.

On the OntoNotes test set, we compare our method with the three best publicly available coreference systems including the Berkeley system (Durrett and Klein, 2013), the IMS system (Björkelund and Farkas, 2012), and the Stanford system (Lee et al., 2013; Recasens et al., 2013). The Berkeley and IMS systems are both supervised approaches with a rich set of lexical features. At the other hand, the Stanford system is a deterministic system with a set of entity-level features that needs to go through all mentions for incorporating each of the input features. The Stanford system is the winner of the CoNLL2011 shared

OntoNotes-Dev
Same speaker > *Compatible head match* > *Substring* > *String match* > *Proper head match* > *Acronym*
ACE2004-nwire
Compatible head match > *Substring* > *Proper head match* > *String match* > *Demonym* > *Apposition* > *Same speaker* > *Role apposition* > *Relative pronoun* > *Acronym* > *Predicate nominative*

Table 3: The resulting ranking of informativeness scores on different data sets.

task. The IMS system is the 3rd best system on the CoNLL2012 shared task. The Berkeley system is a state-of-the-art supervised coreference system that outperforms both the Stanford and IMS systems. Despite being totally unsupervised and using pairwise features, the results of our system are on par with those of the Stanford system (according to the approximate randomization test, there is no significant difference). The comparison with this state-of-the-art rule based system (Lee et al., 2013), indicates the effectiveness of our coreference resolution approach, as it uses the same preprocessing modules and a simpler and smaller set of features. All results in the Table 1 are reported using the scorer-v7[1] of the CoNLL-2012 shared task (Pradhan et al., 2014).

On the ACE data sets, we compare our performance to those of the unsupervised systems mentioned in Section 2. As Table 2 shows, our method considerably outperforms other unsupervised systems on all data sets (except only for the MUC measure on the ACE2004-Culotta-Test data set).

5 Discussion

5.1 Informativeness Score

As discussed in Section 3.2.1, we determine the discriminative power of mention pair relations in coreference decisions based on the informativeness score (Equation 10), in which the statistical test is computed on the unsupervised estimated set of entities. The resulting ranking of the informativeness score for our input relations is presented in Table 3 on both OntoNotes and ACE data sets.

Another point that needs to be mentioned here is that we are currently using a set of simple and precise input relations. While using these input relations, the informativeness score cannot be efficiently used. The effectiveness of our informativeness score can be usefully assessed with complex relations (i.e. combinatorial features). However, learning of the informativeness scores for complex relations is not possible in a totally unsupervised configuration and one should at least use an informative initial state to guide the learning. We address this issue in our future work.

5.2 Domain/Genre Adaptation

The OntoNotes data set has seven genres regarding the type of text's sources: newswire (NW), broadcast news (BN), broadcast conversation (BC), magazine (MZ), telephone conversation (TC), web data (WB), pivot text (PT). Domain or genre adaptation is one of the current obstacles in language processing. In order to test the effect of genre adaptation in our approach, we try a variant of our approach in which the informativeness scores of the input relations (i.e. $\{IS(r)\}$) are learned separately for each genre. The results of this evaluation are presented in Table 1 by the name "& Genre".

As can be seen in Table 1, the genre-specific variant of our system is performing as well as the base version. This experiment indicates the robustness of our approach regarding the genre/domain adaptation. It can learn an appropriate approximation of the informativeness scores from a small amount of data (i.e. the data provided for a single genre instead of the data from all genres). The learned orderings of the informativeness scores for all genres are presented in Table 4.

When evaluated on each genre separately, the system has the best performance on PT, and the worst performance on the WB genre. The total ordering of genres based on the performance of our system is

[1] http://conll.cemantix.org/2012/software.html

Broadcast conversation, Web data
Same speaker > *Compatible head match* > *Substring* > *String match* > *Proper head match* > *Acronym*
Telephone conversation
Same speaker > *Compatible head match* > *Substring* > *String match* > *Proper head match*
Broadcast news, Newswire
Substring > *Compatible head match* > *String match* > *Proper head match* > *Same speaker* > *Acronym*
Pivot text
Same speaker > *Compatible head match* > *String match* > *Substring* > *Proper head match*
Magazine
Compatible head match > *Substring* > *String match* > *Proper head match* > *Same speaker* > *Acronym*

Table 4: The genre-specific ranking of informativeness scores.

as follow: PT, MZ, TC, BN, NW, BC, WB.

5.3 EM Initial State and Convergence

For the initial state of our EM algorithm, we need a ranking of the informativeness scores of the input relations. We try different initial states for the EM algorithm, from an informative ranking based on linguistic intuition about the precision of input relations to a misleading ranking (the informative order reversed). However, in all cases, the EM algorithm leads to the same ranking (as listed in Table 3). This indicates the robustness of our modeling.

It is more likely that a more precise relation will also get a higher value for its corresponding join probability of Equation 3, because it is unlikely that a precise relation connects a mention to several candidate entities. However, relations with low precision may connect a mention to several different entities, because they are spread over more different entities than relations with higher precision.

In our experiments, for all tested initial states, the model converges in 4 iterations on the OntoNotes data sets and 5 iterations on the ACE data sets.

5.4 Promising Alternative for the Stanford System

Our coreference resolution method is a self- contained approach, that does not need any external linguistic knowledge regarding the coreference relations. However, we can also consider a simple variant of this system in which a predefined ordering of features (based on linguistic intuition) is given, like the Stanford system. In this case, the EM algorithm will be no longer needed, and therefore, the algorithm resolves all mentions in a single iteration.

Therefore, this variant of our system can be considered as an efficient alternative to the Stanford system, that uses a simpler (pairwise instead of entity-based) and smaller (5 instead of 7 string matches) set of relations, and more importantly processes all mentions in a single iteration (instead of iterating over all mentions for each relation), and it still performs as well as its entity-based multi-sieve variant.

6 Conclusions

In this paper, we presented a new unsupervised coreference resolution method. We deploy a new precision-oriented inference method that decides about joining a mention to a candidate entity based on only the most informative mention pair relation that exists between the given mention entity pair. In order to determine the most informative relation of a mention and its candidate entity, we introduce an informativeness score for scoring mention-mention relations based on their global association with

coreference links. A relation whose existence strongly indicates a coreference link will get a high score, and a relation which is randomly spread among coreferring and non-coreferring mentions will get a low score. The informativeness score is robustly learned during a very few iterations of the EM algorithm.

Our proposed method performs well on all benchmark data sets. In the future we intend to apply this robust and efficient approach to new genres, domains, and also new languages.

Acknowledgments

The authors would like to thank Sebastian Martschat for his helpful comments. This work has been funded by the Klaus Tschira Foundation, Heidelberg, Germany. The first author has been supported by a Heidelberg Institute for Theoretical Studies PhD. scholarship.

References

Roxana Angheluta, Patrick Jeuniaux, Rudradeb Mitra, and Marie-Francine Moens. 2004. Clustering algorithms for noun phrase coreference resolution. In *Proceedings of the 7èmes Journées Internationales d'Analyse Statistique des Données Textuelles,* Louvain La Neuve, Belgium, 10–12 March 2004, pages 60–70.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the 1st International Conference on Language Resources and Evaluation,* Granada, Spain, 28–30 May 1998, pages 563–566.

Breck Baldwin. 1997. CogNIAC: High precision coreference with limited knowledge and linguistic resources. In *Proceedings of the ACL Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Text, Madrid, Spain, July 1997*, pages 38–45.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing,* Waikiki, Honolulu, Hawaii, 25–27 October 2008, pages 294–303.

Anders Björkelund and Richárd Farkas. 2012. Data-driven multilingual coreference resolution using resolver stacking. In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pages 49–55.

Luigi Borghesi and Chiara Favareto. 1982. Flexible parsing of discretely uttered sentences. In *Proceedings of the 9th International Conference on Computational Linguistics,* Prague, Czechoslovakia, 5–10 July 1982, pages 37–42.

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

Claire Cardie and Kiri Wagstaff. 1999. Noun phrase coreference as clustering. In *Proceedings of the 1999 SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora,* College Park, Md., 21–22 June 1999, pages 82–89.

Eugene Charniak and Micha Elsner. 2009. EM works for pronoun anaphora resolution. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics,* Athens, Greece, 30 March – 3 April 2009, pages 148–156.

Colin Cherry and Shane Bergsma. 2005. An expectation maximization approach to pronoun resolution. *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 88–95.

Nancy Chinchor and Beth Sundheim. 2003. Message Understanding Conference (MUC) 6. LDC2003T13, Philadelphia, Penn: Linguistic Data Consortium.

Nancy Chinchor. 2001. Message Understanding Conference (MUC) 7. LDC2001T02, Philadelphia, Penn: Linguistic Data Consortium.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of the 1999 SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora,* College Park, Md., 21–22 June 1999, pages 100–110.

Aron Culotta, Michael Wick, and Andrew McCallum. 2007. First-order probabilistic models for coreference resolution. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics,* Rochester, N.Y., 22–27 April 2007, pages 81–88.

Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Greg Durrett and Dan Klein. 2013. Easy victories and uphill battles in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing,* Seattle, Wash., 18–21 October 2013, pages 1971–1982.

Jenny Rose Finkel and Christopher Manning. 2008. Enforcing transitivity in coreference resolution. In *Companion Volume to the Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics,* Columbus, Ohio, 15–20 June 2008, pages 45–48.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics,* Ann Arbor, Mich., 25–30 June 2005, pages 363–370.

Aria Haghighi and Dan Klein. 2007. Unsupervised coreference resolution in a nonparametric Bayesian model. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics,* Prague, Czech Republic, 23–30 June 2007, pages 848–855.

Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing,* Singapore, 6–7 August 2009, pages 1152–1161.

Aria Haghighi and Dan Klein. 2010. Coreference resolution in a modular, entity centered model. In *Proceedings of Human Language Technologies 2010: The Conference of the North American Chapter of the Association for Computational Linguistics,* Los Angeles, Cal., 2–4 June 2010, pages 385–393.

Tibor Kiss and Jan Strunk. 2006. Unsuperivsed multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics,* Sapporo, Japan, 7–12 July 2003, pages 423–430.

Hamidreza Kobdani, Hinrich Schuetze, Michael Schiehlen, and Hans Kamp. 2011. Bootstrapping coreference resolution using word associations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics,* Portland, Oreg., 19–24 June 2011, pages 783–792.

Shalom Lappin and Herbert J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Xiaoqiang Luo, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the Bell Tree. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics,* Barcelona, Spain, 21–26 July 2004, pages 136–143.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the Human Language Technology Conference and the 2005 Conference on Empirical Methods in Natural Language Processing,* Vancouver, B.C., Canada, 6–8 October 2005, pages 25–32.

Vincent Ng. 2008. Unsupervised models for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing,* Waikiki, Honolulu, Hawaii, 25–27 October 2008, pages 640–649.

Vincent Ng. 2010. Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics,* Uppsala, Sweden, 11–16 July 2010, pages 1396–1411.

Hoifung Poon and Pedro Domingos. 2008. Joint unsupervised coreference resolution with Markov Logic. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing,* Waikiki, Honolulu, Hawaii, 25–27 October 2008, pages 650–659.

Sameer Pradhan, Alessandro Moschitti, and Nianwen Xue. 2012. CoNLL-2012 Shared Task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pages 1–40.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the ACL 2014 Conference Short Papers,* Baltimore, Md., 22–27 June 2014. To appear.

Marta Recasens, Marie-Catherine de Marneffe, and Christopher Potts. 2013. The life and death of discourse entities: Identifying singleton mentions. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* Atlanta, Georgia, 9–14 June 2013, pages 627–633.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th Message Understanding Conference (MUC-6)*, pages 45–52, San Mateo, Cal. Morgan Kaufmann.

Guodong Zhou and Jian Su. 2004. A high-performance coreference resolution system using a constraint-based multi-agent strategy. In *the 16th International Conference on Computational Linguistics (COLING)*.

Knowledge Sharing via Social Login: Exploiting Microblogging Service for Warming up Social Question Answering Websites

Yang Xiao[1], **Wayne Xin Zhao**[2], **Kun Wang**[1] **and Zhen Xiao**[1]
[1]School of Electronics Engineering and Computer Science, Peking University, China
[2]School of Information, Renmin University of China, China
{xiaoyangpku, batmanfly}@gmail.com
{wangkun, xiaozhen}@net.pku.edu.cn

Abstract

Community Question Answering (CQA) websites such as Quora are widely used for users to get high quality answers. Users are the most important resource for CQA services, and the awareness of user expertise at early stage is critical to improve user experience and reduce churn rate. However, due to the lack of engagement, it is difficult to infer the expertise levels of newcomers. Despite that newcomers expose little expertise evidence in CQA services, they might have left footprints on external social media websites. Social login is a technical mechanism to unify multiple social identities on different sites corresponding to a single person entity. We utilize the social login as a bridge and leverage social media knowledge for improving user performance prediction in CQA services. In this paper, we construct a dataset of 20,742 users who have been linked across Zhihu (similar to Quora) and Sina Weibo. We perform extensive experiments including hypothesis test and real task evaluation. The results of hypothesis test indicate that both prestige and relevance knowledge on Weibo are correlated with user performance in Zhihu. The evaluation results suggest that the social media knowledge largely improves the performance when the available training data is not sufficient.

1 Introduction

One of the main challenges for social startup websites is how to gain a considerable number of users quickly. A growing number of social startups outsource sign-up process to existing social networking services. They allow users to log in to the services using their existing social media accounts. For example, Quora allows users to log in with their Google, Twitter or Facebook accounts based on the OpenID technology. Lots of startup web services benefit from the huge number of users and rich relationships accumulated by social network sites. Social login helps the newborn web services to collect crowds of users in a short time. Moreover, startup web services can gain reliable profiles through social login. It also offers a convenient mechanism for users to surf the web using a unified social identity (e.g., Twitter account). For example, by the end of 2013, there are about 600,000 web services including mobile applications using social login offered by Sina Weibo.

When we go beyond simple import of profiles and consider the general problem of leveraging knowledge from social media, many subtasks arise. One of them is how to incorporate data from social media and startup web service to better predict user performance. In this paper, we take the largest social based question answering service Zhihu in China, which closely resembles Quora, as the testbed. Different from traditional CQA sites such as Baidu Zhidao, Zhihu have more prominent social features, which supports login with Sina Weibo accounts. Although Zhihu grows quickly and attracts more and more users, about 85% of the users answer fewer than 10 questions and 60% of the users answer fewer than 4 questions in our dataset, which is a large sample of Zhihu.

Previously, many studies have been proposed to improve expertise ranking on CQA services. Link analysis based approaches (Jurczyk and Agichtein, 2007; Zhang et al., 2007) exploit the question-answering relationships to construct a graph and run PageRank or HITS on the graph. Jeon et al. (2006)

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 656–666, Dublin, Ireland, August 23-29 2014.

propose a method based on the non-textual behaviors. Moreover, co-training model (Bian et al., 2009) jointly infers answer quality and user expertise. Liu et al. (2011) formalize expertise ranking as a competition game with the insight that the best answerer beats other answerers in the same question thread. However, the above studies highly rely on the history data, which might not work well for newcomers or users with few answering records. For startup services, many users may not accumulate sufficient data to support the reliable estimation for their expertise levels. Indeed, the importance of newcomers has been noted in related studies, and it has been shown that the effective evaluation of users' performance at an early stage significantly affects the overall development of QA services (Nam et al., 2009; Sung et al., 2013).

In this paper, we propose a method that incorporates social media and social startup data to predict newcomers' performance. This problem is technically challenging due to the heterogeneous characteristics across websites. Given a user, we hypothesize that her capability of contributing high quality answers is dependent on her prestige and relevance. The more contents a user publishes on an area and the higher prestige a user has on social media sites, the higher likelihood that user can offer high quality answers. Thus, the first goal is to precisely measure the relevance between question and a user's tweets. Owing to the short question length and noisy tweet content, this problem brings technical challenges. We make use of user-annotated tags and adopt a translation based model to improve relevance estimation. For prestige, a straightforward way is to use the standard graph based ranking algorithm, however, Zhihu users have very sparse links on Weibo and the standard PageRank algorithm does not work well on sparse graphs. To address it, we add virtual links to alleviate the sparsity problem by finding available paths on a large Weibo graph. Furthermore, we propose a performance biased random walk algorithm and naturally incorporates Zhihu performance history as the supervised information.

We carefully construct a dataset of 20,742 users who have been linked across Zhihu and Weibo, which represent the social startup and the social media site respectively. We first conduct Spearman correlation test for these two hypotheses. Our results have shown that prestige in Weibo has a strong correlation with overall performance in Zhihu. For the performance in question level, we have found that the relevance of Weibo contents is also significantly correlated with answer quality in Zhihu. Based on these findings, we further incorporate the extracted prestige and relevance knowledge into the existing framework for user performance prediction. To simulate the process of history data accumulation, we also conduct experiments with the varying observed number of answers. The experiment results suggest that the borrowed social media knowledge, i.e., prestige and relevance information in Weibo, largely improves the performance when the available training data is not sufficient. Interestingly, we have found that even individual prestige feature can achieve very competitive results.

Although our approach is tested on a joint combination of Weibo and Zhihu, it is equally applicable to other knowledge sharing startup web services. The flexibility of our approach lies in that we identify two important and general types of knowledge that are easy to leverage from external social media sites.

The rest of this paper is organized as follows. The construction of the dataset collection and the problem formulation are given in Section 2 and 3 respectively. Section 4 presents the detailed feature engineering and is followed by the experiment part in Section 5. Finally, the related work and conclusions are given in Section 6 and 7 respectively.

2 Construction of the Dataset Collection

We focus on a popular social question answering website, Zhihu as the studied service. We select Sina Weibo, the largest Chinese microblogging service as the external website to help improve user expertise estimation task in Zhihu. We exploit the social login mechanism to identify the same user across these two platforms: if a user logs in Zhihu with her Weibo account, her Zhihu profile will contain the corresponding Weibo account link. This approach accurately links users across websites.

Zhihu dataset. Zhihu[1] is a social based question answering site in China, which is similar to Quora in terms of overall design and service. Zhihu has three major components: users, questions, and topics. Users on Zhihu can ask and answer questions, furthermore, they can comment on or vote for answers.

[1] http://www.zhihu.com

Each question is usually assigned with a small set of topic tags by the asker and opens a discussion thread consisting of candidate answers. Topics are represented as tags and organized in a directed acyclic graph where a child topic can have multiple parent topics.

Zhihu was founded in January 2011, and we obtain the data between January 2011 and November 2013 via a Web crawler. The dataset contains 266,672 users, 819,125 questions and 2,730,013 answers. These questions are associated with 44,333 topic tags. Since the aim is to examine whether knowledge extracted from Weibo is helpful to improve tasks in Zhihu, we only keep the users who explicitly use social login and get 136,002 cross-site users, which roughly covers 50% of the users in our dataset. For a robust evaluation, we further remove users who have answered fewer than ten questions. Finally, we obtain a total of 20,742 users and summarize the data statistics in Table 1.

#users	#topics	#questions	#answers
20,742	44,333	335,145	883,373

Table 1: Basic statistics of Zhihu dataset for linked users.

Weibo Dataset. Sina Weibo is the largest Chinese microblogging service which has about 500 million registered users by the end of 2012. We have crawled all the detailed information of these 20,742 linked users, including tweets, followers, and following links. These users are indeed active on Weibo and have posted 21,121,955 tweets in total. In later sections, we will adopt the PageRank algorithm to estimate the prestige scores of these linked users, thus we need a dense following graph for reliable estimation. By using these linked users as seeds, we further crawl their followings and followers as well as the following links between all the crawled users. Finally, we obtain 253,361,449 edges between 1,322,425 users. Note that we only use these 20,742 linked users for further study, and the rest are only used to help compute more accurate PageRank scores.

In what follows, we refer to a user who has both a Weibo account and a Zhihu account in our dataset as a *linked user*.

3 Problem Formulation

Users are the most valuable resource in community question answering (CQA) services. Discovering users' expertise at an early stage is important to improve the service quality. A typical task on CQA services is to predict users' performance or expertise: given a question, it aims to estimate the user expertise level and identify experts who can provide good answers to this question.

Borrowing the ideas from information retrieval, we solve the performance prediction task via the learning to rank framework (Liu, 2009). Formally, we assume that there are a set of m questions (i.e., queries) $\mathcal{Q} = \{q^{(1)}, q^{(2)}, q^{(3)}, ...q^{(m)}\}$. A question is associated with a set of $n^{(i)}$ answers $\{a_1^{(i)}, ..., a_{n^{(i)}}^{(i)}\}$ provided by $n^{(i)}$ users $\{u_1^{(i)}, ..., u_{n^{(i)}}^{(i)}\}$ respectively. For each user, let $y_j^{(i)}$ denote the performance score of user $u_j^{(i)}$ with respect to query $q^{(i)}$. A higher value of $y_j^{(i)}$ indicates better performance for query $q^{(i)}$. In our work, we instantiate the performance score by *the number of votes* that a user receives on a question. A feature vector $\boldsymbol{x}_j^{(i)}$ is constructed based on a pair of question and user $(q^{(i)}, u_j^{(i)})$. The aim of the learning task is to derive a ranking function f such that, for each feature vector $\boldsymbol{x}_j^{(i)}$, it outputs a prediction score $f(\boldsymbol{x}_j^{(i)})$ for the performance of user $u_j^{(i)}$ on the question $q^{(i)}$. With this function, when a new question comes, we can predict who will be competent at it.

For prediction tasks, the answer information $\{a_1^{(i)}, ..., a_{n^{(i)}}^{(i)}\}$ is not available during training. Besides users' accumulated history data on Zhihu, external knowledge from Weibo is available to help construct the query-user feature vector. We assume that the studied Zhihu users have already been linked to the corresponding Weibo accounts, and we can obtain their Weibo information, including tweets and followings/followers. The key of the learning to rank framework is how to derive effective features. In our task, we consider two types of features, i.e., Zhihu features and Weibo features. Our focus in this paper is how to leverage microblogging information for improving CQA service, i.e., how to incorporate knowledge from Weibo as features into the learning to rank framework.

4 Feature Engineering

In this section, we discuss how to derive effective features from both Zhihu and Weibo. In particular, we mainly study how to leverage Weibo knowledge for the current task.

4.1 Weibo features

In our work, we focus on two types of Weibo features: prestige and relevance. For prestige, it aims to capture the social status of a user. In our setting, it refers to the status or authority level of a user on online social networks (Anderson et al., 2012). We hypothesize that a user is likely to have similar status levels across multiple online communities, thus the prestige scores of Zhihu users can be roughly estimated based on the rich link information of Weibo. The second type of knowledge we consider is relevance. A user is more likely to be an expert on an area that she is interested in, and Weibo provides a good platform to identify users' interests. Since Weibo and Zhihu are text based websites, we hypothesize that a user will show similar interests on these two medias.

Prestige. Prestige features aim to capture the status of one user. Status characteristic theory posits that one with higher status characteristic is expected to perform better in the group task (Oldmeadow et al., 2003). Prestige estimation has been a classical problem in both web graph analysis and social networking analysis (Easley and Kleinberg, 2012). We are motivated by previous study on authority ranking in Twitter (Kwak et al., 2010), which utilizes the following relations as the evidence of authority. A straightforward way is to run standard PageRank algorithm on the Weibo subgraph consisting of these 20,742 linked users. However, the subgraph of these linked users is very sparse, each linked user has only about 5 out-links to other linked users on average. Such a sparse graph will not produce meaningful ranking results.

Our solution is to add virtual links between linked users. Let N ($N = 20,742$) denote the number of linked users and $\mathbf{M}_{N\times N}$ denote the transition matrix based on the graph of these linked users. Given two users u_i and u_j, we check whether there is a directed path between them on our large Weibo graph. Recall that we have 253,361,449 edges between 1,322,425 users in Weibo dataset. We run the breadth-first search algorithm to find the shortest path between two linked users. If there exists a directed path between two linked users, we add a virtual link between them and set the weight to *the reciprocal of the shortest path length*, i.e., $I(i,j) = \frac{1}{\text{len}(u_i \rightarrow u_j)}$, where $\text{len}(u_i \rightarrow u_j)$ denotes the length of the shortest path between u_i and u_j. In this way, we have $M_{ij} = \frac{I(i,j)}{\sum_k I(i,k)}$. By adding virtual links, we obtain a more dense graph of these linked users. Formally, the standard PageRank algorithm (Brin and Page, 1998) can be formulated as:

$$\boldsymbol{r}^{(n+1)} = \mu \cdot \mathbf{M}^{\mathbf{T}} \cdot \boldsymbol{r}^{(n)} + (1-\mu) \cdot \boldsymbol{y} \tag{1}$$

where μ is the damping factor usually set to 0.85 and $\boldsymbol{y}$ is the restart probability vector usually set to be uniform (Yan et al., 2012). When the algorithm converges, we can obtain the stationary distribution of users (i.e., $\boldsymbol{r}$) as the prestige scores.

The above method assumes that users have same restart probability, which may not be true in reality. Since we are considering improving Zhihu service quality, we incorporate users' history data from Zhihu as supervised information. The main idea is that instead of using a uniform restart distribution $\boldsymbol{y}$, we use a performance biased restart distribution in Eq. 1. We set the restart probability of a user to her average vote ratio based on the questions she has answered. Formally, we set $y_u = \text{Average}(\sum_q \frac{\#vote(q,u)}{\sum_v \#vote(q,v)})$, where $\#vote(q,u)$ denotes the number of votes user u receives on question q and $\sum_v \#vote(q,v)$ denotes the total number of votes that all users receive on question q. We do not use other measures such as best answer ratio because we assume that the history window is very limited and our proposed method provides more robust estimation. Let us further explain the idea. At the beginning of each iteration, each user is assigned to her performance score estimated based on Zhihu data: the more competent she is, the larger score she has. During the iteration, each user begins to collect authority evidence from her incoming neighbors on the Weibo graph. The final score is indeed a trade-off between her own performance on Zhihu and her authority on Weibo.

There are also other measures to consider, e.g., the follower number and the times of being retweeted. In our experiments, we have tried these variants and found that no one is more effective than the above method.

Relevance. Intuitively, a user is more likely to be an expert on an area that she is interested in. In the setting of Zhihu, a user tends to perform better on the topics that are more relevant to her interests. Status characteristic theory also conveys that task relevance is an important factor which affects one's performance (Oldmeadow et al., 2003). Weibo provides a good platform to infer users' interests, which is helpful to derive relevance scores.

We formulate relevance estimation as an information retrieval task. Let $\mathcal{V}$ denote a term vocabulary and w denote a word in $\mathcal{V}$. Note that we take the union of the Weibo vocabulary and Zhihu vocabulary. The interest of a user u is modeled as a multinomial distribution over the terms in $\mathcal{V}$, i.e., $\theta^u = \{\theta^u_w\}_{w\in\mathcal{V}}$. Given a question q, we also model it as a multinomial distribution over the terms in $\mathcal{V}$, i.e., $\theta^q = \{\theta^q_w\}_{w\in\mathcal{V}}$. Following (Zhai, 2008), the relevance score between question q and user u can be estimated by the negative Kullback-Leibler divergence between θ^q and θ^u:

$$\text{Rel}(q,u) = -KL(\theta^q,\theta^u) = -\sum_{w\in\mathcal{V}} p(w|\theta^q)\log\frac{p(w|\theta^q)}{p(w|\theta^u)} \tag{2}$$

We first estimate θ^q. The straightforward way is to estimate θ^q based on the question text. However, the question text is usually short and noisy, which does not yield good results in our experiments. Recall a question is associated with a small set of user-annotated topic tags, and tags are good semantic indicators of the question. A topic tag usually indexes a considerable amount of questions, and we can use tags to leverage semantics from the indexed questions. Formally, we adopt the translation based model (Zhai, 2008) to estimate the question model:

$$\theta^q_w \propto \sum_{t\in q} p(w,t|q) = \sum_{t\in q} p(w|t)p(t|q) \tag{3}$$

where $p(w|t)$ is the translation probability from a tag to a term, and $p(t|q)$ is the empirical distribution of tag t in question q. Here we make an independent assumption: given a tag, the question is independent of a word, i.e., $p(w|t,q) = p(w|t)$. The procedure can be interpreted as follows: sample a tag from the question and then compute the probability of translating the tag into a specific word. We estimate the tag-term translation probability as $p(w|t) = \frac{\#(w,t)+1}{\sum_{w'\in\mathcal{V}}\#(w',t)+|\mathcal{V}|}$, where $\#(w,t)$ denotes the term frequency of w in the question text that tag t indexes. We use the additive-one smoothing.

We also try to incorporate the question text into the above estimation formula. However, it does not result in any improvement. The main reason is that the question words may be too specific, as a comparison, tags provide a general level of semantics, which is more effective to identify expertise areas.

Next, we estimate user interest model θ^u. We consider aggregating all the tweets of a user as a "document", and then estimate the document-term probability as $\theta^u_w = \frac{\#(w,u)+1}{\sum_{w'\in\mathcal{V}}\#(w',u)+|\mathcal{V}|}$, where $\#(w,u)$ denotes the term frequency of w in the aggregated document of user u.

4.2 Zhihu features

Now we describe the features extracted from Zhihu, and we refer to them as *baseline features* since we take the performance of them as a base reference. We summarize these features in Table 2.

These features have been extensively tested to be very effective by previous related studies (Song et al., 2010; Liu et al., 2011), which represent the state-of-art of the current task.

Summary. We have considered two general types of knowledge in social media which are potential to improve user expertise estimation in Zhihu. It is easy to see that our approach can be equally applicable to other third-party websites which is text based and contain manually annotated tags.

Features	Abbr	Formulas
Number of Best Answers	NBA	—
Number of Answers	NA	—
Number of Received Votes	NV	—
Average Number of Votes	AVA	—
Smoothed Average number of Votes	SAVA	$\text{SAVA}(u) = \frac{\sum_q \sigma(v(q,u))}{\text{NA}(u)}, \sigma(x) = \frac{1}{1+e^{(-x)}}$
Best Answer Ratio	BAR	$\text{BAR}(u) = \frac{\text{NBA}(u)}{\text{NA}(u)}$
Smoothed Best Answer Ratio	SBAR	$\text{SBAR}(u) = \frac{\text{BAR}(u)*\text{NA}(u)+\text{BAR}_{\text{avg}}*\text{NA}_{\text{avg}}}{\text{NA}_{\text{avg}}+\text{NA}(u)}$
Average Answer Length	AAL	—

Table 2: List of baseline features with corresponding abbreviations and formulas. Here u denotes a Zhihu user.

5 Experiment

Questions with fewer than five answerers do not receive much attention, and we only keep questions which involve at least six users. In this way, we have obtained a total of 25,262 questions. The number of votes is used as the measure of answer quality. The question threads are sorted by the post time, and we can simulate the cold-start phenomenon to examine the performance of different methods. We split the dataset into a training set and a test set by question threads with the ratio of 3:1. The "history" data of a user is put into the training set and the rest is treated as test data. We further vary the size of "history data" that can be used for performance prediction in three levels, i.e., at most 3, 5, and 10 "historical" question threads have been observed for a given user.

5.1 Hypothesis Testing

In this part, we first examine the fundamental hypotheses of our work: whether Weibo knowledge is potentially effective to improve the performance of tasks in Zhihu. We conduct significance test to examine the correlation between user features extracted from Weibo and user performance in Zhihu. We adopt the Spearman's rank correlation coefficient as the test measure. For a sample of size n, the n raw scores X_i, Y_i are converted to ranks x_i, y_i, and the Spearman correlation coefficient ρ is computed as $\rho = 1 - \frac{6\sum_i d_i^2}{n(n^2-1)}$, where $d_i = x_i - y_i$. The Spearman's coefficient ρ lies in the interval $[-1, 1]$, and a value of "+1" or "-1" indicates a perfect, positive or negative Spearman correlation.

Test of prestige. In our test, the overall performance of a user is estimated by the average vote counts she receives per answer, and the prestige level of a user is estimated by her PageRank score on the original Weibo following graph with a uniform restart probability. With these two measures, it is straightforward to generate two rankings of users, either by user prestige level or by user performance. However, it is noted that ρ is usually very sensitive when the sample size is too large, and it is difficult to obtain robust correlation values in this case. To better capture the overall correlation patterns, we group users according to their prestige levels and examine the correlation degree in the group level. We sort users according to their PageRank scores in a descending order, and split users equally into 100 buckets. The correlation value between performance and PageRank is $\rho = 0.5617$ at the significance level of $9.879e^{-10}$, which indicates there is a strong correlation between performance and prestige.

Test of relevance. Different from prestige, relevance is defined to be question specific, so we cannot perform global correlation analysis. We perform the correlation analysis in the question level. For each question, we have two rankings of involved users: the relevance ranking and the question-specific performance ranking. Let ρ denote the correlation coefficient between the relevance ranking and the performance ranking for a given question. Formally, given a question, we have the null hypothesis H_0 being "ρ is zero", whereas H_1 being "ρ is not zero". If H_0 is rejected, we can conclude that prestige in Weibo is correlated with users' performance on Zhihu for the given question. Our experiments have shown that 14.48% of the questions rejected the H_0 hypothesis at the confidence level of 0.9.

5.2 Evaluation metrics

In the above, we have shown that prestige and relevance knowledge extracted from Weibo are correlated with user performance in Zhihu. Next we are going a step further to examine the feasibility of using these external features to improve user performance ranking in CQA service. In this paper, we consider studying this problem in two aspects: in the first case, we only focus on the user who provides the *best answer*; while in the second case, we focus on the *overall ranking* of all engaged answerers in a given question thread. By following previous studies (Song et al., 2010; Deng et al., 2012), we adopt traditional evaluation metrics in information retrieval for evaluating user performance prediction in CQA services.

Best answer prediction. Our first task is to predict which user will provide the best answer given a question. The user who has received the maximum vote counts in a question thread will be labeled as relevant and the rest will be treated as non-relevant. Then we can adopt the widely used relevance metrics Precision at rank n (P@n) and Mean Reciprocal Rank (MRR).

Top expert recommendation. Unlike best answer prediction, top expert recommendation aims to provide a short list of candidate experts given a question. By following the study (Liu et al., 2011), we use nDCG (normalized Discounted Cumulative Gain) as the evaluation metrics. Let $vote(i)$ denote the vote counts of the answer ranked at i in a system output. To reduce the effects of large outliers, we set the gain value for an answer with the vote counts v to be $\log(v+1)$. The metrics are formally defined as follows:

$$DCG@n = \sum_{i=1}^{n} \frac{\log(vote(i)+1)}{\log(i+1)} \tag{4}$$

$$maxDCG@n = \sum_{i=1}^{n} \frac{\log(vote^*(i)+1)}{\log(i+1)} \tag{5}$$

$$nDCG@n = \frac{DCG@n}{maxDCG@n} \tag{6}$$

where $vote^*$ denotes the vote counts list of the ideal ranking system, i.e., the answer list is sorted by vote counts in the descending order.

Similar to query-specific information retrieval tasks, all our experiments are question specific. For a system, we evaluate its performance of each question and then average all the results as the final performance.

5.3 Results

As studied in Section 3, the above two tasks can be formulated as the learning to rank problem. Following previous work (Song et al., 2010), we adopt SVMRank as the ranking model and implement SVMRank using the tool package SVMLight[2]. We use the linear kernel for SVMRank, and report the results in Table 3 and Table 4.

We refer to the system with all Zhihu features as *Baseline*. We use two ways to compute prestige features: *P+UniformG* denotes the system which implements the standard PageRank algorithm with uniform restart probability, while *P+HisG* denotes the system which implements the biased PageRank algorithm with users' history performance on Zhihu as the restart probability. *Rel* denotes the system with only relevance features and *Baseline+Weibo* denotes the system with all the features.

Analysis of baseline results. The baseline system is built with all Zhihu features, which are estimated using history data, and it is natural to see that the performance of the baseline system improves with the increasing of the history data. Recall that all the question threads in our dataset contain more than six answers, indeed, 36.3% of them contain more than ten answers. A random algorithm to guess the best answer can only achieve a poor P@1 value of 11.07%. Results in Table 3 and Table 4 show that our baseline is competitive even on long question threads. In our experiments, the system performance begins to stay stable when the history window is set to ten question threads since quite a few users have engaged in fewer than ten question threads.

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

History Window Size	Systems	NDCG@1	NDCG@3	NDCG@5
NULL	P+UniformG	0.510	0.555	0.621
	Rel.	0.360	0.434	0.519
$\leq$3 question threads	(B)aseline	0.508	0.582	0.656
	P+HisG	0.550	0.596	0.658
	B.+Weibo	0.580	0.617	0.676
	vs. B.	+14.17%**	+6.01%***	+3.05%***
$\leq$5 question threads	(B)aseline	0.509	0.578	0.658
	P+HisG	0.556	0.603	0.668
	B.+Weibo	0.589	0.625	0.687
	vs. B.	+15.72%***	+8.13%***	+4.41%***
$\leq$10 question threads	(B)aseline	0.534	0.602	0.671
	P+HisG	0.568	0.616	0.679
	B.+Weibo	0.595	0.637	0.696
	vs. B.	+11.42%	+5.81%	+3.73%*

Table 3: Overall ranking performance with varying history window sizes. "*", "**", "***" indicate the improvement is significant at the level of 0.1, 0.05 and 0.01 respectively.

Analysis of the effect of Weibo features. We now incorporate Weibo features and check whether they can help improve the system performance. In Table 3 and Table 4, we present the improvement ratios over baselines with the incorporation of Weibo features. We can see that Weibo features yield a large improvement over the baseline system, especially when the size of history window is small, i.e., $\leq$3 question threads. This indicates the effectiveness of Weibo features on alleviating the cold-start problem in Zhihu. When we have more history data, i.e., $\leq$10 question threads, the improvement becomes smaller.

It is noteworthy that the single prestige feature (i.e., *P+UniformG* and *P+HisG*) achieves good performance. Especially, *P+HisG* obtains very competitive results compared with the baseline system. *P+HisG* naturally combines history data on Zhihu and prestige information on Weibo, which largely improves the standard prestige estimation method *P+UniformG*. As a comparison, the relevance feature is not that effective but still improves the overall performance a bit. These findings indicate that the incorporation of social media data can be a very promising way to improve the tasks of startup services.

History Window Size	Systems	MRR	P@1	P@3
NULL	P+UniformG	0.457	0.261	0.544
	Rel.	0.353	0.157	0.404
$\leq$3 question threads	(B)aseline	0.474	0.263	0.589
	P+HisG	0.498	0.303	0.604
	B.+Weibo	0.516	0.323	0.624
	vs. B.	+8.86%***	+22.81%**	+5.94%**
$\leq$5 question threads	(B)aseline	0.478	0.271	0.590
	P+HisG	0.501	0.303	0.613
	B.+Weibo	0.521	0.327	0.627
	vs. B.	+9.00%***	+20.66%***	+6.27%***
$\leq$10 question threads	(B)aseline	0.494	0.286	0.612
	P+HisG	0.514	0.316	0.627
	B.+Weibo	0.530	0.332	0.643
	vs. B.	+7.29%	+16.08%	+5.07%

Table 4: Best answer prediction performance with varying history window sizes. "*", "**", "***" indicate the improvement is significant at the level of 0.1, 0.05 and 0.01 respectively.

6 Related Work

Our task is built on community question and answering site and researchers have studied CQA from many perspectives. One perspective focuses on user expertise estimation. Generally, there are two principle methods for expertise ranking, interaction graph analysis and interest modeling. Interaction graph based methods (Jurczyk and Agichtein, 2007; Zhang et al., 2007) construct a graph using interaction(e.g., asking and answering) behavior, and rank users using some generalization of PageRank (Brin and Page, 1998) or HITS (Kleinberg, 1999). Interest modeling methods characterize users' interests using question category (Guo et al., 2008) or latent topic modeling (Liu et al., 2005). There are also methods that combine both interest modeling and graph structure (Zhou et al., 2012; Yang et al., 2013) to rank users. Another research perspective on question answering service is quality prediction including answer quality prediction (Harper et al., 2008; Shah and Pomerantz, 2010; Severyn and Moschitti, 2012; Severyn et al., 2013) and question quality prediction (Anderson et al., 2012). However, since the methods mentioned above are based on the history data, the system will experience the cold start problem. Our work explore to what extent can external features help relieve the problem.

This work is also concerned with mining across heterogeneous social networks. Recently, many researches focus on mapping accounts from different sites to one single identity (Zafarani and Liu, 2013; Liu et al., 2013; Kong et al., 2013). By utilizing these recent studies on linking users across communities, our work can be extended to larger scale datasets. From another perspective, cross-domain recommendation has also been widely studied. Zhang et al. (Zhang and Pennacchiotti, 2013a; Zhang and Pennacchiotti, 2013b) explore how Facebook profiles can help boost product recommendation on e-commerce site. Previous work (Zhang et al., 2014) analyze user novelty seeking traits on social network and e-commerce site, which can be used to personalized recommendation and targeted advertisement. Different from simply borrowing user's profiles or psychological traits, our work integrates user footprints from heterogenous social networks and captures performance related characteristics more precisely.

7 Conclusion

In this paper, we take the initiative attempt to leverage social media knowledge for improving the social startup service. We carefully construct a dataset of 20,742 users who have been linked across Zhihu and Weibo, which are social startup and external social media websites respectively. We hypothesize that a user with higher prestige and more relevant Weibo contents to a question is more likely to have better performance.

We first carefully construct testing experiments for these two hypotheses. Our results indicate that prestige in Weibo has strong correlation with overall performance in Zhihu. For question specific performance, we have found that relevance between questions and a user's tweets also correlates with user performance on Zhihu. Based on these findings, we further add prestige and relevance knowledge into existing user performance prediction framework. The experiment results show that prestige and relevance information in Weibo largely improve the performance when the available training data is not sufficient. Moreover, individual prestige feature achieves very competitive results. Our approach is equally applicable to other knowledge sharing web services with appropriate external social media information.

Acknowledgements

The authors would like to thank the anonymous reviewers for their comments. This work is supported by the National Grand Fundamental Research 973 Program of China under Grant No.2014CB340405 and the National Natural Science Foundation of China (Grant No.61170056). The contact author is Zhen Xiao.

References

Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2012. Discovering value from community activity on focused question answering sites: a case study of stack overflow. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 850–858. ACM.

Jiang Bian, Yandong Liu, Ding Zhou, Eugene Agichtein, and Hongyuan Zha. 2009. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In *Proceedings of the 18th international conference on World wide web*, pages 51–60. ACM.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117.

Hongbo Deng, Jiawei Han, Michael R Lyu, and Irwin King. 2012. Modeling and exploiting heterogeneous bibliographic networks for expertise ranking. In *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries*, pages 71–80. ACM.

David Easley and Jon Kleinberg. 2012. Networks, crowds, and markets: Reasoning about a highly connected world.

Jinwen Guo, Shengliang Xu, Shenghua Bao, and Yong Yu. 2008. Tapping on the potential of q&a community by recommending answer providers. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 921–930. ACM.

F Maxwell Harper, Daphne Raban, Sheizaf Rafaeli, and Joseph A Konstan. 2008. Predictors of answer quality in online q&a sites. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 865–874. ACM.

Jiwoon Jeon, W Bruce Croft, Joon Ho Lee, and Soyeon Park. 2006. A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 228–235. ACM.

Pawel Jurczyk and Eugene Agichtein. 2007. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 919–922. ACM.

Jon M Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632.

Xiangnan Kong, Jiawei Zhang, and Philip S Yu. 2013. Inferring anchor links across multiple heterogeneous social networks. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 179–188. ACM.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM.

Xiaoyong Liu, W Bruce Croft, and Matthew Koll. 2005. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 315–316. ACM.

Jing Liu, Young-In Song, and Chin-Yew Lin. 2011. Competition-based user expertise score estimation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 425–434. ACM.

Jing Liu, Fan Zhang, Xinying Song, Young-In Song, Chin-Yew Lin, and Hsiao-Wuen Hon. 2013. What's in a name?: An unsupervised approach to link users across communities. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331.

Kevin Kyung Nam, Mark S Ackerman, and Lada A Adamic. 2009. Questions in, knowledge in?: a study of naver's question answering community. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 779–788. ACM.

Julian Oldmeadow, Michael Platow, Margaret Foddy, and Donna Anderson. 2003. Self-categorization, status, and social influence. *Social Psychology Quarterly*, 66(2):138–152.

Aliaksei Severyn and Alessandro Moschitti. 2012. Structural relationships for large-scale learning of answer re-ranking. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 741–750. ACM.

Aliaksei Severyn, Massimo Nicosia, and Alessandro Moschitti. 2013. Learning adaptable patterns for passage reranking. *CoNLL-2013*, page 75.

Chirag Shah and Jefferey Pomerantz. 2010. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 411–418. ACM.

Young-In Song, Jing Liu, Tetsuya Sakai, Xin-Jing Wang, Guwen Feng, Yunbo Cao, Hisami Suzuki, and Chin-Yew Lin. 2010. Microsoft research asia with redmond at the ntcir-8 community qa pilot task. In *NTCIR-8*.

Juyup Sung, Jae-Gil Lee, and Uichin Lee. 2013. Booming up the long tails: Discovering potentially contributive users in community-based question answering services.

Rui Yan, Mirella Lapata, and Xiaoming Li. 2012. Tweet recommendation with graph co-ranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 516–525. Association for Computational Linguistics.

Liu Yang, Minghui Qiu, Swapna Gottipati, Feida Zhu, Jing Jiang, Huiping Sun, and Zhong Chen. 2013. Cqarank: jointly model topics and expertise in community question answering. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 99–108. ACM.

Reza Zafarani and Huan Liu. 2013. Connecting users across social media sites: a behavioral-modeling approach. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 41–49. ACM.

ChengXiang Zhai. 2008. Statistical language models for information retrieval. *Synthesis Lectures on Human Language Technologies*, pages 1–141.

Yongzheng Zhang and Marco Pennacchiotti. 2013a. Predicting purchase behaviors from social media. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1521–1532. International World Wide Web Conferences Steering Committee.

Yongzheng Zhang and Marco Pennacchiotti. 2013b. Recommending branded products from social media. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 77–84. ACM.

Jun Zhang, Mark S Ackerman, and Lada Adamic. 2007. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th international conference on World Wide Web*, pages 221–230. ACM.

Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, and Xing Xie. 2014. Mining novelty-seeking trait across heterogeneous domains. In *Proceedings of the 23rd international conference on World wide web*, pages 373–384. International World Wide Web Conferences Steering Committee.

Guangyou Zhou, Siwei Lai, Kang Liu, and Jun Zhao. 2012. Topic-sensitive probabilistic model for expert finding in question answer communities. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1662–1666. ACM.

Review Topic Discovery with Phrases using the Pólya Urn Model

Geli Fei
Department of Computer Science, University of Illinois at Chicago, Chicago, USA
gfei2@uic.edu

Zhiyuan Chen
Department of Computer Science, University of Illinois at Chicago, Chicago, USA
czyuanacm@gmail.com

Bing Liu
Department of Computer Science, University of Illinois at Chicago, Chicago, USA
liub@cs.uic.edu

Abstract

Topic modelling has been popularly used to discover latent topics from text documents. Most existing models work on individual words. That is, they treat each topic as a distribution over words. However, using only individual words has several shortcomings. First, it increases the co-occurrences of words which may be incorrect because a phrase with two words is not equivalent to two separate words. These extra and often incorrect co-occurrences result in poorer output topics. A multi-word phrase should be treated as one term by itself. Second, individual words are often difficult to use in practice because the meaning of a word in a phrase and the meaning of a word in isolation can be quite different. Third, topics as a list of individual words are also difficult to understand by users who are not domain experts and do not have any knowledge of topic models. In this paper, we aim to solve these problems by considering phrases in their natural form. One simple way to include phrases in topic modelling is to treat each phrase as a single term. However, this method is not ideal because the meaning of a phrase is often related to its composite words. That information is lost. This paper proposes to use the generalized Pólya Urn (GPU) model to solve the problem, which gives superior results. GPU enables the connection of a phrase with its content words naturally. Our experimental results using 32 review datasets show that the proposed approach is highly effective.

1 Introduction

Topic models such as LDA (Blei et al., 2003) and pSLA (Hofmann 1999) and their extensions have been popularly used to find topics in text documents. These models are mostly governed by the phenomenon called "higher-order co-occurrence" (Heinrich 2009), i.e., how often terms co-occur in different contexts. Word w_1 co-occurring with word w_2 which in turn co-occurs with word w_3 denotes a second-order co-occurrence between w_1 and w_3. Almost all these models regard each topic as a distribution over words. The words under each topic are often sorted according to their associated probabilities. Those top ranked words are used to represent the topic. However, this representation of topics as a list of individual words has some [1]major shortcomings:

- Topics are often difficult to understand or interpret by users unless they are domain experts and also knowledgeable about topic models. In most real-life situations, these are not the case. In some of our applications, we show users several good topics, but they have no idea what they are because many domain phrases cannot be split to individual words. For example, "battery" and "life" are put under the same topic, which is not bad. But the users wondered why "battery" and "life" are the same because they thought words under a topic should somehow have similar meanings. We had to explain that it is due to "battery life." As another example, sentences such as "This hotel has a very nice sandy beach" may cause a topic model to put "hotel" and "sandy" in a topic, which is not wrong but again it is hard to understand by a user who may not be able to connect the two words. Thus in order to interpret topics well, the user must know the phrases (they are split into individual words) that may

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 667–676, Dublin, Ireland, August 23-29 2014.

be used in a domain and how words may be associated with each other. To make the matters worse, in most cases, the topics generated from a topic model are not perfect. There are some wrong words under a topic, which make the interpretation even harder.

- Individual words are difficult to use in practice because in some cases a word under a topic may not have its intended meaning for the topic in a particular sentence context. This can cause many mistakes. For example, in sentiment analysis of product reviews, a topic is often regarded as a set of words indicating a product feature or attribute. This is not true in many cases. For example, if "battery" and "life" are put in one topic, when the system sees "life," it assumes it is related to "battery." But in the sentence "The life expectancy of the machine is about 2 years," this "life" has nothing to do with battery or battery life. This causes an error. If the system can directly use phrases, "battery life" and "life expectancy," the error will not occur.
- Splitting phrases into multiple individual words causes extra co-occurrences that may result in poor or wrong topics involving other words. For example, due to sentences like "Beach staffs are rude" and "The hotel has a nice sandy beach," a topic model may put "staff" and "sandy" under a topic for staff and/or put "beach" and "rude" together under the topic of beach views.

Based on our experiences in opinion mining and social media mining, these are major issues with topic models. We believe that they must be dealt with before wide spread adaptation of topic models in real-life applications. In this paper, we make an attempt to solve this problem. We will use *term* to represent both word and phrase, and use *word* or *phrase* when we want to distinguish them.

One obvious way to consider phrases is to use a natural language parser to find all phrases and then treat each phrase as one term, e.g., "battery life," "sandy beach" and "beach staff." However, the problem with this approach is that it may lose the connection of many related words or phrases in a topic. For example, under the topic for beach, we may not find "sandy beach" because there is no co-occurrence of "sandy beach" and "beach" if we treat "sandy beach" as a single term. This is clearly not a good solution as it may miss a lot of topical terms (words or phrases) for a topic. It can also result in poor topics due to the loss of co-occurrences.

Another obvious solution is to use individual words as they are, but add an extra term representing the phrase. For example, we can turn the sentence "This hotel has a nice sandy beach" to "This hotel has a nice sandy beach <sandy beach>." This solution helps deal with the problem of losing co-occurrences to some extent, but because the words are still treated individually, the three problems discussed above still exist, although the phrase "sandy beach" now can show up in some topics. However, due to the fact that phrases are obviously less frequent than individual words, they may be ranked very low, which make little difference to solving the three problems.

In this paper, we propose a novel approach to solve the problem, which is based on the generalized Pólya urn (GPU) model (Mahmoud 2008). GPU was first introduced into LDA in (Mimno et al., 2011) to concentrate words with high co-document frequency. However, Mimno et al. (2011) and other researchers Chen et al., (2013) still use them in the framework of individual words. In the GPU model, we can deal with the problems above by treating phrases as individual terms and allowing their component words to have some connections or co-occurrences with them. Furthermore, we can push phrases up in a topic as phrases are important for understanding but are usually less frequent than individual words and ranked low in a topic. The intuition here is that when we see a phrase, we also see a small fraction of their component words; and when we see each individual word, we also see a small fraction of its related phrases. Further, in a phrase not all words are equally important. For example, in "hotel staff", "staff" is more important as it is the head noun, which represents the semantic category of the phrase.

Our experiments are conducted using online review collections from 32 domains. We will see that the proposed method produces significantly better results both quantitatively based on the statistical measure of topic coherence and qualitatively based on human labeling of topics and topical terms.

In summary, this paper makes the following contributions:

1. It proposes to consider phrases in topic models, which as we have explained above, is important for accurate topic generation, the use of the resulting topics and human interpretation. As we will see in Section 2, although some prior works exist, they are based on n-grams (Mukherjee and Liu, 2013). They are different from our approach. N-grams can generate many non-understandable phrases. Furthermore, due to infrequency of n-grams (much less frequent than individual words),

typically a huge amount of data is needed in order to produce reasonable topics, which many applications simply do not have.
2. It proposes to use the generalized Pólya Urn (GPU) model to deal with the problems arising in considering phrases. To the best of our knowledge, the GPU model has not been used in the context of phrases. This model not only generates better topics, but also rank phrases relatively high in their topics, which greatly helps understanding of the generated topics.
3. Comprehensive experiments conducted using product and service review collections from 32 domains demonstrate the effectiveness of the proposed model.

2 Related Work

GPU was first introduced to topic modelling in (Mimno et al., 2011), in which GPU is used to concentrate words with high co-document frequency based on corpus-specific co-occurrence statistics. Chen et al. (2013) applied GPU to deal with the adverse effect of using prior domain knowledge in topic modeling by increasing the counts of rare words in the knowledge sets. However, these works still use only individual words.

Topics in most topic models like LDA are unigram distributions over words and assume words to be exchangeable at the word level. However, there exists some work that tries to take word order into consideration by including n-gram language models. Wallach (2006) proposed the Bigram Topic Model (BTM) which integrates bigram statistics with topic-based approaches to document modeling. Wang et al. (2007) proposed the Topical N-gram Model (TNG), which is a generalization of the BTM. It generates words in their textual order by first sampling a topic, then sampling its status as a unigram or bigram, and then sampling the word from a topic-specific unigram or bigram distribution. Although the "bag-of-words" assumption does not always hold in real-life applications, it offers a great computational advantage over more complex models taking word order into account for discovering significant n-grams. Our approach is different from these works in two ways. First, we still follow the "bag-of-words" or rather "bag-of-terms" assumption. Second, we find actual phrases rather than just n-grams. Most n-grams are still hard to understand because they are not natural phrases.

Blei and Lafferty (2009), Liu et al. (2010) and Zhao et al. (2011) also try to extract keyphrases from texts. Their methods, however, are very different because they identify multi-word phrases using relevance and likelihood scores in the post-processing step based on the discovered topical unigrams.

Mukherjee and Liu (2013) and Mukherjee et al. (2013) all try to include n-grams to enhance the expressiveness of their models while preserving the advantages of "bag-of-words" assumption, which has a similar idea as our paper. However, as we point out in the introduction, this way of including phrases/n-grams suffers from several shortcomings. Solving these problems is the goal of our paper.

Finally, since we use product reviews as our datasets, our work is also related to opinion mining using topic models, e.g. (Mei et al., 2007; Lu and Zhai, 2008; Titov and McDonald, 2008; Zhao et al., 2010; Li et al., 2010; Sauper and Barzilay, 2013; Lin and He, 2009; Jo and Oh, 2011). However, none of these models uses phrases.

3 Proposed Model

We start by briefly reviewing the Latent Dirichlet Allocation (LDA) model (Blei et al., 2003). Then we describe the simple Pólya urn (SPU) model, which is embedded in LDA. After that, we present the generalized Pólya urn (GPU) model and discuss how it can be applied to our context. The proposed model uses GPU for its inference. It shares the same graphical model as LDA. However, the GPU inference mechanism is very different from that of LDA, which cannot be reflected in the graphical model or the generative process as it only helps to infer more desirable posterior distributions of topic models.

3.1 Latent Dirichlet Allocation

LDA is a generative probabilistic model for a document collection. It assumes that documents are represented as a mixture of latent topics, and each latent topic is characterized by a distribution over terms. In order to generate a term $w_n^{(d)}$ in document d, where n is its position, we first draw a discrete topic assignment $z_n^{(d)}$ from a document-specific distribution over T topics θ_d, which is drawn from a prior Dirichlet distribution with hyperparameter α. Then we draw a term from the topic-specific distribution

over the vocabulary $\phi_{z_n^{(d)}}$, which is drawn from a prior Dirichlet distribution with hyperparameter β.

For inference, instead of directly estimating θ and ϕ, Gibbs sampling is used to approximate them based on the posterior estimates of latent topic assignment $\boldsymbol{z}$. The Gibbs sampling procedure considers each term in the documents in turn, and estimates the probability of assigning the current term to each topic, conditioned on the topic assignments to all other terms. Griffiths and Steyvers (2004) showed this could be calculated by:

$$p\left(z_n^{(d)} = t \middle| \boldsymbol{z}_{-d,n}, W, \alpha, \beta\right) \propto \frac{C_{t|d} + \alpha}{C_d + T\alpha} \times \frac{N_{w_n^{(d)}|t} + \beta}{N_t + V\beta} \quad (1)$$

where $z_n^{(d)} = t$ represents the topic assignment of term $w_n^{(d)}$ to topic t, and $\boldsymbol{z}_{-d,n}$ refers to the topic assignments of all other terms. W denotes all terms in the document collection, V denotes the size of vocabulary of the collection, T is the number of topics in the corpus, $N_{w|t}$ is the count of term w under topic t, $N_t = \sum_{w'} N_{w'|t}$, and $C_{t|d}$ refers the count of topic t being assigned to some terms in document d, $C_d = \sum_{t'} C_{t'|d}$. All these counts exclude the current term.

3.2 Simple Pólya Urn Model

Traditionally, the *Pólya urn* model is designed in the context of colored balls and urns. In the context of topic models, a term can be seen as a ball of a certain color and the urn contains a mixture of balls with various colors. The classic topic-word (or topic-term) distribution can be reflected by the color proportion of balls in the urn. LDA follows the *simple Pólya urn* (SPU) model, which works as follows: when a ball of a particular color is drawn from an urn, that ball is put back to the urn along with another ball of the same color. This process corresponds to assigning a topic to a term in the Gibbs sampler of LDA. Based on the topic-specific "collapsed" probability of a term w given topic t, $\frac{N_{w|t}+\beta}{N_t+V\beta}$, which is essentially the second ratio in (1), drawing a term w will only increase the probability of seeing w in the future sampling process. This self-reinforcing property is known as "the rich get richer". In the next subsection, we will introduce the *generalized Pólya urn* (GPU) model, which increases the probability of seeing certain other terms when we sample a term.

3.3 Generalized Pólya Urn Model

The *generalized Pólya urn* (GPU) model differs from SPU in that, when a ball of a certain color is drawn, two balls of that color is put back along with a certain number of balls of some other colors. Unlike SPU, GPU sampling not only allows us to see a ball of the same color again with higher probability, but also increases the probability of seeing balls with certain other colors. These additional balls of certain other colors added to the urn increase their proportions in the urn. We call this the *promotion* of these colored balls. Applying the idea, there are two directions of promotion in our application (Note that in each sentence, we need to identify each phrase, but do not need to add any extra information):

1. Word to phrase: When an individual word is assigned to a topic (analogous to drawing a ball of a certain color), each phrase containing the word will be promoted, meaning that the phrase will be added to the same topic with a small count. That is, a fraction of the phrase will be assigned to the topic. This is justified because it is reasonable to assume that the phrase is related to the word to some extent in meaning.
2. Phrase to word: When a phrase is assigned to a topic, each component word in it is also promoted with a certain small count. That is, each word is also assigned the topic by a certain amount. In most cases, the head nouns are more important. Thus, we promote the head nouns more. For example, in "hotel staff", "staff" is the head noun that determines the category of the noun phrase. The rationale of this promotion is similar to that above.

Let $w_n^{(d)}$ be a word and p_w be the word itself or a phrase containing the word $w_n^{(d)}$. v represents a term, and p_v indicates all the related terms of v. The new GPU sampling is as follows:

$$p\left(z_n^{(d)} = t \middle| \boldsymbol{z}_{-d,n}, W, \alpha, \beta, A\right) \propto \frac{C_{t|d} + \alpha}{C_d + T\alpha} \times \frac{\sum_{p\_w} N_{p\_w|t} A_{p\_w, w_n^{(d)}} + \beta}{\sum_v \sum_{p\_v} N_{p\_v|t} A_{p\_v,v} + V\beta} \quad (2)$$

where A is a $V \times V$ real-value matrix, each cell of which contains a real value *virtualcount*, indicating the amount of promotion of a term under a topic when assigning this topic to another term. V is size of all terms. The new model retains the document-topic component of standard LDA, which is the first ratio in (1), but replaces the usual Pólya urn topic-word (topic-term) component, the second ratio in (1), with a generalized Pólya urn framework (Mahmoud 2008; Mimno et al., 2011). The simple Pólya urn model is a simplified version of GPU in which matrix A is an identity matrix. In this paper, A is an asymmetric matrix because the main goal of using GPU is to promote the less frequent phrases in the documents.

4 EXPERIMENTS

In this section, we evaluate the proposed method of considering phrases in topic discovery, and compare it with three baselines. The first baseline discovers topics using LDA in a traditional way without considering phrases, i.e., using only individual words. We refer to this baseline as *LDA*(*w*). The second baseline considers phrases by treating each whole phrase as a separate term in the corpus. We refer to this baseline as *LDA*(*p*). The third baseline considers phrases by keeping individual component words in the phrases as they are, but also adding phrases as extra terms. We refer to this baseline as *LDA*(*w_p*). We refer to our proposed method as *LDA*(*p_GPU*). Note that for those words that are not in any phrases, they are treated as individual words (or unigrams).

Data Set: We use product reviews from 30 sub-categories (types of product) in the electronics domain from Amazon.com. The sub-categories are "Camera", "Mouse", "Cellphone," etc (see the whole list below Figure 1). Each domain contains 1,000 reviews. Besides, we also use a collection of hotel reviews and a collection of restaurant reviews from TripAdvisor.com and Yelp.com. The hotel review data contains 101,234 reviews, and the restaurant review data contains 25,459 reviews. We thus have a total of 32 domains. We ran the Stanford Parser to perform sentence detection, lemmatization and POS tagging. Punctuations, stopwords, numbers and words appearing less than 5 times in each dataset are removed. Domain names are also removed, e.g., word "camera" for the domain Camera, since it co-occurs with most words in the dataset, leading to high similarity among topics/aspects.

Sentences as Documents: As noted in (Titov and McDonald, 2008), when standard topic models are applied to reviews as documents, they tend to produce topics that correspond to global properties of products (e.g., product brand name), but cannot separate different product aspects or features well. The reason is that all reviews of the same product type basically evaluate the same aspects of the product type. Only the brand names and product names are different. Thus, using individual reviews for modelling is ineffective for finding product aspects or features, which are our topics. Although there are approaches which model sentences (Jo and Oh, 2011; Zhao et al., 2010; Titov and McDonald, 2008), we take the approach in (Brody and Elhadad, 2010; Chen et al., 2013), dividing each review into sentences and treating each sentence as an independent document.

Noun Phrase Detection: Although there are different types of phrases, in this first work we focus only on noun phrases as they are more representative of topics in online reviews. We will deal with other types of phrases in the future. Our first step is thus to obtain all noun phrases from each domain. Due to the efficiency issue of full natural language parser with a huge number of reviews, instead of applying the Stanford Parser to recognize noun phrases, we design a rule-based approach to recognize noun phrases as consecutive nouns based on POS tags of sentences. Although the Stanford Parser may give us better noun phrases, our simple method serves the purpose and gives us very good results. In fact, based on our initial experiments, the Stanford Parser also gives many wrong phrases.

Parameter Settings: In all our experiments, the posterior inference was drawn after 2000 Gibbs sampling iterations with a burn-in of 400 iterations. Following (Griffiths and Steyvers, 2004), we fix the Dirichlet priors as follows: for all document-topic distributions, we set α=50/K, where K is the number of topics. And for all topic-term distributions, we set β=0.1. We also experimented with other settings of these priors and did not notice much difference.

Setting the number of topics/aspects in topic models is often tricky as it is difficult to know the exact number of topics that a corpus has. While non-parametric Bayesian approaches (Teh et al., 2005) do exist for estimating the number of topics, it's not the focus of this paper. We empirically set the number of topics to 15. Although 15 may not be optimum, since all models use the same number, there is no bias against any model.

In Section 3.3, we introduced the *promotion* concept for the GPU model. When we sample a topic for a word, we add *virtualcount* of topic assignment to all its related phrases. However, not all words in a phrase are equally important. For example, in phrase "hotel staff", "staff" is more important, and we call such words the head nouns. In this work, we apply a simple method used in (Wang et al., 2007), which is to always assume that the last word in a noun phrase is the head noun. Although we are aware of the potential harm to our model when we promote a wrong word, we will leave it as our future work. Again, because we want to connect phrases with their component words and promote the rank of phrases in their topics, we add less virtual counts to individual words. Thus, we add 0.5 * *virtualcount* to the last word in a phrase and add 0.25 * *virtualcount* to all other words. We set *virtualcount* = 0.1 in our experiments empirically.

Based on the discovered topics, we conduct statistical evaluation using topic coherence, human evaluation and also a case study to quantitatively and qualitatively show the superiority of the proposed method in terms of both interpretability and topic wellness.

4.1 Statistical Evaluation

Perplexity and KL-divergence are often used to evaluate topic models statistically. However, researchers have found that perplexity on held-out documents is not always a good predictor of human judgments of topics (Chang et al., 2009). In our application, we are not concerned with the test on future data using the hold-out set. KL-divergence measures the difference of distributions, and thus can be used to measure the distinctiveness of topics. However, distinctiveness of topics does not necessarily mean human agreeable topics. Recently, Mimno et al. (2011) proposed a new measure called topic coherence, which has been shown to correlate with human judgments of topic quality quite well. Higher topic coherence score indicates higher quality of topics, i.e., better topic coherence. Topic coherence is computed as below.

$$TC\left(t; V^{(t)}\right) = \sum_{m=2}^{M} \sum_{l=1}^{m-1} log \frac{D\left(v_m^{(t)}, v_l^{(t)}\right) + 1}{D\left(v_l^{(t)}\right)} \tag{3}$$

in which $D(v)$ is the document frequency of term v (i.e., the number of documents with at least one term v) and $D(v, v')$ is the co-document frequency of term v and term v' (i.e., the number of documents containing both term v and term v'). Also, $V^{(t)} = \left(v_1^{(t)}, \ldots, v_M^{(t)}\right)$ is the list of M most probable terms in topic t. 1 is added as a smoothing count to avoid taking the logarithm of zero.

We thus use this measure to score all four experiments. Figure 1 and Figure 2 show the topic coherence using top 15 terms and top 30 terms respectively on the 32 different domains. Notice the topic coherence is a negative value, and a smaller absolute value is better than a larger one. Firstly, we can see from both charts that our proposed model *LDA*(*p_GPU*) is better than all other three baselines by a large margin. Secondly, the performance of the other three baselines are quite similar. In general, *LDA*(*p*) is slightly worse than the other two baselines. It is because replacing many words with phrases decreases the number of co-occurrences in the corpus. In contrast, *LDA*(*w_p*) is slightly better than the other two baselines on most domains because some frequent phrases add more reliable co-occurrences in the corpus. However, as we point out in the introduction, some problems still exist. Firstly, it does not solve the problem of phrases and their component words having different meanings, and thus artificially creating such wrong co-occurrences may damage the overall performance. Secondly, even if the number of co-occurrences increases, most of the phrases are still too infrequent to be ranked high in their associated topics to be useful in helping users understand the topic.

In order to test the significance of the improvement, we conduct paired *t*-tests on the topic coherence results. Using both 15 top terms and 30 top terms, statistical tests show that our proposed method, *LDA*(*p_GPU*), outperforms all three baselines significantly ($p < 0.01$). However, there's no significant improvement between any pair of the three baselines.

4.2 Manual Evaluation

Although several statistical measures, such as perplexity, KL-divergence and topic coherence, have been used to statistically evaluate topic models, since topic models are mostly (including ours) unsupervised,

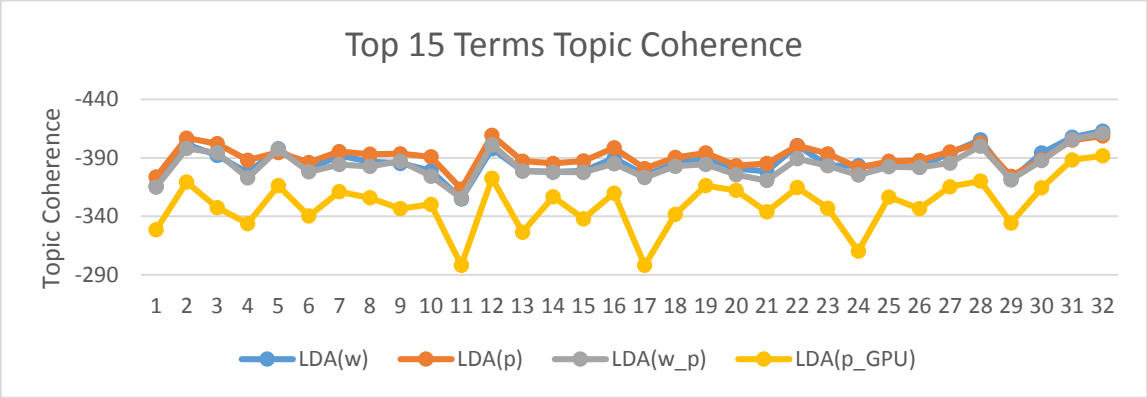


Figure 1: Topic coherence of the top 15 terms of each model on each of the 32 datasets. Notice that since topic coherence is a negative value, a smaller absolute value is better than a larger one.

Domain/dataset names are listed as follows (1:Amplifier; 2:BluRayPlayer; 3:Camera; 4:CellPhone; 5:Computer; 6:DVDPlayer; 7:GPS; 8:HardDrive; 9:Headphone; 10:Keyboard; 11:Kindle; 12:MediaPlayer; 13:Microphone; 14:Monitor; 15:Mouse; 16:MP3Player; 17:NetworkAdapter; 18:Printer; 19:Projector; 20:RadarDetector; 21:RemoteControl; 22:Scanner; 23:Speaker; 24:Subwoofer; 25:Tablet; 26:TV; 27:VideoPlayer; 28:VideoRecorder; 29:Watch; 30:WirelessRouter; 31:Hotel; 32:Restaurant).

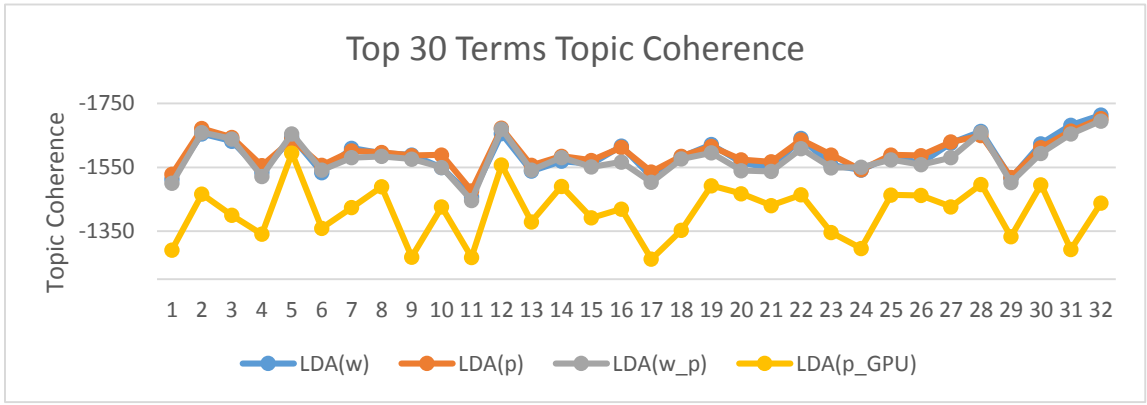


Figure 2: Topic coherence of the top 30 terms of each model on each dataset. Notice again that since topic coherence is a negative value, a smaller absolute value is better than a larger one. X-axis indicates the domain id numbers, whose names are listed below Figure 1.

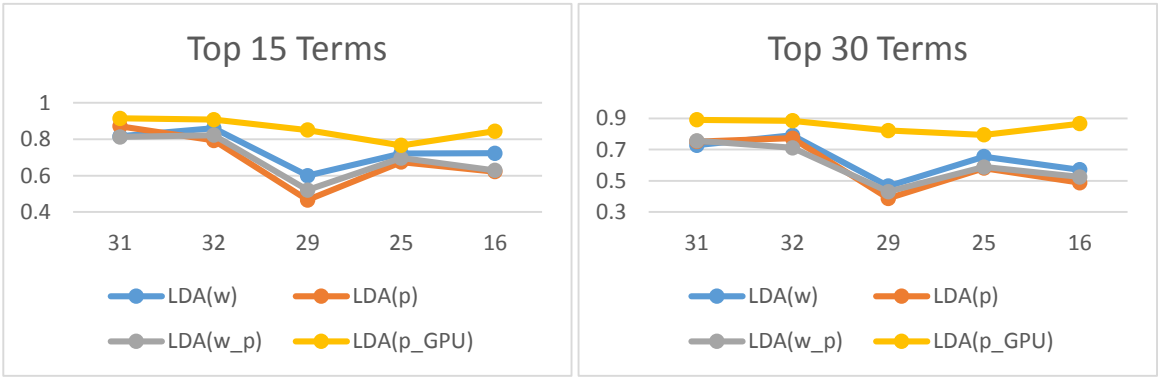


Figure 3: Human evaluation on five domains using top 15 and top 30 terms. X-axis indicates the domain id numbers, whose corresponding domain names are listed below Figure 1. Y-axis indicates the ratio of correct topic terms.

statistical measures may not always correlate with human interpretations or judgments. Thus, in this sub-section, we perform a manual evaluation through manual labeling of topics and topical terms.

Manual labeling was done by two annotators, who are familiar with reviews and topic models. The labeling was carried out in two stages sequentially: (1) labeling of topics and (2) labeling of topical terms in each topic. After the first stage, an annotator agreement is computed and then the two annotators discuss about the disagreed topics to reach a consensus. Then, they move on to the next stage to label the top ranked topical terms in each topic (based on their probabilities in the topic). For the annotator

Table 1: Example topics discovered by LDA(w) and LDA(p_GPU)

Hotel		**Restaurant**		**Watch**	
LDA(w)	LDA(p_GPU)	LDA(w)	LDA(p_GPU)	LDA(w)	LDA(p_GPU)
bed	clean	service	service	hand	big
comfortable	comfortable	star	friendly	minute	hand
small	quiet	staff	server	hour	minute
sleep	sleep	atmosphere	staff	beautiful	cheap
size	large	friendly	atmosphere	casual	hour
large	spacious	server	waiter	christmas	automatic
tv	size	waiter	attentive	setting	seconds
pillow	king size bed	attentive	star	condition	line
king	pillow	reason	service staff	worth	hour hand
chair	queen size bed	decor	star service	weight	durable
table	bed size	quick	customer service	red	analog hand
mattress	bed nd pillow	customer	table service	press	hand move
clean	bed sheet	waitress	delivery service	gift	hand line
double	bed linen	tip	rush hour service	run	seconds hand
big	sofa bed	pleasant	service attitude	functionality	hand sweep

Tablet		**MP3Player**	
LDA(w)	LDA(p_GPU)	LDA(w)	LDA(p_GPU)
screen	screen	battery	battery
touch	size	headphone	hour
software	easier	life	battery life
hard	pro	media	price
pad	touch screen	car	worth
option	bigger	windows	charge
version	area	hour	replacement
website	inch	decent	free
angle	screen protector	reason	market
car	screen size	xp	aaa battery
charger	inch screen	program	aa battery
ipod	draw	aaa	purchase
worth	home screen	window	hour battery
gb	screen look	set	aaa
drive	line	pair	life

agreement, we compute Kappa scores. The Kappa score for topic labeling is 0.838, and the Kappa score for topical terms labeling is 0.846. Both scores indicate strong agreement in the labeling.

Evaluation measure. A commonly used evaluation measure in human evaluation is *precision@n* (or *P@n* for short), which is the precision at a particular rank position *n* in a topic. For example, *Precision*@5 means the precision of the top ranked 5 terms for a topic. To be consistent with the automatic evaluation, we use *Precision*@15 and 30. Top 15 terms is usually sufficient to represent the topic. However, since we include phrases in our experiments which may lead to some other terms ranked lower than using only words, we labeled up to top 30 terms. The *Precision*@n measure is also used in (Zhao et al., 2010) and some others, e.g., (Chen et al., 2013).

In our experiments, we labeled four results for each domain, i.e., those of *LDA*(*w*), *LDA*(*p*), *LDA*(*w_p*) and *LDA*(*p_GPU*). Due to the large amount of human labeling effort, we only labeled 5 domains. We find that it is sometimes hard to figure out what some of the topics are about and whether some terms are related to a topic or not, so we give the results to our human evaluators together with the phrases in each domain extracted by our rules in order to let them be familiar with the domain vocabulary. The human evaluation results are shown in Figure 3.

Results and Discussions. Again, we conduct paired *t*-tests on the human evaluation results of top 15 and 30 terms. Statistical tests show that our proposed method, *LDA*(*p_GPU*), outperforms all other three methods significantly ($p < 0.05$) using both top 15 and top 30 terms. However, there's no significant improvement between any pair of the three baselines.

4.3 Case Study

In order to illustrate the importance of phrases in enhancing human readability, we conduct case study using one topic from each of the five manually labeled domains. Due to space limitations, we only compare the results of our model *LDA*(*p_GPU*) with *LDA*(*w*).

In the above table, we notice that with phrases, the topics are much more interpretable than only reading individual words given by *LDA*(*w*). For example, "hand" in "Watch" domain given by *LDA*(*w*) is quite confusing at first, but in *LDA*(*p_GPU*), "hour hand" makes it more understandable. Another example is "aaa" in "MP3Player" domain. It is quite confusing at first, but "aaa battery" should make it more interpretable by an application user who is not familiar with topic models or does not have extensive domain knowledge. Also, due to wrong co-occurrences created by individual words in a phrase, the *LDA*(*w*) results contain much more noise than those of *LDA*(*p_GPU*).

5 CONCLUSION

This paper proposed a new method to consider phrases in discovering topics using topic models. The method is based on the generalized Pólya urn (GPU) model, which allows us to connect phrases with their component words during the inference and rank phrases higher in their related topics. Our method preserves the advantages of "bag-of-words" assumption while preventing the side effects that traditional methods have when considering phrases. We tested our method against three baselines across 32 different domains, and demonstrated the superiority of our method in improving the topic quality and human interpretability both quantitatively and qualitatively.

References

David M. Blei and John D. Lafferty. 2009. "Visualizing Topics with Multi-Word Expressions." Tech. Report. (arXiv:0907.1013).

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. "Latent Dirichlet Allocation." Journal of Machine Learning Research 993-1022.

Samuel Brody and Noemie Elhadad. 2010. "An Unsupervised Aspect-Sentiment Model for Online Reviews." NAACL. Los Angeles, California: ACL.

Jonathan Chang, Jordan Boyd-Graber, Sean Gerrish, Chong Wang, and David M. Blei. 2009. "Reading Tea Leaves: How Humans Interpret Topic Models." Neural Information Processing Systems.

Zhiyuan Chen, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. "Exploiting Domain Knowledge in Aspect Extraction." EMNLP.

Thomas L. Griffiths, and Mark Steyvers. 2004. "Finding scientific topics." Proceedings of National Academy of Sciences.

Gregor Heinrich. 2009. "A Generic Approach to Topic Models." ECML PKDD. ACM. Pages 517 - 532.

Thomas Hofmann. 1999. "Probabilistic latent semantic analysis." UAI.

Yohan Jo and Alice Oh. 2011. "Aspect and Sentiment Unification Model for Online Review Analysis." WSDM. Hong Kong, China: ACM.

Chenghua Lin and Yulan He. 2009. "Joint Sentiment/Topic Model for Sentiment Analysis". CIKM. Hong Kong, China.

Fangtao Li, Minlie Huang, Xiaoyan Zhu. 2010. "Sentiment Analysis with Global Topics and Local Dependency". AAAI

Yue Lu and Chengxiang Zhai. 2008. "Opinion Integration Through Semi-supervised Topic Modeling." WWW. 2008, Beijing, China: ACM.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. "Automatic Keyphrase Extraction via Topic Decomposition." EMNLP.

Arjun Mukherjee and Bing Liu. 2013. "Discovering User Interactions in Ideological Discussions." ACL.

Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Sharon Meraz. 2013. "Public Dialogue: Analysis of Tolerance in Online Discussions." ACL.

David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. "Optimizing Semantic Coherence in Topic Models." EMNLP. Edinburgh, Scotland, UK: ACL.

Hosan Mahmoud. 2008. Polya Urn Models. Chapman & Hall/CRC Texts in Statistical Science.

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. "Topic Sentiment Mixture: Modeling Facets and Opinions in Weblogs." WWW. Banff, Alberta, Canada: ACM.

Christina Sauper and Regina Barzilay. 2013. "Automatic Aggregation by Joint Modeling of Aspects and Values". Journal of Artificial Intelligence Research 46 (2013) 89-127

Ivan Titov and Ryan McDonald. 2008. "Modeling Online Reviews with Multi-grain Topic Models." WWW. 2008, Beijing, China: ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2005. "Hierarchical Dirichlet Processes." Journal of the American Statistical Association.

Hanna M. Wallach. 2006. "Topic Modeling: Beyond Bag-of-Words." ICML. Pittsburgh, PA: ACM.

Xuerui Wang, Andrew McCallum, and Xing Wei. 2007. "Topical N-grams: Phrase and Topic Discovery, with an Application to Information Retrieval." ICDM.

Wayne Xin Zhao, Jing Jiang, Hongfei Yan, and Xiaoming Li. 2010. "Jointly Modeling Aspects and Opinions with a MaxEnt-LDA Hybrid." EMNLP. Massachusetts, USA: ACL.

Wayne Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li. 2011. "Topical Keyphrase Extraction from Twitter." ACL.

Joint Opinion Relation Detection Using One-Class Deep Neural Network

Liheng Xu, Kang Liu and Jun Zhao
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China
{lhxu, kliu, jzhao}@nlpr.ia.ac.cn

Abstract

Detecting opinion relation is a crucial step for fine-gained opinion summarization. A valid opinion relation has three requirements: a correct opinion word, a correct opinion target and the linking relation between them. Previous works prone to only verifying two of these requirements for opinion extraction, while leave the other requirement unverified. This could inevitably introduce noise terms. To tackle this problem, this paper proposes a joint approach, where all three requirements are simultaneously verified by a deep neural network in a classification scenario. Some seeds are provided as positive labeled data for the classifier. However, negative labeled data are hard to acquire for this task. We consequently introduce one-class classification problem and develop a One-Class Deep Neural Network. Experimental results show that the proposed joint approach significantly outperforms state-of-the-art weakly supervised methods.

1 Introduction

Opinion summarization aims to extract and summarize customers' opinions from reviews on products or services (Hu and Liu, 2004; Cardie et al., 2004). With the rapid expansion of e-commerce, the number of online reviews is growing at a high speed, which makes it impractical for customers to read throughout large amounts of reviews to choose better products. Therefore, it is imperative to automatically generate opinion summarization to help customers make more informed purchase decisions, where detecting opinion relation is a crucial step for opinion summarization.

Before going further, we first introduce some notions. An *opinion relation*, is a triple $o = (s, t, r)$, where three factors are involved: s is an *opinion word* which refers to those words indicating sentiment polarities; t is an *opinion target*, which can be any entity or aspect of an entity about which an opinion has been expressed; r refers to the linking relation between s and t. As in Example 1, s=$\{$*clear*$\}$, t=$\{$*sceen*$\}$, and there is a linking relation between the two words because *clear* is used to modify *screen*.

Example 1. *This mp3 has a clear screen.*

For a valid opinion relation, there are three requirements corresponding to the three factors: (i) the opinion word indicates sentiment polarity; (ii) the opinion target is related to current domain; (iii) the opinion word modifies the opinion target. Previous weakly supervised methods often expand a seed set and identify opinion relation either by co-occurrence statistics (Hu and Liu, 2004; Hai et al., 2012) or syntactic dependencies (Popescu and Etzioni, 2005; Qiu et al., 2009) following the assumption below.

Assumption 1. Terms that are likely to have linking relation with the seed terms are believed to be opinion words or opinion targets.

For example, if one has an opinion word seed *clear* (which satisfies requirement i), and one finds that it modifies the word *screen* in Example 1 (which satisfies requirement iii). Then one infers that *screen* is an opinion target according to Assumption 1 (whether *screen* is correct is not checked). However, in Example 2(a), we can see that *good* is an opinion word and it modifies *thing*, but *thing* is not related to

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 677–687, Dublin, Ireland, August 23-29 2014.

mp3 domain. If one follows Assumption 1, *thing* will be mistaken as an opinion target. Similarly, in Example 2(b), if one uses *mp3* to extract *another* as an opinion word, he may get an objective word.

Example 2. (a) *This mp3 has many good things.* (b) *Just another mp3 I bought.*

The reason for the errors above is that Assumption 1 only verifies two requirements for an opinion relation. Unfortunately, this issue occurs frequently in online reviews. As a result, previous methods often suffer from these noise terms. To produce more precise opinion summary, it is argued that we shall follow a more restricted assumption as follows.

Assumption 2. The three requirements: the opinion word, the opinion target and the linking relation between them, shall be all verified during opinion relation detection.

To make accordance with Assumption 2, this paper proposes a novel joint opinion relation detection method, where opinion words, opinion targets and linking relations are simultaneously considered in a classification scenario. Following previous works, we provide a small set of seeds (i.e. opinion words or targets) for supervision, which are regarded as positive labeled examples for classification. However, negative labeled examples (i.e. noise terms) are hard to acquire, because we do not know which term is not an opinion word or target. This leads to One-Class Classification (OCC) problem (Moya et al., 1993). The key to OCC is semantic similarity measuring between terms, and Deep Neural Network (DNN) with word embeddings is a powerful tool for handling this problem. We consequently integrate DNN into a OCC classifier and develop a *One-Class Deep Neural Network* (OCDNN). Concretely, opinion words/targets/relations are first represented by embedding vectors and then *jointly* classified. Experimental results show that the proposed joint method which follows Assumption 2 significantly outperforms state-of-the-art weakly supervised methods which are based on Assumption 1.

2 Related Work

In opinion relation detection task, previous works often used co-occurrence statistics or syntax information to identify opinion relations. For co-occurrence statistical methods, Hu and Liu (2004) proposed a pioneer research for opinion summarization based on association rules. Popescu and Etzioni (2005) defined some syntactic patterns and used Pointwise Mutual Information (PMI) to extract product features. Hai et al. (2012) proposed an opinion feature mining method which employed Likelihood Ratio Tests (LRT) (Dunning, 1993) as the co-occurrence statistical measure. For syntax-based approaches, Riloff and Wiebe (2003) performed syntactic pattern learning while extracting subjective expressions. Zhuang et al. (2006) used various syntactic templates from an annotated movie corpus and applied them to supervised movie feature extraction. Kobayashi et al. (2007) identified opinion relations by searching for useful syntactic contextual clues. Qiu et al. (2009) proposed a bootstrapping framework called *Double Propagation* which introduced eight heuristic syntactic rules to detect opinion relations.

However, none of the above methods could verify opinion words/targets/relations simultaneously during opinion relation detection. To perform joint extraction, various models had been proposed, most of which employed classification or sequence labeling models, such as HMM (Jin and Ho, 2009), SVM (Wu et al., 2009) and CRFs (Breck et al., 2007; Jakob and Gurevych, 2010; Li et al., 2010). Besides, optimal models such as Integer Linear Programming (ILP) were also employed to perform joint inference for opinion extraction (Choi et al., 2006; Yang and Cardie, 2013).

Joint methods had been shown to achieve better performance than pipeline approaches. Nevertheless, most existing joint models rely on full supervision, which have the difficulty of obtaining annotated training data in practical applications. Also, supervised models that are trained on one domain often fail to give satisfactory results when shifted to another domain. Our method does not require annotated data.

3 The Proposed Method

To detect opinion relations, previous methods often leverage some seed terms, such as opinion word seeds (Hu and Liu, 2004; Baccianella et al., 2010) and opinion target seeds (Jijkoun et al., 2010; Hai et al., 2012). These seeds can be used as positive labeled examples to train a classifier. However, it is hard to get negative labeled examples for this task. Because opinion words or targets are often domain

dependent and words that do not bear any sentiment polarity in one domain may be used to express opinion in another domain. It is also very hard to specify in what case there is no linking relation between two words.

To deal with this problem, we employ one-class classification, and develop a One-Class Deep Neural Network (OCDNN) for opinion relation detection. The architecture of OCDNN is shown in Figure 1, which consists of two levels. The lower level learns feature representations unsupervisedly for opinion words/targets/relations, where the left component uses word embedding learning to represent opinion words/targets, and the right component maps linking relations to embedding vectors by a recursive autoencoder. Then the upper level uses the learnt features to perform one-class classification.

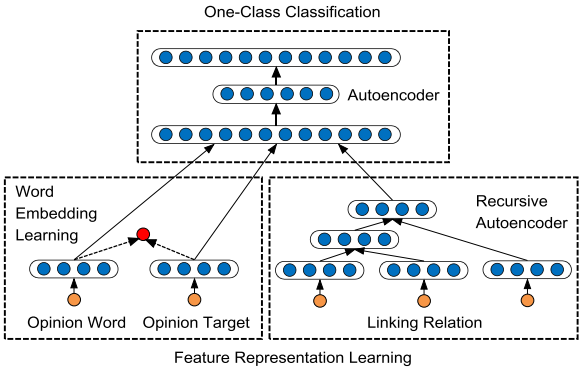


Figure 1: The architecture of OCDNN.

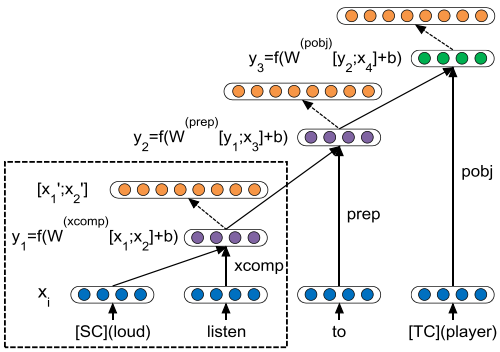


Figure 2: An example of recursive autoencoder.

3.1 Opinion Seed Generation

To obtain training data for OCDNN, we shall first get some seed terms as follows.

Opinion Word Seeds. We manually pick 186 domain independent opinion words from SentiWordNet (Baccianella et al., 2010) as the opinion word seed set SS.

Opinion Target Seeds. Likelihood Ratio Tests (LRT) (Dunning, 1993) used in (Hai et al., 2012) is employed to generate opinion target seeds. LRT aims to measure how greatly two terms T_i and T_j are associated with each other by sentence-level corpus statistics which is defined as follows,

$$LRT = 2[\log L(p_1, k_1, n_1) + \log L(p_2, k_2, n_2) - \log L(p, k_1, n_1) - \log L(p, k_2, n_2)] \quad (1)$$

where $k_1 = tf(T_i, T_j), k_2 = tf(T_i, \bar{T}_j), k_3 = tf(\bar{T}_i, T_j), k_4 = tf(\bar{T}_i, \bar{T}_j), tf(\cdot)$ denotes term frequency; $L(p, k, n) = p^k(1-p)^{n-k}$, $n_1 = k_1 + k_3$, $n_2 = k_2 + k_4$, $p_1 = k_1/n_1$, $p_2 = k_2/n_2$ and $p = (k_1 + k_2)/(n_1 + n_2)$. We measure LRT between a domain name (e.g. *mp3*, *hotel*, etc.) and all opinion target candidates. Then N terms with highest LRT scores are added into the opinion target seed set TS.

Linking Relation Seeds. Linking relation can be naturally captured by syntactic dependency, because it directly models the modification relation between opinion word and opinion target. We employ an automatic syntactic opinion pattern learning method called *Sentiment Graph Walking* (Xu et al., 2013) and get 12 opinion patterns with highest confidence as the linking relation seed set RS.

After seed generation, every opinion relation $s_o = (s_s, s_t, s_r)$ in review corpus that satisfies $s_s \in SS$, $s_t \in TS$ and $s_r \in RS$ is taken as a positive labeled training instance.

3.2 Opinion Relation Candidate Generation

The opinion term candidate set is denoted by $C = \{SC, TC\}$, where SC/TC represents opinion word/target candidate. Following previous works (Hu and Liu, 2004; Popescu and Etzioni, 2005; Qiu et al., 2009), we take adjectives or verbs as opinion word candidates, and take nouns or noun phrases as opinion target candidates. A statistic-based method in Zhu et al. (2009) is used to detect noun phrases.

An opinion relation candidate is denoted by $c_o = (c_s, c_t, c_r)$, where $c_s \in SC$, $c_t \in TC$, and c_r is a potential linking relation. To get c_r, we first get dependency tree of a sentence using Stanford Parser (de

Marneffe et al., 2006). Then, the shortest dependency path between a c_s and a c_t is taken as a c_r. To avoid introducing too many noise candidates, we constrain that there are at most four terms in a c_r.

3.3 Word Representation by Word Embedding Learning

Word embedding, a.k.a *word representation*, is a mathematical object associated with each word, which is often used in a vector form, where each dimension's value corresponds to a feature and might even have a semantic or grammatical interpretation (Turian et al., 2010). By word embedding learning, words are embedded into a hyperspace, where two words that are more semantically similar to each other are located closer. This characteristic is precisely what we want, because the key to one-class classification is semantic similarity measuring (illustrated in Section 3.5).

For word representation, we use a matrix $LT \in \mathbb{R}^{n \times |V_w|}$, where *i*-th column represents the embedding vector for term t_i, n is the size of embedding vector and V_w is the vocabulary of LT. Therefore, we can denote t_i by a binary vector $b_i \in \mathbb{R}^{|V_w|}$ and get its embedding vector by $x_i = LTb_i$. The training criterion for word embeddings is,

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \sum_{c \in C} \sum_{v \in V_w} max\{0, 1 - s_\theta(c) + s_\theta(v)\} \tag{2}$$

where θ is the parameters of neural network used for training. See Collobert et al. (2011) for the detailed implementation.

3.4 Linking Relation Representation by Using Recursive Autoencoder

The goal of this section is to represent the linking relation between an opinion word and an opinion target by a *n*-element vector as we do during word representation. Specifically, we combine embedding vectors of words in a linking relation by a recursive autoencoder (Socher et al., 2011) according to syntactic dependency structure. In this way, linking relations are no longer limited to the initial seeds during classification, because linking relations that are similar to the seed relations will have similar vector representations.

Figure 2 shows a linking relation representation process by an example: *too loud to listen to the player*. First, we get its dependency path between the opinion word c_s*:loud* and the opinion target c_t*:player*. Then c_s and c_t are replaced by wildcards *[SC]* and *[TC]* because they are not concerned in the linking relation. The dash line box in Figure 2 shows a standard autoencoder, which is a three-layer neural network, where the number of nodes in input layer is equal to that of output layer. It takes two *n*-element vectors as input and compresses semantics of the two vectors into one *n*-element vector in hidden layer by,

$$y = f(W^{(dep)}[x_1; x_2] + b),\ W^{(dep)} = \frac{1}{2}[I_1; I_2; I_b] + \epsilon \tag{3}$$

where $[x_1; x_2]$ is the concatenation of the two input vectors and f is the sigmoid function; $W^{(dep)}$ is a parameter matrix that is chosen according to the dependency relation between x_1 and x_2 (In the case of y_1, $W^{(dep)} = W^{(xcomp)}$), which is initialized by I_i, where I_i is a $n \times n$ unit matrix, I_b is a n-element null vector, and ϵ is sampled from a uniform distribution $U[-0.001, 0.001]$ (Socher et al., 2013). Then $W^{(dep)}$ are updated during training. The training criterion of autoencoder is to minimize Euclidean distance between the original input and its output,

$$E_{rae} = ||[x_1; x_2] - [x_1'; x_2']||^2 \tag{4}$$

where $[x_1'; x_2'] = W^{(out)}y$ and $W^{(out)}$ is initialized by ${W^{(dep)}}^T$.

We always start the combination process from *[SC]* and it is repeated along the dependency path. For example, the result vector y_1 of the first combination is used as the input vector when computing y_2. Finally, the linking relation is represented by a n-element vector (the green vector in Figure 2).

3.5 One-Class Classification for Opinion Relation Detection

We represent an opinion relation candidate $c_o = (c_s, c_t, c_r)$ by a vector $v_o = [v_s; v_t; v_r]$, which is a concatenation of the opinion word embedding v_s, the opinion target embedding v_t and the linking relation embedding v_r. Then v_o is feed to the upper level autoencoder in Figure 1.

To perform one-class classification, the number of nodes in the hidden layer of the upper level autoencoder is constrained to be smaller than that of the input layer. By using such a "bottleneck" network structure, characteristics of the input are first compressed into the hidden layer and then reconstructed by the output layer (Japkowicz et al., 1995). Concretely, characteristics of positive labeled opinion relations are first compressed into the hidden layer, and then the autoencoder should be able to adequately reconstruct positive instances in the output layer, but should fail to reconstruct negative instances which present different characteristics from positive instances. Therefore, the detection of opinion relation is equivalent to assessing how well a candidate is reconstructed by the autoencoder. As the input vector v_o consists of representations for opinion words/targets/relations, characteristics of the three factors are *jointly* compressed by one hidden layer. Either false opinion word/target/relation will lead to failure of reconstruction. Consequently, our approach follows Assumption 2.

For opinion relation detection, candidates with reconstruction error scores that are smaller than a threshold ϑ are classified as positive. Determining the exact value of ϑ is very difficult. Inspired by other one-class approaches (Liu et al., 2002; Manevitz and Yousef, 2007), we introduce some negative opinion terms to help to estimate ϑ.[1] Although negative instances are hard to acquire, Xu et al. (2013) show that a set of general nouns (such as *thing*, *one*, etc., we denote them by GN) seldom appear to be opinion targets. One the other hand, we create a 50-opinion-word validation set SV from SentiWordNet.

To estimate ϑ, we first introduce a *positive proportion* (pp) score,

$$pp(t) = tf^{+}(t)/tf(t),\ t \in PE,\ PE = \{c_o | E_r(c_o) < \vartheta\} \tag{5}$$

where PE denotes the opinion relations that are classified as positive, $E_r(\cdot)$ is the reconstruction error of OCDNN and $tf^{+}(\cdot)$ is the frequency of term in PE. Then an error function E_ϑ is minimized, which balances between the proportion of non-target terms (GN) in PE (which shall be as small as possible) and the proportion of opinion words in validation set (SV) in PE (which shall be as large as possible).

$$E_\vartheta = \sum_{t \in GN \cap PE} [pp(t) - 0]^2 + \sum_{s \in SV \cap PE} [pp(s) - 1]^2 \tag{6}$$

3.6 Opinion Target Expansion

We apply bootstrapping to iteratively expand opinion target seeds. It is because the vocabulary of seed set is limited, which cannot fully represent the distribution of opinion targets. So we expand opinion target seeds in a self-training manner to alleviate this issue. After training OCDNN, all opinion relation candidates are classified, and opinion targets are ranked in descent order by,

$$s(t) = \log tf(t) \times pp(t). \tag{7}$$

Then, top M candidates are added into the target seed set TS for the next training iteration.

4 Experiments

4.1 Datasets and Evaluation Metrics

Datasets. Three real world datasets are selected for evaluation. The first one is called *Customer Review Dataset (CRD)* [2] which contains reviews on five products (denoted by D1 to D5). The second is a benchmark dataset (Wang et al., 2011) on *MP3* and *Hotel*[3]. The last one is crawled from www.amazon.com, which involves *Mattress* and *Phone*. Two annotating criteria are applied.

[1] This is not in contradiction with OCC problem, because these negative examples are NOT used during training.

[2] http://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html

[3] http://timan.cs.uiuc.edu/downloads.html

Annotation 1 is used to evaluate opinion words/targets extraction. Firstly, 10,000 sentences are randomly selected from reviews and all possible terms are extracted along with their contexts. Then, annotators are required to judge whether each term is an opinion word or an opinion target.

Annotation 2 is used to evaluate intra-sentence opinion relation detection. Annotators are required to carefully read through each sentence and find out every opinion relation, which consists of an opinion word, an opinion target, as well as the linking relation between them. The annotation is very labor-intensive, so only 5,000 sentences are annotated for *MP3* and *Hotel*.

Two annotators were required to annotate following the criteria above. When conflicts happened, a third annotator would make the final judgment. Note that Annotation 1 and Annotation 2 were annotated by two different groups. Detailed information of the annotated datasets are shown in Table 1. Furthermore, the kappa values between Annotation 1 and Annotation 2 are 0.88 for opinion words and 0.84 for opinion targets, showing highly substantial agreement.

Domain	#OW	#OT	Kappa_OW	Kappa_OT
Hotel	434	1,015	0.72	0.67
MP3	559	1,158	0.69	0.65
Mattress	366	523	0.67	0.62
Phone	391	862	0.68	0.64

(a) Annotation 1

Domain	#LR	#OW	#OT	Kappa_LR
Hotel	2,196	317	735	0.62
MP3	2,328	342	791	0.61

(b) Annotation 2

Table 1: The detailed information of Annotations. OW/OT/LR stands for opinion words/opinion targets/linking relations. The Kappa-values are calculated by using *exact* matching metric for Annotation 1 and *overlap* matching metric for Annotation 2.

Evaluation Metrics. We perform evaluation in terms of Precision(P), Recall(R) and F-measure(F) according to *exact* and *overlap* matching metrics (Wiebe et al., 2005). The *exact* metric is used to evaluate opinion word/target extraction, which requires exact string match. And the *overlap* metric is used to evaluate opinion relation detection, where an extracted opinion relation is regarded as correct when both the opinion word and the opinion target in it overlap with the gold standard.[4]

Evaluation Settings. Four state-of-the-art weakly supervised approaches are selected as competitors. Two are co-occurrence statistical methods and two are syntax-based methods, all of which follow Assumption 1.

AdjRule extracts opinion words/targets by using adjacency rules (Hu and Liu, 2004).

LRTBOOT is a bootstrapping algorithm which employs *Likelihood Ratio Tests* (Dunning, 1993) as the co-occurrence statistical measure (Hai et al., 2012).

DP denotes the *Double Propagation* algorithm (Qiu et al., 2009).

DP-HITS is an enhanced version of *DP* proposed by Zhang et al. (2010), which ranks terms by

$$s(t) = \log tf(t) \times importance(t) \tag{8}$$

where $importance(t)$ is estimated by the HITS algorithm (Kleinberg, 1999).

OCDNN is the proposed method. The target seed size $N = 40$, the opinion targets expanded in each iteration $M = 20$, and the max bootstrapping iteration number is $X = 10$. The representation learning in lower level of OCDNN is trained on the whole corpus, while the test data are the same for all settings. All results of OCDNN are taken by average performance over five runs with randomized parameters.

4.2 OCDNN vs. the State-of-the-art

We compare *OCDNN* with state-of-the-art methods for opinion words/targets extraction. In *OCDNN*, Eq. 7 is used to rank opinion words/targets. The results on *CRD* and the four domains are shown in Table 2 and Table 3. *DP-HITS* does not extract opinion words so their results for opinion words are not taken into account.

[4]Determining the exact boundaries of opinion terms is hard even for human (Wiebe et al., 2005), so we use this relaxation.

Method	D1			D2			D3			D4			D5			Avg.
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	F
Opinion Targets																
AdjRule	0.75	0.82	0.78	0.71	0.79	0.75	0.72	0.76	0.74	0.69	0.82	0.75	0.74	0.80	0.77	0.76
DP	**0.87**	0.81	**0.84**	**0.90**	0.81	**0.85**	**0.90**	0.86	**0.88**	**0.81**	0.84	**0.82**	**0.92**	0.86	**0.89**	**0.86**
DP-HITS	0.83	0.84	0.83	0.86	0.85	**0.85**	0.86	0.88	0.87	0.80	0.85	**0.82**	0.86	0.86	0.86	0.85
LRTBOOT	0.77	**0.87**	0.82	0.74	**0.90**	0.81	0.79	**0.89**	0.84	0.72	**0.88**	0.79	0.74	**0.88**	0.80	0.81
OCDNN	0.83	0.82	0.82	0.86	0.85	**0.85**	0.86	0.87	0.86	0.78	0.84	0.81	0.89	0.85	0.87	0.84
Opinion Words																
AdjRule	0.57	0.75	0.65	0.51	0.76	0.61	0.57	0.73	0.64	0.54	0.62	0.58	0.62	0.67	0.64	0.62
DP	**0.64**	0.73	0.68	0.57	0.79	0.66	0.65	0.70	0.67	0.61	0.65	0.63	**0.70**	0.68	**0.69**	0.67
LRTBOOT	0.60	**0.79**	0.68	0.52	**0.82**	0.64	0.60	**0.76**	0.67	0.56	**0.70**	0.62	0.66	**0.71**	0.68	0.66
OCDNN	**0.64**	0.77	**0.70**	**0.63**	0.79	**0.70**	**0.66**	0.73	**0.69**	**0.68**	**0.70**	**0.69**	**0.70**	0.69	**0.69**	**0.70**

Table 2: Results of opinion terms extraction on *Customer Review Dataset*.

Method	MP3			Hotel			Mattress			Phone			Avg.
	P	R	F	P	R	F	P	R	F	P	R	F	F
Opinion Targets													
AdjRule	0.53	0.55	0.54	0.55	0.57	0.56	0.50	0.60	0.55	0.52	0.51	0.51	0.54
DP	0.66	0.57	0.61	0.66	0.60	0.63	0.55	0.60	0.57	0.60	0.53	0.56	0.59
DP-HITS	0.65	0.62	0.63	0.64	0.66	0.65	0.55	0.67	0.60	0.62	0.64	0.63	0.63
LRTBOOT	0.60	**0.77**	0.67	0.59	**0.78**	0.67	0.55	**0.78**	0.65	0.57	**0.76**	0.65	0.66
OCDNN	**0.70**	0.68	**0.69**	**0.71**	0.70	**0.70**	**0.63**	0.69	**0.66**	**0.69**	0.68	**0.68**	**0.68**
Opinion Words													
AdjRule	0.48	0.65	0.55	0.51	0.68	0.58	0.51	0.68	0.58	0.48	0.61	0.54	0.56
DP	0.58	0.62	0.60	0.60	0.66	0.63	0.54	0.68	0.60	0.55	0.59	0.57	0.60
LRTBOOT	0.52	**0.69**	0.59	0.54	**0.74**	0.62	0.51	**0.73**	0.60	0.50	**0.68**	0.58	0.60
OCDNN	**0.68**	0.65	**0.66**	**0.70**	0.68	**0.69**	**0.59**	0.70	**0.64**	**0.63**	0.59	**0.61**	**0.65**

Table 3: Results of opinion terms extraction on the four domains.

From Table 2, we can see that our method outperforms co-occurrence-based methods *AdjRule* and *LRTBOOT*, but achieves comparable or a little worse results than syntax-based methods *DP* and *DP-HITS*. This is because *CRD* is quite small, which only contains several hundred sentences for each product review set. In this case, methods based on careful-designed syntax rules have superiority over those based on statistics (Liu et al., 2013). For results on larger datasets shown in Table 3, our method outperforms all of the competitors. Comparing *OCDNN* with *DP-HITS*, the two approaches use similar term ranking metrics (Eq. 7 and Eq. 8), but *OCDNN* significantly outperforms *DP-HITS*. Therefore, the *positive proportion* score estimated by *OCDNN* is more effective than the *importance* score in *DP-HITS*. Comparing *OCDNN* with *LRTBOOT*, we find that *LRTBOOT* achieves better recall but lower precision. This is because *LRTBOOT* follows Assumption 1 during bootstrapping, which suffers a lot from error propagation, while our joint classification approach effectively alleviates this issue. We will discuss the impact of error propagation in detail later.

4.3 Assumption 1 vs. Assumption 2

This section evaluates intra-sentence opinion relation detection, which is more useful for practical applications. It also reflects the impacts of Assumption 1 and Assumption 2. The results are shown in Table 4 and Table 5, where *OCDNN* significantly outperforms all competitors. The average improvement of F-measure over the best competitor is 6% on *CRD* and 9% on *Hotel* and *MP3*.

As Assumption 1 only verifies two of the requirements in an opinion relation, it would inevitably introduce noise terms during extraction. For syntax-based method *DP*, it extracts many false opinion relations such as *good thing* and *nice one* (where *thing* and *one* are false opinion targets) or objective expressions like *another mp3* and *every mp3* (which contain false opinion words *another* and *every*). For co-occurrence statistical methods *AdjRule* and *LRTBOOT*, it is very hard to deal with ambiguous linking relations. For example, in phrase *this mp3 is very good except the size*, co-occurrence statistical methods could hardly tell which opinion target does *good* modify (*mp3* or *size*). Our method follows Assumption

Method	D1			D2			D3			D4			D5			Avg.
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	F
AdjRule	0.51	0.66	0.58	0.53	0.63	0.58	0.50	0.61	0.55	0.48	0.60	0.53	0.50	0.61	0.55	0.56
DP	0.66	0.63	0.64	0.68	0.60	0.64	0.69	0.62	0.65	0.66	0.57	0.61	0.67	0.60	0.63	0.64
LRTBOOT	0.53	**0.70**	0.60	0.57	**0.72**	0.64	0.55	**0.69**	0.61	0.52	**0.70**	0.60	0.55	**0.68**	0.61	0.61
OCDNN	**0.76**	0.66	**0.71**	**0.74**	0.67	**0.70**	**0.77**	0.67	**0.72**	**0.70**	0.65	**0.67**	**0.77**	0.66	**0.71**	**0.70**

Table 4: Results of opinion relation detection on *Customer Review Dataset*.

Method	MP3			Hotel			Avg.
	P	R	F	P	R	F	F
AdjRule	0.49	0.55	0.52	0.45	0.53	0.49	0.50
DP	0.63	0.51	0.56	0.59	0.50	0.54	0.55
LRTBOOT	0.54	**0.63**	0.58	0.50	**0.60**	0.55	0.56
OCDNN	**0.73**	0.60	**0.66**	**0.70**	0.59	**0.64**	**0.65**

Table 5: Results of opinion relation detection on the two domains.

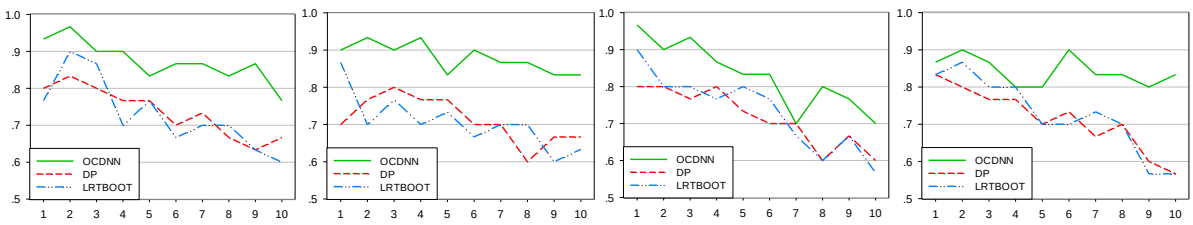
2, which verifies all three requirements for opinion word/target/relation in an opinion relation, so the above errors are greatly reduced. Therefore, Assumption 2 is more reasonable than Assumption 1.

4.4 The Effect of Joint Classification

We evaluate the three bootstrapping methods (*DP*, *LRTBOOT* and *OCDNN*) for opinion target expansion. The precision of each iteration is shown in Figure 3. We can see that *DP* and *LRTBOOT* gradually suffer from error propagation and the precision drops quickly along with the number of iteration increases. For *OCDNN*, although error propagation is inevitable, the precision curve retains at a high level. Therefore, the joint approach produces more precise results.

For more detailed analysis, we give a variation of the proposed method named ***3NN***, which uses 3 individual autoencoders to classify opinion words/targets/relations separately. An opinion relation candidate is classified as positive only when the three factors are all classified as positive. Then opinion relations are ranked by the sum of reconstruction scores of the three factors. In the results of opinion relation detection, when the recall is fixed at 0.6, the precisions of *3NN* are 0.67 for *MP3* and 0.65 for *Hotel*, while the precisions of *OCDNN* are 0.73 for *MP3* and 0.70 for *Hotel*. Therefore, *OCDNN* achieves much better performance than *3NN*.

An example may explain the reason of why *3NN* gets worse performance. In our experiment on *Hotel*, a false opinion relation *happy day* is misclassified as positive by *3NN*. It is because the word *day* has a small reconstruction score in *3NN*. At the same time, *happy* is a correct opinion word, so the whole expression *happy day* also has a small reconstruction score and then be misclassified. In contrast, the reconstruction score of *happy day* from *OCDNN* is quite large so the phrase is dropped. The reason is that the joint approach captures the semantic of a whole phrase rather than its single components. Therefore, it is more reasonable.



(a) MP3 (b) Hotel (c) Mattress (d) Phone

Figure 3: Precision (y-axis) of opinion target seed expansion at each bootstrapping iteration (x-axis).

5 Conclusion and Future Work

This paper proposes One-Class Deep Neural Network for joint opinion relation detection in one-class classification scenario, where opinion words/targets/relations are simultaneously verified during classification. Experimental results show the proposed method significantly outperforms state-of-the-art weakly supervised methods that only verify two factors in an opinion relation.

In future work, we plan to adapt our method and make it be capable of capturing implicit opinion relations.

Acknowledgement

This work was sponsored by the National Natural Science Foundation of China (No. 61202329 and No. 61333018) and CCF-Tencent Open Research Fund.

References

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. *Seventh conference on International Language Resources and Evaluation*, pages 2200–2204.

Eric Breck, Yejin Choi, and Claire Cardie. 2007. Identifying expressions of opinion in context. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 2683–2688, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Claire Cardie, Janyce Wiebe, Theresa Wilson, and Diane Litman. 2004. Low-level annotations and summary representations of opinions for multi-perspective question answering.

Yejin Choi, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 431–439, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, November.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the IEEE / ACL 2006 Workshop on Spoken Language Technology*. The Stanford Natural Language Processing Group.

Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19(1):61–74, March.

Zhen Hai, Kuiyu Chang, and Gao Cong. 2012. One seed to find them all: mining opinion features via association. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, CIKM '12, pages 255–264, New York, NY, USA. ACM.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Niklas Jakob and Iryna Gurevych. 2010. Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1035–1045, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nathalie Japkowicz, Catherine Myers, and Mark Gluck. 1995. A novelty detection approach to classification. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, IJCAI'95, pages 518–523, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Valentin Jijkoun, Maarten de Rijke, and Wouter Weerkamp. 2010. Generating focused topic-specific sentiment lexicons. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 585–594, Stroudsburg, PA, USA. Association for Computational Linguistics.

Wei Jin and Hung Hay Ho. 2009. A novel lexicalized hmm-based learning framework for web opinion mining. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 465–472.

Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September.

Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1065–1074, Prague, Czech Republic, June. Association for Computational Linguistics.

Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 653–661, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. 2002. Partially supervised classification of text documents. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pages 387–394, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kang Liu, Liheng Xu, and Jun Zhao. 2013. Syntactic patterns versus word alignment: Extracting opinion targets from online reviews. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1754–1763, August.

Larry Manevitz and Malik Yousef. 2007. One-class document classification via neural networks. *Neurocomputing*, 70(7C9):1466–1481.

Mary M. Moya, Mark W. Koch, and Larry D. Hostetler. 1993. One-class classifier networks for target recognition applications. In *Proceedings world congress on neural networks*, pages 797–801.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 339–346.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st international jont conference on Artifical intelligence*, IJCAI'09, pages 1199–1204.

Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, EMNLP '03, pages 105–112, Stroudsburg, PA, USA. Association for Computational Linguistics.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 151–161, Stroudsburg, PA, USA. Association for Computational Linguistics.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria, August. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 618–626.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

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

Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen, and Jun Zhao. 2013. Mining opinion words and opinion targets in a two-stage framework. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1764–1773, Sofia, Bulgaria, August. Association for Computational Linguistics.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1640–1649, Sofia, Bulgaria, August. Association for Computational Linguistics.

Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 1462–1470, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

A Generative Model for Identifying Target Companies of Microblogs

Yeyun Gong, Yaqian Zhou, Ya Guo, Qi Zhang, Xuanjing Huang
Shanghai Key Laboratory of Intelligent Information Processing
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, P.R.China
{12110240006, zhouyaqian, 13210240002, qz, xjhuang}@fudan.edu.cn

Abstract

Microblogging services have attracted hundreds of millions of users to publish their status, ideas and thoughts, everyday. These microblog posts have also become one of the most attractive and valuable resources for applications in different areas. The task of identifying the main targets of microblogs is an important and essential step for these applications. In this paper, to achieve this task, we propose a novel method which converts the target company identification problem to the translation process from content to targets. We introduce a topic-specific generative method to model the translation process. Topic specific trigger words are used to bridge the vocabulary gap between the words in microblogs and targets. We examine the effectiveness of our approach via datasets gathered from real world microblogs. Experimental results demonstrate a 20.2% improvement in terms of F1-score over the state-of-the-art discriminative method.

1 Introduction

With the rapid growth of social media, about 72% of adult internet users are also members of a social networking site[1]. Over the past few years, microblogging has become one of the most popular services. Meanwhile, microblogs have also been widely used as sources for analyzing public opinions (Bermingham and Smeaton, 2010; Jiang et al., 2011), prediction (Asur and Huberman, 2010; Bollen et al., 2011), reputation management (Pang and Lee, 2008; Otsuka et al., 2012), and many other applications (Bian et al., 2008; Sakaki et al., 2010; Becker et al., 2010; Guy et al., 2010; Lee and Croft, 2013; Guy et al., 2013). For most of these applications, identifying the microblogs that are relevant to the targets of interest is one of the basic steps (Lin and He, 2009; Amigó et al., 2010; Qiu et al., 2011; Liu et al., 2013). Let us firstly consider the following example:

Example 1: *11" MacBook Air can run for up to five hours on a single charge.*

"*MacBook Air*" can be considered to be the target being discussed on the microblog, and we can also infer from the microblog that it is related to Apple Inc. The ability to discriminate which company is being referred to in a microblog is required by many applications.

Previous studies on fine-grained sentiment analysis and aspect-based opinion mining proposed supervised (Popescu and Etzioni, 2005; Liu et al., 2012a; Liu et al., 2013) and unsupervised methods (Hu and Liu, 2004; Wu et al., 2009; Zhang et al., 2010) to extract targets of opinion expressions. Based on the associations between opinion targets and opinion words, some methods were also introduced to simultaneously solve the opinion expression and target extraction problems (Qiu et al., 2011; Liu et al., 2012a). However, most of the existing methods in this area only focus on extracting items about which opinions are expressed in a given domain. The implicated information of targets is rarely considered. Moreover, domain adaptation is another big challenge for these fine-grained methods in processing different domains.

[1]It is reported by the Pew Research Center's Internet & American Life Project in Aug 5, 2013.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 688–697, Dublin, Ireland, August 23-29 2014.

The WePS-3[2] (Amigó et al., 2010) and RepLab 2013[3] (Amigó et al., 2013) evaluation campaigns also addressed the problem from the perspective of the disambiguation of company names in microblogs. Microblogs that contain company names at a lexical level are classified based on whether it refers to the company or not. Various approaches have been proposed to address the task with different methods (Pedersen et al., 2006; Yerva et al., 2010; Zhang et al., 2012; Spina et al., 2012; Spina et al., 2013). However, the microblogs that do not contain company names cannot be correctly processed using these methods. From analyzing the data, we observe that a variety of microblog posts belong to this type. They only contain products names, slang terms, and other related company content.

To achieve this task, in this paper, we propose the use of a translation based model to identify the targets of microblogs. We assume that the microblog posts and targets describe the same topic using different languages. Hence, the target identification problem can be regarded as a translation process from the content of the microblogs to the targets. We integrate latent topical information into the translation model to facilitate the translation process. Because product names, series, and other related information are important indicators for this task, we also incorporate this background knowledge into the model. To evaluate the proposed method, we collect a large number of microblogs and manually annotate a subset of these as golden standards. We compare the proposed method with state-of-the-art methods using the constructed dataset. Experimental results demonstrate that the proposed approach can achieve better performance than the other approaches.

2 The Proposed Method

2.1 The Generation Process

Given a corpus $D = \{d_i, 1 \leq i \leq |D|\}$, which contains a list of microblogs $\{d_i\}$. A microblog is a sequence of N_d words denoted by $w_d = \{w_{d1}, w_{d2}, ..., w_{dN_d}\}$. Each microblog contains a set of targets denoted by $c_d = \{c_{d1}, c_{d2}, ..., c_{dM_d}\}$. A word is defined as an item from a vocabulary with V distinct words indexed by $w = \{w_1, w_2, ..., w_V\}$. The nth word in the dth microblog is associated with not only one topic z_{dn}, but also an indicator variable l_{dn} which indicates whether w_{dn} belongs to the ontology ($l_{dn} = 1$), which contains company names, product names, series, and other related information, or is a common word ($l_{dn} = 0$). Each target is from the vocabulary with C distinct company names indexed by $c = \{c_1, c_2, ..., c_C\}$. The mth target in the dth microblog is associated with a topic z_{dm}. The notations used in this paper are summarized in Table 1. Fig. 1 shows the graphical representation of the generation process. The generative story for each microblog is as follows:

1. Sample word distribution $\phi^{t,l}$ from $Dir(\beta^l)$ for each topic $t = 1, 2, ..., T$ and each label $l = 1, ..., L$.
2. For each microblog d=1,2,...,$|D|$
   a. Sample topic distribution θ_d from $Dir(\alpha)$
   b. For each word $n = 1, 2, ..., N_d$
      i. Sample a topic $z_{dn} = t$ from $Multinomial(\theta_d)$
      ii. Sample a label $l_{dn} = l$ from the distribution over labels, $v^{d,n}$
      iii. Sample a word w according to multinomial distribution $P(w_{dn} = w|z_{dn} = t, l_{dn} = l, \phi^{t,l})$
   c. For each target $m = 1, 2, ..., M_d$
      i. Sample a topic $z_{dm} = t$ from $Multinomial(\theta_d)$
      ii. Sample a target $c_{dm} = c$ according to probability $P(c_{dm} = c|w_d, l_d, z_{dm} = t, B)$

As described above, we use l_{dn} to incorporate the ontology information into the model. In this work, we construct an ontology which contains 4,926 company names, 7,632 abbreviations, and 26,732 product names. These companies names are collected based on the top search queries in different categories [4]. We propose to use the distribution $v^{d,n}$ to indicate the probability of variable l_{dn}. We set $v^{d,n}$ by applying

[2] http://nlp.uned.es/weps/weps-3

[3] http://www.limosine-project.eu/events/replab2013

[4] http://top.baidu.com/boards

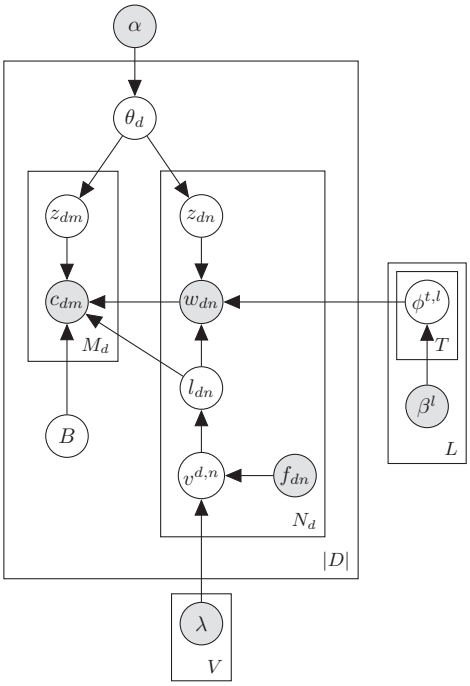


Figure 1: The graphical representation of the proposed model. Shaded circles are observations or constants. Unshaded ones are hidden variables.

various sources of ontology (presented by λ) and the context features of the word w_{dn} (presented by f_{dn}). In this work, we only consider the word itself as its context feature. This information is encoded into the hyperparameters $\{\lambda^w | w \in \{w_1, w_2, ..., w_V\}\}$, where λ^w is hyperparameter for the word w, and $\lambda_0^w + \lambda_1^w = 1$. For each word w in the ontology, we set λ_1^w to a value 0.9, λ_0^w to a value 0.1. For each word w not contained by ontology, we set λ_1^w to a value 0 and λ_0^w to a value 1. Based on the ontology, $v^{d,n}$ could be set as follows:

$$P(l_{dn} = l | w_{dn} = w) = v_l^{d,n} = \frac{\lambda_l^w}{\lambda_1^w + \lambda_0^w}, l \in \{0, 1\} \tag{1}$$

2.2 Model Inference

We use collapsed Gibbs sampling (Griffiths and Steyvers, 2004) to obtain samples of hidden variable assignment and to estimate the model parameters from these samples.

On the microblog content side, the conditional probability of a latent topic and label for the nth word in the dth microblog is:

$$Pr(z_{dn} = t, l_{dn} = l | w_{dn} = w, \mathbf{w}^{\neg n}, \mathbf{z}^{\neg n}, \mathbf{l}^{\neg n}) \propto \frac{\lambda_l^w}{\lambda_1^w + \lambda_0^w} \times \frac{N_{t,l}^{w,\neg n} + \beta^l}{N_{t,l}^{\neg n} + V\beta^l} \times \frac{N_d^{t,\neg n} + \alpha}{N_d^{\neg n} + T\alpha}, \tag{2}$$

where $N_{t,l}^{w,\neg n}$ is the number of the word w that are assigned to topic t under the label l; $N_{t,l}^{\neg n}$ is the number of all the words that are assigned to topic t under the label l; $N_d^{t,\neg n}$ is the number of topic t in the microblog d; $N_d^{\neg n}$ is the number of all the topics in the document d; $\neg n$ indicates taking no account of the current position n.

Given the conditional probability of $z_{dn} = t, l_{dn} = l$, we formalize the marginal probability of $z_{dn} = t$ as follows:

$$Pr(z_{dn} = t | w_{dn} = w, \mathbf{w}^{\neg n}, \mathbf{z}^{\neg n}, \mathbf{l}^{\neg n}) \propto \sum_{l=0}^{L-1} \frac{\lambda_l^w}{\lambda_1^w + \lambda_0^w} \times \frac{N_{t,l}^{w,\neg n} + \beta^l}{N_{t,l}^{\neg n} + V\beta^l} \times \frac{N_d^{t,\neg n} + \alpha}{N_d^{\neg n} + T\alpha} \tag{3}$$

Table 1: The notation used in the proposed model.

$\|D\|$	The number of microblogs in the data set
V	The number of unique words in the vocabulary
C	The number of companies
T	The number of topics
L	The number of labels
N_d	The number of words in the dth microblog
M_d	The number of companies in the dth microblog
w_d	All the words in the dth microblog
c_d	All the targets in the dth microblog
z_d	The topic of the words in the dth microblog
l_d	The label of the words in the dth microblog
B	The topic-specific word alignment table between a word and a target
$\phi^{t,l}$	Distribution of words for each topic t and each label l
θ_d	Distribution of topics in microblog d
$v^{d,n}$	Distribution of labels for word w_{dn}
$N_{t,l}^{w,\neg n}$	The number of the word w that is assigned to topic t under the label l except the position n
$N_{t,l}^{\neg n}$	The number of all the words that are assigned to topic t under the label l. except the position n
$N_d^{t,\neg n}$	The number of topic t in the microblog d except the position n
$N_d^{\neg n}$	The number of all the topics in the microblog d except the position n
$N_{t,l}^{c,w}$	The number of the target c that co-occurs with the word w labeled as l under topic t

After re-assigning the topic $z_{dn} = t$ for the current word, the conditional probability of ontology label for the nth word in the dth microblog is:

$$Pr(l_{dn} = l | w_{dn} = w, z_{dn} = t, \mathbf{w}^{\neg n}, \mathbf{z}^{\neg n}, \mathbf{l}^{\neg n}) \propto \frac{\lambda_l^w}{\lambda_1^w + \lambda_0^w} \times \frac{N_{t,l}^{w,\neg n} + \beta^l}{N_{t,l}^{\neg n} + V\beta^l} \tag{4}$$

On the target side, we perform topic assignments for each target as follows:

$$Pr(z_{dm} = t | c_{dm} = c, \mathbf{c}^{\neg m}, \mathbf{w}, \mathbf{l}, \mathbf{z}^{\neg m}) \propto \sum_{n=1}^{N_d} \delta^{l_{dn}} \frac{N_{t,l_{dn}}^{c,w_{dn},\neg m}}{N_{t,l_{dn}}^{w_{dn}} + \gamma C} \times \frac{N_d^{t,\neg m} + \alpha}{N_d^{\neg m} + T\alpha}, \tag{5}$$

where $\delta^{l_{dn}}$ is the weight for the label ($\delta^1 > 1, \delta^0 = 1$); $N_{t,l_{dn}}^{c,w_{dn},\neg m}$ is the number of the company c that co-occurs with the word w_{dn} labeled as l_{dn} under topic t; γC is a smoothing part; $N_{t,l_{dn}}^{w_{dn}}$ is the number of the word w_{dn} labeled as l_{dn} under topic t; $N_d^{t,\neg m}$ is the number of occurrences of topic t in the document d; $N_d^{\neg m}$ is the number of occurrences of all the topics in the document d; $\neg m$ indicates taking no account of the current position m.

Based on the above equations, after enough sampling iterations, we can estimate word alignment table B, $B_{c,w,t,l} = \delta^l \frac{N_{t,l}^{c,w}}{N_{t,l}^w + \gamma C}$. Some companies just occur few times, and most of the words co-occur with them also alignment with other companies, for this case, we use γC to smooth, where C represent the number of company c. And also we can estimate topic distribution θ for each document, and word distribution ϕ for each topic and each label, as follows:

$$\theta_d^t = \frac{N_d^t + \alpha}{N_d + T\alpha}, \quad \phi_w^{t,l} = \frac{N_{t,l}^w + \beta^l}{N_{t,l} + V\beta^l}$$

The possibility table $B_{c,w,t,l}$ has a potential size of $V \cdot C \cdot T \cdot L$. The data sparsity may pose a problem in estimating $B_{c,w,t,l}$. To reduce the data sparsity problem, we introduce the remedy in our model. We

employ a linear interpolation with topic-free word alignment probability to avoid data sparsity problem:

$$B^*_{c,w,t,l} = \sigma B_{c,w,t,l} + (1-\sigma)P(c|w), \tag{6}$$

where $P(c|w)$ is topic-free word alignment probability between the word w and the company c. σ is trade-off of two probabilities ranging from 0.0 to 1.0.

2.3 Target Company Extraction

Just like standard LDA, the proposed method itself finds a set of topics but does not directly extract targets. Suppose we have a dataset which contains microblogs without targets, we can use the collapsed Gibbs sampling to estimate the topic and label for the words in each microblog. The process is the same as described in Section 3.2.

After the hidden topics and label of the words in each microblog become stable, we can estimate the distribution of topics for the dth microblog by: $P(t|w_d) = \theta^t_d = \frac{N^t_d + \alpha}{N_d + T\alpha}$. With the word alignment table B^*, we can rank companies for the dth microblog in unlabeled data by computing the scores:

$$Pr(c_{dm}|w_d) \propto \sum_{t=1}^{T}\sum_{n=1}^{N_d} P(c_{dm}|t, w_{dn}, l_{dn}, B^*) \cdot P(t|w_d)P(w_{dn}|w_d), \tag{7}$$

where $P(w_{dn}|w_d)$ is the weight of the word w_{dn} in the microblog content w_d. In this paper, we use inverse document frequency (IDF) score to estimate it. Based on the ranking scores calculated by Eq.(7), we can extract the top-ranked targets for each microblog to users.

3 Experiments

In this section, we will introduce the experimental results and datasets we constructed for training and evaluation. We will firstly describe the how we construct the datasets and their statistics. Then we will introduce the experiment configurations and baseline methods. Finally, the evaluation results and analysis will be given.

3.1 Datasets

We started by using Sina Weibo's API[5] to collect public microblogs from randomly selected users. The dataset contains 282.2M microblogs published by 1.1M users. We use *RAW-Weibo* to represent it in the following sections. Based on the collected raw microblogs, we constructed three datasets for evaluation and training.

3.1.1 Training data

Since social media users post thoughts, ideas, or status on various topics in social medias, there are a huge number of related companies. Manually constructing training data is a time consuming and cost process. In this work, we propose a weakly manual method based on ontology and hashtag. A hashtag is a string of characters preceded by the symbol #. In most cases, hashtags can be viewed as an indication to the context of the tweet or as the core idea expressed in the tweet. Hence, we can use hashtag as the targets.

We extract the microblogs whose hashtags contain ontology items as training data and the corresponding ontology items as targets. Obviously, the training data constructed based on this method is not perfect. However, since this method can effectively generate a great quantity of data, we think that general characteristics can be modeled with the generated training data. To evaluate the corpus, we randomly selected 100 microblogs from the training data and manually labeled their targets. The accuracy of the sampled dataset is 91%. It indicates that the proposed training data generation method is effective. From the *RAW-Weibo* dataset, we extracted a total of 1.79M microblogs whose hashtags contain more than one target. Training instances for 2,574 target companies are included in the training data.

[5]http://open.weibo.com/

3.1.2 Test data

For evaluation, we manually constructed a dataset *RAN-Weibo*, which contains 2,000 microblogs selected from *RAW-Weibo*. Three annotators were asked to label the target companies for each microblog. To evaluate the quality of annotated dataset, we validate the agreements of human annotations using Cohen's kappa coefficient. The average κ among all annotators is 0.626. It indicates that the annotations are reliable.

Since some targets are ambiguous, inspired by the evaluation campaigns WePS-3 and RepLab 2013, we also constructed a dataset *AMB-Weibo*, where microblogs include 10 popular company names which may cause ambiguity. For each target, we randomly selected and annotated 200 microblogs as golden standards. Three annotators were also asked to label whether the microblog is related the given target or not. The agreements of human annotations were also validated through Cohen's kappa coefficient. The average κ among all annotators is 0.692.

3.2 Experiment Configurations

We use precision (P), recall (R), and F1-score (F_1) to evaluate the performance. We ran our model with 500 iterations of Gibbs sampling. We use 5-fold cross-validation in the training data to optimize hyperparameters. The number of topics is set to 30. The other settings of hyperparameters are as follows: $\alpha = 50/T$, $\beta = 0.1$, $\delta = 20$, $\gamma = 0.5$. The smoothing parameter σ is set to 0.8.

For baselines, we compare the proposed model with the following baseline methods.

- **Naive Bayes (NB)**: The target identification task can be easily formalized as a classification task, where each target is considered as a classification label. Hence, we applied Naive Bayes to model the posterior probability of each target given a microblog.
- **Support Vector Machine (SVM)**: The content of microblogs are represented as vectors and SVM is used to model the classification problem.
- **IBM1**: Translation model (IBM model-1) is applied to obtain the alignment probability between words and targets.
- **TTM**: Topical translation model (TTM) was proposed by Ding et al. (2013) to achieve microblog hashtag suggestion task. We adopted it to estimate the alignment probability between words and targets.

3.3 Experimental Results

We evaluate the proposed method from the following perspectives: 1) comparing the proposed method with the state-of-the-art methods on the two evaluation datasets; 2) identifying the impacts of parameters.

Table 2 shows the comparisons of the proposed method with the state-of-the-arts discriminative and generative methods on the evaluation dataset *RAN-Weibo*. "*Our*" denotes the method proposed in previous sections. "*Our w/o BG*" represents the proposed method without background knowledge. From the results, we can observe that the proposed method is better than other methods. Discriminative methods achieve worse results than generative methods. We think that the large number of targets is one of the main reasons of the low performances. The results of the proposed models with and without ontology information also show that background knowledge can benefit both the precision and recall. TTM achieves better performance than IBM1. It indicates that topical information is useful for this task. The performances of our method are significantly better than TTM. It illustrates that our smoothing method and incorporation of background knowledge are effective.

From the description of the proposed model, we can know that there are several hyperparameters in the proposed model. To evaluate the impacts of them, we evaluate two crucial ones among all of them, the number of topics T and the smoothing factor σ. Table 3 shows the influence of the number of topics. From the table, we can observe that the proposed model obtains the best performance when T is set to 30. And performance decreases with more number of topics. We think that data sparsity may be one of the main reasons. With much more topic number, the data sparsity problem will be more serious when

Table 2: Evaluation results of NB, SVM, IBM1, TTM, and our method on the evaluation dataset *RAN-Weibo*.

Methods	Precision	Recall	F_1
NB	0.168	0.154	0.161
SVM	0.312	0,286	0.298
IBM1	0.236	0.214	0.220
TTM	0.356	0.327	0.341
Our w/o BG	0.488	0.448	0.467
Our	**0.522**	**0.479**	**0.500**

Table 3: The influence of the number of topics T of the proposed method.

T	Precision	Recall	F_1
10	0.516	0.473	0.493
30	**0.522**	**0.479**	**0.500**
50	0.508	0.466	0.486
70	0.489	0.449	0.468
100	0.488	0.448	0.467

estimating topic-specific translation probability. Table 4 shows the influence of the translation probability smoothing parameter σ. When σ is set to 0.0, it means that the topical information is omitted. Comparing the results of $\sigma = 0.0$ and other values, we can observe that the topical information can benefit this task. When σ is set to 1.0, it represents the method without smoothing. The results indicate that it is necessary to address the sparsity problem through smoothing.

Figure 2 shows the results of different methods on the dataset *AMB-Weibo*. All the models are trained with same dataset as the above experiments. From the results, we can observe that the F1-scores vary from less than 0.40 up to almost 0.60. The performances' variations of other methods are also huge. We think that training data size and difficulty level are two main reasons. The size of training data of different targets vary greatly in the dataset. However, comparing with other method, the proposed method is the most stable one. Comparing with other methods, the proposed method achieves better performance than other methods for all targets.

4 Related Work

Organization name disambiguation task is fundamental problems in many NLP applications. The task aims to distinguish the real world relevant of a given name with the same surface in context. WePS-3[6] (Amigó et al., 2010) and RepLab 2013[7] (Amigó et al., 2013) evaluation campaigns have also addressed the problem from the perspective of disambiguation organization names in microblogs. Pedersen et al. (2006) proposed an unsupervised method for name discrimination. Yerva et al. (2010) used support vector machines (SVM) classifier with various external resources, such as WordNet, metadata profile, category profile, Google set, and so on. Kozareva and Ravi (2011) proposed to use latent dirichlet allocation to incorporate topical information. Zhang et al. (2012) proposed to use adaptive method for this task. However, most of these methods focused on the text with predefined surface words. The documents which do not contain organization names or person names can not be well processed by these methods.

To bridge the vocabulary gap between content and hashtags, Liu et al. (2012b) proposed to use translation model to handle it. They modeled the tag suggestion task as a translation process from

[6]http://nlp.uned.es/weps/weps-3

[7]http://www.limosine-project.eu/events/replab2013

Table 4: The influence of the smoothing parameter σ of the propose method.

σ	Precision	Recall	F_1
0.0	0.471	0.432	0.451
0.2	0.490	0.449	0.469
0.4	0.495	0.454	0.474
0.6	0.511	0.468	0.489
0.8	**0.522**	**0.479**	**0.500**
1.0	0.519	0.476	0.496

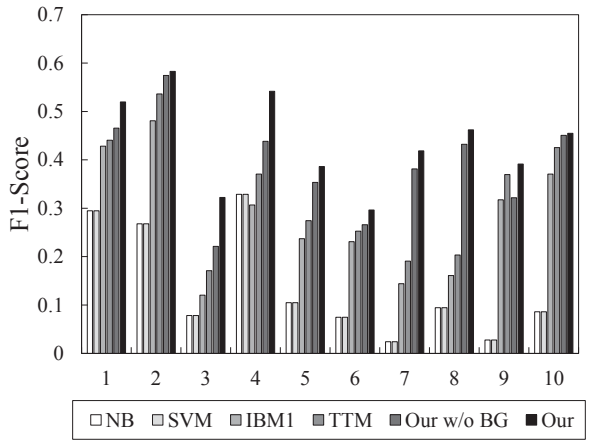


Figure 2: Evaluation results of NB, SVM, IBM1, TTM, and our method on the different companies in the test dataset *AMB-Weibo*.

document content to tags. Ding et al. (2013) extended the translation based method and introduced a topic-specific translation model to process the multiple meanings of words in different topics. Motivated by these methods, we also propose to use topic-specific translation model to handle vocabulary problem. Based on the model, in this work, we incorporate the background knowledge information into the model.

5 Conclusions

To identify target companies of microblogs, in this paper, we propose a novel topical translation model to achieve the task. The main assumption is that the microblog posts and targets describe the same thing with different languages. We convert the target identification problem to a translation process from content of microblogs to targets. We integrate latent topical information into translation model to hand the themes of microblogs in facilitating the translation process. We also incorporate background knowledge (such as product names, series, et al.) into the generation model. Experimental results on a large corpus constructed from a real microblog service and a number of manually labeled golden standards of easily ambiguous entities demonstrate that the proposed method can achieve better performance than other approaches.

6 Acknowledgement

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by 973 Program (2010CB327900), National Natural Science Foundation of China (61003092,61073069), Shanghai Leading Academic Discipline Project (B114) and "Chen Guang" project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation(11CG05).

References

Enrique Amigó, Javier Artiles, Julio Gonzalo, Damiano Spina, Bing Liu, and Adolfo Corujo. 2010. Weps3 evaluation campaign: Overview of the on-line reputation management task. In *CLEF (Notebook Papers/LABs/Workshops)*.

Enrique Amigó, Jorge Carrillo de Albornoz, Irina Chugur, Adolfo Corujo, Julio Gonzalo, Tamara Martín, Edgar Meij, Maarten Rijke, and Damiano Spina. 2013. Overview of replab 2013: Evaluating online reputation monitoring systems. In *Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, volume 8138 of *Lecture Notes in Computer Science*, pages 333–352. Springer Berlin Heidelberg.

S. Asur and B.A. Huberman. 2010. Predicting the future with social media. In *Proceedings of WI-IAT 2010*.

Hila Becker, Mor Naaman, and Luis Gravano. 2010. Learning similarity metrics for event identification in social media. In *Proceedings of WSDM '10*.

Adam Bermingham and Alan F. Smeaton. 2010. Classifying sentiment in microblogs: is brevity an advantage? In *Proceedings of CIKM '10*.

Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. 2008. Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of WWW '08*.

Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8.

Zhuoye Ding, Xipeng Qiu, Qi Zhang, and Xuanjing Huang. 2013. Learning topical translation model for microblog hashtag suggestion. In *Proceedings of IJCAI 2013*.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. In *Proceedings of the National Academy of Sciences*, volume 101, pages 5228–5235.

Ido Guy, Naama Zwerdling, Inbal Ronen, David Carmel, and Erel Uziel. 2010. Social media recommendation based on people and tags. In *Proceedings of SIGIR '10*.

Ido Guy, Uri Avraham, David Carmel, Sigalit Ur, Michal Jacovi, and Inbal Ronen. 2013. Mining expertise and interests from social media. In *Proceedings of WWW '13*.

Minqing Hu and Bing Liu. 2004. Mining opinion features in customer reviews. In *Proceedings of AAAI'04*.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of ACL-HLT 2011*, Portland, Oregon, USA.

Zornitsa Kozareva and Sujith Ravi. 2011. Unsupervised name ambiguity resolution using a generative model. In *Proceedings of the First Workshop on Unsupervised Learning in NLP*, EMNLP '11, pages 105–112, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chia-Jung Lee and W. Bruce Croft. 2013. Building a web test collection using social media. In *Proceedings of SIGIR '13*, SIGIR '13.

Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceedings of CIKM '09*.

Kang Liu, Liheng Xu, and Jun Zhao. 2012a. Opinion target extraction using word-based translation model. In *Proceedings of EMNLP-CoNLL '12*.

Zhiyuan Liu, Chen Liang, and Maosong Sun. 2012b. Topical word trigger model for keyphrase extraction. In *Proceedings of COLING*.

Kang Liu, Liheng Xu, and Jun Zhao. 2013. Syntactic patterns versus word alignment: Extracting opinion targets from online reviews. In *Proceedings of ACL 2013*, Sofia, Bulgaria.

Takanobu Otsuka, Takuya Yoshimura, and Takayuki Ito. 2012. Evaluation of the reputation network using realistic distance between facebook data. In *Proceedings of WI-IAT '12*, Washington, DC, USA.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January.

Ted Pedersen, Anagha Kulkarni, Roxana Angheluta, Zornitsa Kozareva, and Thamar Solorio. 2006. An unsupervised language independent method of name discrimination using second order co-occurrence features. In *Computational Linguistics and Intelligent Text Processing*, pages 208–222.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of HL-EMNLP 2005*, Vancouver, British Columbia, Canada.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Comput. Linguist.*, 37(1):9–27, March.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 851–860, New York, NY, USA. ACM.

Damiano Spina, Edgar Meij, Maarten de Rijke, Andrei Oghina, Minh Thuong Bui, and Mathias Breuss. 2012. Identifying entity aspects in microblog posts. In *Proceedings of SIGIR '12*.

Damiano Spina, Julio Gonzalo, and Enrique Amigó. 2013. Discovering filter keywords for company name disambiguation in twitter. *Expert Systems with Applications*, 40(12):4986 – 5003.

Yuanbin Wu, Qi Zhang, Xuangjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *Proceedings of EMNLP 2009*, Singapore.

Surender Reddy Yerva, Zoltn Mikls, and Karl Aberer. 2010. It was easy, when apples and blackberries were only fruits. In Martin Braschler, Donna Harman, and Emanuele Pianta, editors, *CLEF (Notebook Papers/LABs/Workshops)*.

Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *Proceedings of COLING '10*.

Shu Zhang, Jianwei Wu, Dequan Zheng, Yao Meng, and Hao Yu. 2012. An adaptive method for organization name disambiguation with feature reinforcing. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*, pages 237–245, Bali,Indonesia, November. Faculty of Computer Science, Universitas Indonesia.

Inducing Latent Semantic Relations for Structured Distributional Semantics

Sujay Kumar Jauhar
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
sjauhar@cs.cmu.edu

Eduard Hovy
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
hovy@cs.cmu.edu

Abstract

Structured distributional semantic models aim to improve upon simple vector space models of semantics by hypothesizing that the meaning of a word is captured more effectively through its relational — rather than its raw distributional — signature. In accordance, they extend the vector space paradigm by structuring elements with relational information that decompose distributional signatures over discrete relation dimensions. However, the number and nature of these relations remains an open research question, with most previous work in the literature employing syntactic dependencies as surrogates for truly semantic relations. In this paper we propose a novel structured distributional semantic model with latent relation dimensions, and instantiate it using latent relational analysis. Evaluation of our model yields results that significantly outperform several other distributional approaches on two semantic tasks and performs competitively on a third relation classification task.

1 Introduction

The distributional hypothesis, articulated by Firth (1957) in the popular dictum "You shall know the word by the company it keeps", has established itself as one of the most popular models of modern computational semantics. With the rise of massive and easily-accessible digital corpora, computation of co-occurrence statistics has enabled researchers in NLP to build distributional semantic models (DSMs) that have found relevance in many application areas. These include information retrieval (Manning et al., 2008), question answering (Tellex et al., 2003), word-sense disambiguation (McCarthy et al., 2004) and selectional preference modelling (Erk, 2007), to name only a few.

The standard DSM framework, which models the semantics of a word by co-occurrence statistics computed over its neighbouring words, has several known short-comings. One severe short-coming derives from the fundamental nature of the vector space model, which characterizes the semantics of a word by a single vector in a high dimensional space (or some lower dimensional embedding thereof).

Such a modelling paradigm goes against the grain of the intuition that the semantics of a word is neither unique nor constant. Rather, it is composed of many facets of meaning, and similarity (or dissimilarity) to other words is an outcome of the aggregate harmony (or dissonance) between the individual facets under consideration. For example, a shirt may be similar along one facet to a balloon in that they are both coloured blue, at the same time being similar to a shoe along another facet for both being articles of clothing, while being dissimilar along yet another facet to a t-shirt because one is stitched from linen while the other is made from polyester.

Structured distributional semantic models (SDSMs) aim to remedy this fault with DSMs by decomposing distributional signatures over discrete relation dimensions, or facets. This leads to a representation that characterizes the semantics of a word by a distributional tensor, rather than a vector. Previous attempts in the literature include the work of Padó and Lapata (2007), Baroni and Lenci (2010) and Goyal et al. (2013). However, all these approaches assume a simplified representation in which truly semantic relations are substituted by syntactic relations obtained from a dependency parser.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 698–708, Dublin, Ireland, August 23-29 2014.

We believe that there are limiting factors to this approximation. Most importantly, the set of syntactic relations, while relatively uncontroversial, is unable to capture the full extent of semantic nuance encountered in natural language text. Often, syntax is ambiguous and leads to multiple semantic interpretations. Conversely, passivization and dative shift are common examples of semantic invariance in which multiple syntactic realizations are manifested. Additionally, syntax falls utterly short in explaining more complex phenomena – such as the description of buying and selling – in which implicit semantics are tacit from complex interactions between multiple participants.

While it is useful to consider relations that draw their origins from semantic roles such as Agent, Patient and Recipient, it remains unclear what this set of semantic roles should be. This problem is one that has long troubled linguists (Fillmore, 1967; Sowa, 1991), and has been previously noted by researchers in NLP as well (Màrquez et al., 2008). Proposed solutions range from a small set of generic Agent-like or Patient-like roles in Propbank (Kingsbury and Palmer, 2002) to an effectively open-ended set of highly specific and fine-grained roles in Framenet (Baker et al., 1998). In addition to the theoretic uncertainty of the set of semantic relations there is the very real problem of the lack of high-performance, robust semantic parsers to annotate corpora. These issues effectively render the use of pre-defined, linguistically ordained semantic relations intractable for use in SDSM.

In this paper we propose a novel approach to structuring distributional semantic models with *latent* relations that are automatically discovered from corpora. This approach effectively solves the conceptual dilemma of selecting the most expressive set of semantic relations. To the best of our knowledge this is the first paper to propose latent relation dimensions for SDSMs. The intuition for generating these latent relation dimensions leads to a generic framework, which — in this paper — is instantiated with embeddings obtained from latent relational analysis (Turney, 2005).

We conduct experiments on three different semantic tasks to evaluate our model. On a similarity scoring task and another synonym ranking task the model significantly outperforms other distributional semantic models, including a standard window-based model, a syntactic SDSM based on previous approaches proposed in the literature, and a state-of-the-art semantic model trained using recursive neural networks. On a relation classification task, our model performs competitively, outperforming all but one of the models it is compared against.

2 Related Work

Since the distributional hypothesis was first proposed by Firth (1957), a number of different research initiatives have attempted to extend and improve the standard distributional vector space model of semantics. Insensitivity to the multi-faceted nature of semantics has been one of the focal points of several papers. Earlier work in this regard is a paper by Turney (2012), who proposes that the semantics of a word is not obtained along a single distributional axis but simultaneously in two different spaces. He proposes a DSM in which co-occurrence statistics are computed for neighbouring nouns and verbs separately to yield independent domain and function spaces of semantics.

This intuition is taken further by a stance which proposes that a word's semantics is distributionally decomposed over *many* independent spaces – each of which is a unique relation dimension. Authors who have endorsed this perspective are Erk and Padó (2008), Goyal et al. (2013), Reisinger and Mooney (2010) and Baroni and Lenci (2010). Our work relates to these papers in that we subscribe to the multiple space semantics view. However, we crucially differ from them by structuring our semantic space with information obtained from latent semantic relations rather than from a syntactic parser. In this paper the instantiation of the SDSM with latent relation dimensions is obtained using LRA (Turney, 2005), which is an extension of LSA (Deerwester et al., 1990) to induce relational embeddings for pairs of words.

From a modelling perspective, SDSMs characterize the semantics of a word by a distributional tensor. Other notable papers on tensor based semantics or semantics of compositional structures are the simple additive and multiplicative models of Mitchell and Lapata (2009), the matrix-vector neural network approach of Socher et al. (2012), the physics inspired quantum view of semantic composition of Grefenstette and Sadrzadeh (2011) and the tensor-factorization model of Van de Cruys et al. (2013).

A different, partially overlapping strain of research attempts to induce word embeddings using meth-

ods from deep learning, yielding state-of-the-art results on a number of different tasks. Notable research papers on this topic are the ones by Collobert et al. (2011), Turian et al. (2010) and Socher et al. (2010).

Other related work to note is the body of research concerned with semantic relation classification, which is one of our evaluation tasks. Research community wide efforts in the SemEval-2007 task 4 (Girju et al., 2007), the SemEval-2010 task 8 (Hendrickx et al., 2009) and the SemEval-2012 task 2 (Jurgens et al., 2012) are notable examples. However, different from our work, most previous attempts at semantic relation classification operate on the basis of feature engineering and contextual cues (Bethard and Martin, 2007).

3 Structured Distributional Semantics and Latent Semantic Relation Induction

In this section we formalize the notion of SDSM as an extension of DSM and present a novel SDSM with latent relation dimensions.

A DSM is a vector space V that contains $|\Sigma|$ elements in $\mathbb{R}^n$, where $\Sigma = \{w_1, w_2, ..., w_k\}$ is a vocabulary of k distinct words. Every vocabulary word w_i has an associated semantic vector $\vec{v_i}$ representing its distributional signature. Each of the n elements of $\vec{v_i}$ is associated with a single dimension of its distribution. This dimension may correspond to another word — that may or may not belong to Σ — or a latent dimension as might be obtained from an SVD projection or an embedding learned via a deep neural network. Additionally, each element in $\vec{v_i}$ is typically a normalized co-occurrence frequency count, a PMI score, or a number obtained from an SVD or RNN transformation. The semantic similarity between two words w_i and w_j in a DSM is the vector distance defined by $cos(\vec{v_i}, \vec{v_j})$ on their associated distributional vectors.

An SDSM is an extension of DSM. Formally, it is a space U that contains $|\Sigma|$ elements in $\mathbb{R}^{d \times n}$, where $\Sigma = \{w_1, w_2, ..., w_k\}$ is a vocabulary of k distinct words. Every vocabulary word w_i has an associated semantic tensor $\vec{\vec{u_i}}$, which is itself composed of d vectors $\vec{u_{i1}}, \vec{u_{i2}}, ..., \vec{u_{id}}$ each having n dimensions. Every vector $\vec{u_{il}} \in \vec{\vec{u_i}}$ represents the distributional signature of the word w_i in a relation (or along a facet) r_l. The d relations of the SDSM may be syntactic, semantic, or latent (as in this paper). The n dimensional relational vector $\vec{u_{il}}$ is configurationally the same as a vector $\vec{v_i}$ of a DSM. This definition of an SDSM closely relates to an alternate view of Distributional Memories (DMs) (Baroni and Lenci, 2010) where the semantic space is a third-order tensor, whose modes are Word $\times$ Link $\times$ Word.

The semantic similarity between two words w_i and w_j in an SDSM is the similarity function defined by $sim(\vec{\vec{u_i}}, \vec{\vec{u_j}})$ on their associated semantic tensors. We use the following decomposition of the similarity function:

$$sim(\vec{\vec{u_i}}, \vec{\vec{u_j}}) = \frac{1}{d} \sum_{l=1}^{d} cos(\vec{u_{il}}, \vec{u_{jl}}) \tag{1}$$

Mathematically, this corresponds to the ratio of the normalized Frobenius product of the two matrices representing $\vec{\vec{u_i}}$ and $\vec{\vec{u_j}}$ to the number of rows in both matrices. Intuitively it is simply the average relation-wise similarity between the two words w_i and w_j.

3.1 Latent Relation Induction for SDSM

The intuition behind our approach for inducing latent relation dimensions revolves around the simple observation that SDSMs, while representing semantics as distributional signatures over relation dimensions, also effectively encode relational vectors between pairs of words. Our method thus works backwards from this observation — beginning with a relational embedding for pairs of words, that are subsequently transformed to yield an SDSM.

Concretely, given a vocabulary $\Gamma = \{w_1, w_2, ..., w_k\}$ and a list of word pairs of interest from the vocabulary $\Sigma_V \subseteq \Gamma \times \Gamma$, we assume that we have some method for inducing a DSM V' that has a vector representation $\vec{v'_{ij}}$ of length d for every word pair $w_i, w_j \in \Sigma_V$, which intuitively embeds the distributional signature of the relation binding the two words in d latent dimensions. We then construct an SDSM U where $\Sigma_U = \Gamma$. For every word $w_i \in \Gamma$ a tensor $\vec{\vec{u_i}} \in \mathbb{R}^{d \times k}$ is generated. The tensor $\vec{\vec{u_i}}$

has d unique k dimensional vectors $\vec{u_{i1}}, \vec{u_{i2}}, ..., \vec{u_{id}}$. For a given relational vector $\vec{u_{il}}$, the value of the jth element is taken from the lth element of the vector $\vec{v_{ij}}$ belonging to the DSM V'. If the vector $\vec{v_{ij}}$ does not exist in V' – as is the case where the pair $w_i, w_j \notin \Sigma_V$ – the value of the jth element of $\vec{u_{il}}$ is set to 0. By applying this mapping to generate semantic tensors for every word in Γ, we are left with an SDSM U that effectively embeds latent relation dimensions. From the perspective of DMs we matricize the third-order tensor and perform truncated SVD, before restoring the resulting matrix to a third-order tensor.

3.1.1 Latent Relational Analysis

In what follows, we present our instantiation of this model with an implementation that is based on Latent Relational Analysis (LRA) (Turney, 2005) to generate the DSM V'. While other methods (such as RNNs) are equally applicable in this scenario, we use LRA for its operational simplicity as well as proven efficacy on semantic tasks such as analogy detection. The parameter values we chose in our experiments are not fine-tuned and are guided by recommended values from Turney (2005), or scaled suitably to accommodate the size of Σ_V.

The input to LRA is a vocabulary $\Gamma = \{w_1, w_2, ..., w_k\}$ and a list of word pairs of interest from the vocabulary $\Sigma_V \subseteq \Gamma \times \Gamma$. While one might theoretically consider a large vocabulary with all possible pairs, for computational reasons we restrict our vocabulary to approximately 4500 frequent English words and only consider about 2.5% word pairs with high PMI (as computed on the whole of English Wikipedia) in $\Gamma \times \Gamma$. For each of the word pairs $w_i, w_j \in \Sigma_V$ we extract a list of contexts by querying a search engine indexed over the combined texts of the whole of English Wikipedia and Gigaword corpora (approximately 5.8×10^9 tokens). Suitable query expansion is performed by taking the top 4 synonyms of w_i and w_j using Lin's thesaurus (Lin, 1998). Each of these contexts must contain both w_i, w_j (or appropriate synonyms) and optionally some intervening words, and some words to either side.

Given such contexts, patterns for every word pair are generated by replacing the two target words w_i and w_j with placeholder characters X and Y, and replacing none, some or all of the other words by their associated part-of-speech tag or a wildcard symbol. For example, if w_i and w_j are "eat" and "pasta" respectively, and the queried context is "I eat a bowl of pasta with a fork", one would generate patterns such as "* X * NN * Y IN a *", "* X DT bowl IN Y with DT *", etc. For every word pair, only the 5000 most frequent patterns are stored.

Once the set of all relevant patterns $P = p_1, p_2, ..., p_n$ have been computed a DSM V is constructed. In particular, the DSM constitutes a Σ_V based on the list of word pairs of interest, and every word pair w_i, w_j of interest has an associated vector $\vec{v_{ij}}$. Each element m of the vector $\vec{v_{ij}}$ is a count pertaining to the number of times that the pattern p_m was generated by the word pair w_i, w_j.

3.1.2 SVD Transformation

The resulting DSM V is noisy and very sparse. Two transformations are thus applied to V. Firstly all co-occurrence counts between word pairs and patterns are transformed to PPMI scores (Bullinaria and Levy, 2007). Then given the matrix representation of V — where rows correspond to word pairs and columns correspond to patterns — SVD is applied to yield $V = M\Delta N$. Here M and N are matrices that have unit-length orthogonal columns and Δ is a matrix of singular values. By selecting the d top singular values, we approximate V with a lower dimension projection matrix that reduces noise and compensates for sparseness: $V' = M_d\Delta_d$. This DSM V' in d latent dimensions is precisely the one we then use to construct an SDSM, using the transformation described above.

Since the large number of patterns renders it effectively impossible to store the entire matrix V in memory we use a memory friendly implementation[1] of a multi-pass stochastic algorithm to directly approximate the projection matrix (Halko et al., 2011; Rehurek, 2010). A detailed analysis to see how change in the parameter d effects the quality of the model is presented in section 4.

The optimal SDSM embeddings we trained and used in the experiments detailed below are available for download at `http://www.cs.cmu.edu/~sjauhar/Software_files/LR-SDSM.tar.`

[1] `http://radimrehurek.com/gensim/`

Model	Spearman's ρ
Random	0.000
DSM	0.179
synSDSM	0.315
SENNA	0.510
LR-SDSM (300)	0.567
LR-SDSM (130)	**0.586**

Table 1

Model	Acc.
Random	0.25
DSM	0.28
synSDSM	0.27
SENNA	0.38
LR-SDSM (300)	0.47
LR-SDSM (130)	**0.51**

Table 2

Results on the WS-353 similarity scoring task and the ESL synonym selection task. LRA-SDSM significantly outperforms other structured and non-structured distributional semantic models.

gz. This SDSM contains a vocabulary of 4546 frequent English words with 130 latent relation dimensions.

4 Evaluation

Section 3 has described a method for embedding latent relation dimensions in SDSMs. We now turn to the problem of evaluating these relations within the scope of the distributional paradigm in order to address two research questions: 1) Are latent relation dimensions a viable and empirically competitive solution for SDSM? 2) Does structuring lead to a semantically more expressive model than a non-structured DSM? In order to answer these questions we evaluate our model on two generic semantic tasks and present comparative results against other structured and non-structured distributional models. We show that we outperform all of them significantly, thus answering both research questions affirmatively.

While other research efforts have produced better results on these tasks (Jarmasz and Szpakowicz, 2003; Gabrilovich and Markovitch, 2007; Hassan and Mihalcea, 2011), they are either lexicon or knowledge based, or are driven by corpus statistics that tie into auxiliary resources such as multi-lingual information and structured ontologies like Wikipedia. Hence they are not relevant to our experimental validation, and are consequently ignored in our comparative evaluation.

4.1 Word-Pair Similarity Scoring Task

The first task consists in using a semantic model to assign similarity scores to pairs of words. The dataset used in this evaluation setting is the WS-353 dataset from Finkelstein et al. (2002). It consists of 353 pairs of words along with an averaged similarity score on a scale of 1.0 to 10.0 obtained from 13–16 human judges. Word pairs are presented as-is, without any context. For example, an item in this dataset might be "book, paper $\rightarrow$ 7.46".

System scores are obtained by using the standard cosine similarity measure between distributional vectors in a non-structured DSM. In the case of a variant of SDSM, these scores can be found by using the cosine-based similarity functions in Equation 1 of the previous section. System generated output scores are evaluated against the gold standard using Spearman's rank correlation coefficient.

4.2 Synonym Selection Task

In the second task, the same set of semantic space representations is used to select the semantically closest word to a target from a list of candidates. The ESL dataset from Turney (2002) is used for this task, and was selected over the slightly larger TOEFL dataset (Landauer and Dumais, 1997). The reason for this choice was because the latter contained more complex vocabulary words — several of which were not present in our simple vocabulary model. The ESL dataset consists of 50 target words that appear with 4 candidate lexical substitutes each. While disambiguating context is also given in this dataset, we discard it in our experiments. An example item in this dataset might be "rug $\rightarrow$ sofa, ottoman, carpet, hallway", with "carpet" being the most synonym-like candidate to the target.

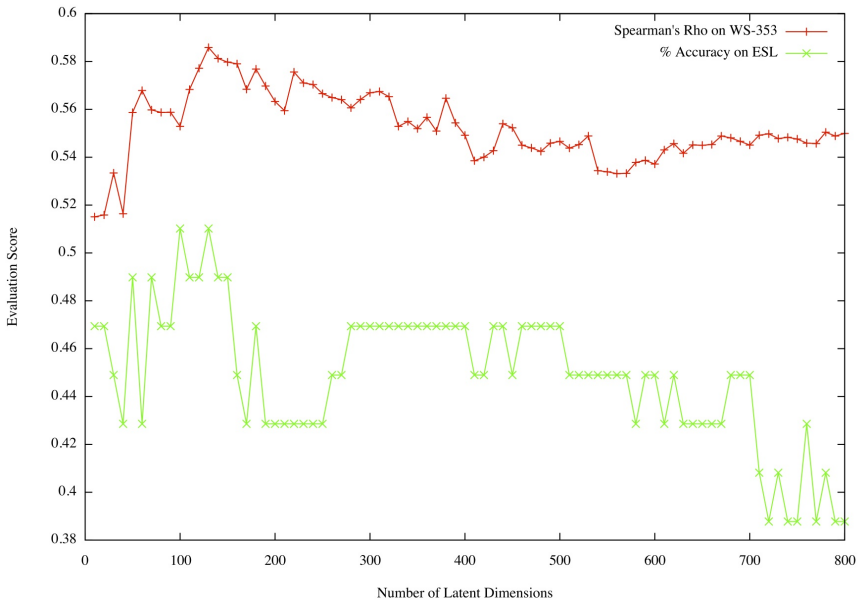


Figure 1: Evaluation results on WS-353 and ESL with varying number of latent dimensions. Generally high scores are obtained in the range of 100-150 latent dimensions, with optimal results on both datasets at 130 latent dimensions.

Similarity scores — which are obtained in the same manner as for the previous evaluation task — are extracted between the target and each of the candidates in turn. These scores are then sorted in descending order, with the top-ranking score yielding the semantically closest candidate to the target. Systems are evaluated on the basis of their accuracy at discriminating the top-ranked candidate.

4.3 Results

We compare our model (LR-SDSM) to several other distributional models in these experiments. These include a standard distributional vector space model (DSM) trained on the combined text of English Wikipedia and Gigaword with a window-size of 3 words to either side of a target, a syntax-based SDSM (Goyal et al., 2013; Baroni and Lenci, 2010) (synSDSM) trained on the same corpus parsed with a dependency parser (Tratz and Hovy, 2011) and the state-of-the-art neural network embeddings from Collobert et al. (2011) (SENNA). We also give the expected evaluation scores from a random baseline, for comparison.

An important factor to consider when constructing an SDSM using LRA is the number of latent dimensions selected in the SVD projection. In Figure 1 we investigate the effects of selecting different number of latent relation dimensions on both semantic evaluation tasks, starting with 10 dimensions up to a maximum of 800 (which was the maximum that was computationally feasible), in increments of 10. We note that optimal results on both datasets are obtained at 130 latent dimensions. In addition to the SDSM obtained in this setting we also give results for an SDSM with 300 latent dimensions (which has been a recommended value for SVD projections in the literature (Landauer and Dumais, 1997)) in our comparisons against other models. Comparative results on the Finkelstein WS-353 similarity scoring task are given in Table 1, while those on the ESL synonym selection task are given in Table 2.

4.4 Discussion

The results in Tables 1 and 2 show that LR-SDSM outperforms the other distributional models by a considerable and statistically significant margin (p-value < 0.05) on both types of semantic evaluation tasks. It should be noted that we do not tune to the test sets. While the 130 latent dimension SDSM yields the best results, 300 latent dimensions also gives comparable performance and moreover outperforms all the other baselines. In fact, it is worth noting that the evaluation results in figure 1 are almost all better

	Random	SENNA-Mik	DSM		SENNA		**LR-SDSM**
			AVC	MVC	AVC	MVC	
Prec.	0.111	0.273	0.419	0.382	**0.489**	0.416	0.431
Rec.	0.110	0.343	0.449	0.443	**0.516**	0.457	0.475
F-1.	0.110	0.288	0.426	0.383	**0.499**	0.429	0.444
% Acc.	11.03	34.30	44.91	44.26	**51.55**	45.65	47.48

Table 3: Results on Relation Classification Task. LR-SDSM scores competitively, outperforming all but the SENNA-AVC model.

than the results of the other models on either datasets.

We conclude that structuring of a semantic model with latent relational information in fact leads to performance gains over non-structured variants. Also, the latent relation dimensions we propose offer a viable and empirically competitive alternative to syntactic relations for SDSMs.

Figure 1 shows the evaluation results on both semantic tasks as a function of the number of latent dimensions. The general trend of both curves on the figure indicate that the expressive power of the model quickly increases with the number of dimensions until it peaks in the range of 100–150, and then decreases or evens out after that. Interestingly, this falls roughly in the range of the 166 frequent (those that appear 50 times or more) frame elements, or fine-grained relations, from FrameNet that O'Hara and Wiebe (2009) find in their taxonomization and mapping of a number of lexical resources that contain semantic relations.

5 Semantic Relation Classification and Analysis of the Latent Structure of Dimensions

In this section we conduct experiments on the task of semantic relation classification. We also perform a more detailed analysis of the induced latent relation dimensions in order to gain insight into our model's perception of semantic relations.

5.1 Semantic Relation Classification

In this task, a relational embedding is used as a feature vector to train a classifier for predicting the semantic relation between previously unseen word pairs. The dataset used in this experiment is from the SemEval-2012 task 2 on measuring the degree of relational similarity (Jurgens et al., 2012), since it characterizes a number of very distinct and interesting semantic relations. In particular it consists of an aggregated set of 3464 word pairs evidencing 10 kinds of semantic relations. We prune this set to discard pairs that don't contain words in the vocabularies of the models we consider in our experiments. This leaves us with a dataset containing 933 word pairs in 9 classes (1 class was discarded altogether because it contained too few instances). The 9 semantic relation classes are: "Class Inclusion", "Part-Whole", "Similar", "Contrast", "Attribute", "Non-Attribute", "Case Relation", "Cause-Purpose" and "Space-Time". For example, an instance of a word pair that exemplifies the "Part-Whole" relationship is "engine:car". Note that, as with previous experiments, word pairs are given without any context.

5.2 Results

We compare LR-SDSM on the semantic relation classification task to several different models. These include the additive vector composition (AVC) and multiplicative vector composition methods (MVC) proposed by Mitchell and Lapata (2009); we present both DSM and SENNA based variants of these models. We also compare against the vector difference method of Mikolov et al. (2013) (SENNA-Mik) which sees semantic relations as a meaning preserving vector translation in an RNN embedded vector space. Finally, we note the performance of random classification as a baseline, for reference. We attempted to produce results of a syntactic SDSM on the task; however, the hard constraint imposed by syntactic adjacency meant that effectively all the word pairs in the dataset yielded zero feature vectors.

To avoid overfitting on all 130 original dimensions in our optimal SDSM, and also to render results comparable, we reduce the number of latent relation dimensions of LR-SDSM to 50. We similarly reduce

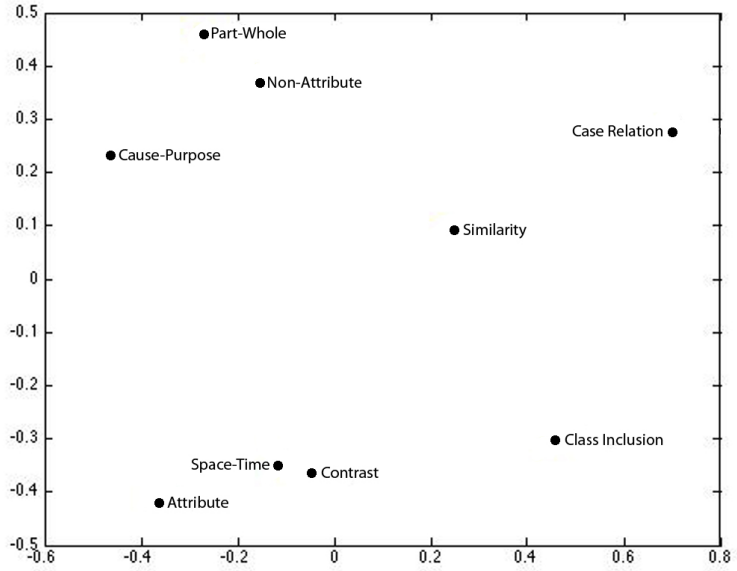


Figure 2: Correlation distances between semantic relations' classifier weights. The plot shows how our latent relations seem to perceive humanly interpretable semantic relations. Most points are fairly well spaced out, with opposites such as "Attribute" and "Non-Attribute" as well as "Similar" and "Contrast" being relatively further apart.

the feature vector dimension of DSM-AVC and DSM-MVC to 50 by feature selection. The dimensions of SENNA-AVC, SENNA-MVC and SENNA-Mik are already 50, and are not reduced further.

For each of the methods we train a logistic regression classifier. We don't perform any tuning of parameters and set a constant ridge regression value of 0.2, which seemed to yield roughly the best results for all models. The performance on the semantic relation classification task in terms of averaged precision, recall, F-measure and percentage accuracy using 10-fold cross-validation is given in Table 3.

Additionally, to gain further insight into the LR-SDSM's understanding of semantic relations, we conduct a secondary analysis. We begin by training 9 one-vs-all logistic regression classifiers for each of the 9 semantic relations under consideration. Then pairwise correlation distances are measured between all pairs of weight vectors of the 9 models. Finally, the distance adjacency matrix is projected into 2-d space using multidimensional scaling. The result of this analysis is presented in Figure 2.

5.3 Discussion

Table 3 shows that LR-SDSM performs competitively on the relation classification task and outperforms all but one of the other models. The performance differences are statistically significant with a p-value < 0.5. We believe that some of the expressive power of the model is lost by compressing to 50 latent relation dimensions, and that a greater number of dimensions might improve performance. However, testing a model with a 130-length dense feature vector on a dataset containing 933 instances would likely lead to overfitting and also not be comparable to the SENNA-based models that operate on 50-length feature vectors.

Other points to note from Table 3 are that the AVC variants of the the DSM and SENNA composition models tend to perform better than their MVC counterparts. Also, SENNA-Mik performs surprisingly poorly. It is worth noting, however, that Mikolov et al. (2013) report results on fairly simple lexico-syntactic relations between words – such as plural forms, possessives and gender – while the semantic relations under consideration in the SemEval-2012 dataset are relatively more complex.

In the analysis of the latent structure of dimensions presented in Figure 2, there are few interesting points to note. To begin with, all the points (with the exception of one pair) are fairly well spaced out. At

the weight vector level, this implies that different latent dimensions need to fire in different combinations to characterize distinct semantic relations, thus resulting in low correlation between their corresponding weight vectors. This indicates the fact that the latent relation dimensions seem to capture the intuition that each of the classes encodes a distinctly different semantic relation. The notable exception is "Space-Time", which is very close to "Contrast". This is probably due to the fact that distributional models are ineffective at capturing spatio-temporal semantics. Moreover, it is interesting to note that "Attribute" and "Non-Attribute" as well as "Similar" and "Contrast", which are intuitively semantic inverses of each other are also (relatively) distant from each other in the plot.

These general findings indicate an interesting avenue for future research, which involves mapping the empirically learnt latent relations to hand-built semantic lexicons or frameworks. This could help to validate the empirical models at various levels of linguistic granularity, as well as establish correspondences between different views of semantic representation.

6 Conclusion and Future Work

In this paper we have proposed a novel paradigm for SDSMs, that allows for structuring via latent relational information. We have introduced a generic operational framework that allows for building such SDSMs and outlined an instantiation of the model with LRA. Experimental results of the model support our claim that the resulting SDSM captures the semantics of words more effectively than a number of other semantic models, and presents a viable — and empirically competitive — alternative to syntactic SDSMs. Additionally we have conducted experiments on a relation classification task and shown promising results, as well as performed analyses to investigate the structure of, and interactions between, the latent relation dimensions.

These findings motivate a number of future directions of research. Since our framework is fairly general we hope to explore techniques other than LRA (such as RNNs) to generate relational embeddings for word pairs. A desiderata for future techniques is scalability so that we can characterize vocabularies that are larger than the one in our current experiments. We also hope to explore mappings between our empirically learnt latent relations, and semantic lexicons and frameworks that catalog semantic relations. Finally, we hope to test our model on more realistic application task such as event coreference, recognizing textual entailment, and semantic parsing in future work.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. This work was supported in part by the following grants: NSF grant IIS-1143703, NSF award IIS-1147810, DARPA grant FA87501220342.

References

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics.

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Comput. Linguist.*, 36(4):673–721, December.

Steven Bethard and James H Martin. 2007. Cu-tmp: temporal relation classification using syntactic and semantic features. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 129–132. Association for Computational Linguistics.

John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510–526.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 999888:2493–2537, November.

Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 897–906, Stroudsburg, PA, USA. Association for Computational Linguistics.

Katrin Erk. 2007. A simple, similarity-based model for selectional preferences.

Charles J Fillmore. 1967. The case for case.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. In *ACM Transactions on Information Systems*, volume 20, pages 116–131, January.

John R. Firth. 1957. A Synopsis of Linguistic Theory, 1930-1955. *Studies in Linguistic Analysis*, pages 1–32.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*, volume 7, pages 1606–1611.

Roxana Girju, Preslav Nakov, Vivi Nastase, Stan Szpakowicz, Peter Turney, and Deniz Yuret. 2007. Semeval-2007 task 04: Classification of semantic relations between nominals. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 13–18. Association for Computational Linguistics.

Kartik Goyal, Sujay Kumar Jauhar, Huiying Li, Mrinmaya Sachan, Shashank Srivastava, and Eduard Hovy. 2013. A structured distributional semantic model for event co-reference. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL – 2013)*.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1394–1404, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288.

Samer Hassan and Rada Mihalcea. 2011. Semantic relatedness using salient semantic analysis. In *AAAI*.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 94–99. Association for Computational Linguistics.

Mario Jarmasz and Stan Szpakowicz. 2003. Roget's thesaurus and semantic similarity. *Recent Advances in Natural Language Processing III: Selected Papers from RANLP*.

David A Jurgens, Peter D Turney, Saif M Mohammad, and Keith J Holyoak. 2012. Semeval-2012 task 2: Measuring degrees of relational similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 356–364. Association for Computational Linguistics.

Paul Kingsbury and Martha Palmer. 2002. From treebank to propbank. In *LREC*. Citeseer.

Thomas K Landauer and Susan T. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Lluís Màrquez, Xavier Carreras, Kenneth C Litkowski, and Suzanne Stevenson. 2008. Semantic role labeling: an introduction to the special issue. *Computational linguistics*, 34(2):145–159.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. *Proceedings of NAACL-HLT*, pages 746–751.

Jeff Mitchell and Mirella Lapata. 2009. Language models based on semantic composition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 430–439, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tom O'Hara and Janyce Wiebe. 2009. Exploiting semantic role resources for preposition disambiguation. *Computational Linguistics*, 35(2):151–184.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

Radim Rehurek. 2010. Fast and faster: A comparison of two streamed matrix decomposition algorithms. *NIPS 2010 Workshop on Low-rank Methods for Large-scale Machine Learning*.

Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1201–1211, Stroudsburg, PA, USA. Association for Computational Linguistics.

John F. Sowa. 1991. Principles of semantic networks. Morgan Kaufmann.

Stefanie Tellex, Boris Katz, Jimmy J. Lin, Aaron Fernandes, and Gregory Marton. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *SIGIR*, pages 41–47.

Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1257–1268, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Peter D. Turney. 2002. Mining the web for synonyms: Pmi-ir versus lsa on toefl. *CoRR*.

Peter Turney. 2005. Measuring semantic similarity by latent relational analysis. In *Proceedings of the 19th international Conference on Aritifical Intelligence*, pages 1136–1141.

Peter D Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44:533–585.

Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. 2013. A tensor-based factorization model of semantic compositionality. In *Proceedings of NAACL-HLT*, pages 1142–1151.

Improving distributional thesauri by exploring the graph of neighbors

Vincent Claveau
IRISA - CNRS
Campus de Beaulieu
35042 Rennes, France
vincent.claveau@irisa.fr

Ewa Kijak
IRISA - Univ. of Rennes 1
Campus de Beaulieu
35042 Rennes, France
ewa.kijak@irisa.fr

Olivier Ferret
CEA, LIST
LVIC
91191 Gif-sur-Yvette, France
olivier.ferret@cea.fr

Abstract

In this paper, we address the issue of building and improving a distributional thesaurus. We first show that existing tools from the information retrieval domain can be directly used in order to build a thesaurus with state-of-the-art performance. Secondly, we focus more specifically on improving the obtained thesaurus, seen as a graph of k-nearest neighbors. By exploiting information about the neighborhood contained in this graph, we propose several contributions. 1) We show how the lists of neighbors can be globally improved by examining the reciprocity of the neighboring relation, that is, the fact that a word can be close of another and vice-versa. 2) We also propose a method to associate a confidence score to any lists of nearest neighbors (i.e. any entry of the thesaurus). 3) Last, we demonstrate how these confidence scores can be used to reorder the closest neighbors of a word. These different contributions are validated through experiments and offer significant improvement over the state-of-the-art.

1 Introduction

Distributional thesauri are useful for many NLP tasks and their construction is an issue widely discussed for several years (Grefenstette, 1994). However this is still a very active research field, maintained by the increasingly large number of available corpus and by many applications. These thesauri associate each of their entry with a list of words that are desired semantically close to the entry. This notion of proximity varies (synonymy, other paradigmatic relations, syntagmatic relations (Budanitsky and Hirst, 2006; Adam et al., 2013, for a discussion)), but the methods used for the automatic construction of thesauri are often shared. For the most part, these methods rely on the distributional hypothesis of (Firth, 1957): each word is characterized by the set of contexts in which it appears, and the semantic proximity of two words can be inferred from the proximity of their contexts. This hypothesis has been implemented in different ways, and several propositions to improve the results have been explored (see next section for a state of the art).

The work presented in this article are part of this framework. We propose several contributions on the creation of these distributional thesauri and their improvement. We first show that models from information retrieval (IR) can provide information on semantic relationships, and are thus adapted to the task of creating these thesauri. In addition, they offer very competitive results compared to the state of the art, while enjoying existing tools (Section 3).

The most important part of our work then focuses on the exploitation of such semantic neighborhood relations. The IR models indeed provide lists ordering all words by decreasing similarity, that form a graph of nearest neighbors. We propose to take advantage of some of the neighborhood information contained in this graph and we derive three contributions.
1) We globally improve neighbor lists by taking into account the reciprocity of the neighborhood relationship, that is to say the fact that a word is a close neighbor of another and vice versa (Section 4).
2) We also propose a method that associates each neighbor list (i.e. each entry of the thesaurus built) with a confidence score (Section 5). This method uses the nearest neighbor graph to estimate the probabilities that a given word is the i-th neighbor of another word.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 709–720, Dublin, Ireland, August 23-29 2014.

3) Finally, on the basis of this work, we show how to use this confidence score and these probabilities to reorder the list of nearest neighbors (Section 6). To achieve this goal, we model the reranking as an optimization problem of assignments, solved by the Hungarian algorithm (Kuhn and Yaw, 1955).

2 Related work

The notion of distributional thesaurus, as it was initially defined by Grefenstette (1994), followed by Lin (1998a) and Curran and Moens (2002), is not often considered specifically, probably because of its strong link with the notion of semantic similarity. As a consequence, the improvement of distributional thesauri has been first a side effect of the improvement of the distributional similarity measures used for their building and more precisely, of the distributional data they rely on. Both the nature of the constituents of distributional contexts and their weighting have been considered in this regard. Concerning their weighting, Broda et al. (2009) proposed to turn the weights of context constituents into ranks to make them less dependent on a specific weighting function while Zhitomirsky-Geffet and Dagan (2009), extended by Yamamoto and Asakura (2010), defined a bootstrapping method for modifying the weights of constituents in the distributional context of a word according to the similarity with its semantic neighbors.

The nature of distributional contexts has been first considered through the distinction between window-based and syntactic co-occurrents (Grefenstette, 1994; Curran and Moens, 2002). However, most of the work related to this issue has focused on the fact that the "traditional" representation of distributional contexts is very sparse and redundant, as illustrated by Hagiwara et al. (2006). Hence, several methods for dimension reduction were tested in this context: from Latent Semantic Analysis (Landauer and Dumais, 1997), extended for syntactic co-occurrents (Padó and Lapata, 2007), to Random Indexing (Sahlgren, 2001), Non-negative Matrix Factorization (Van de Cruys, 2010) and more recently, lexical representations learnt by neural networks (Huang et al., 2012; Mikolov et al., 2013).

The work we present in this article follows a different perspective as our objective is to improve an existing distributional thesaurus by relying on its structure through a reranking of its neighbors. Such approach was adopted to some extent by Zhitomirsky-Geffet and Dagan (2009) as it exploited the neighbors of an entry in an initial thesaurus for reweighting its distributional representation and finally, reranking its neighbors. Ferret (2013) proposed a more indirect method in which the reranking is based on the downgrading of the neighbors that are detected as not similar to their entry through a pseudo word sense disambiguation task: such detection occurs if a certain proportion of the occurrences of a neighbor are not tagged as the entry. Finally, the closest work to ours is (Ferret, 2012), which selects in an unsupervised way a set of examples of semantically similar words from an initial thesaurus for training a classifier whose decision function is used for reranking the neighbors of each entry. Its unsupervised selection of examples is more precisely based on the symmetry of semantic similarity relations.

As Ferret (2012), our work exploits a certain kind of symmetry in the relation of distributional neighborhood between words but extends it to a larger scale by considering the initial thesaurus as a k-nearest neighbor graph and using the relations in this graph for reranking the neighbors of each entry, similarly to Pedronette et al. (2014) in the context of image retrieval.

3 IR models for building distributional thesauri

3.1 Principles

As mentioned in the state of the art, distributional approaches aim to calculate similarities between textual representations of word contexts. Methods to calculate similarities from IR seem then relevant for this problem. For a given word, the set of contexts of all its occurrences is considered as a document. The proximity between two words is then measured on their contexts by a similarity function from IR. This idea has many links with the work from the state of the art, but seems relatively unexplored, with the exeption of (Vechtomova and Robertson, 2012) in the specific context of similar named entities. It offers the advantage of being easily implementable because of the numerous IR tools available. Some adaptations are of course required. In contrast to IR, the stop words are kept as well as their positions relative to the considered occurrence. Lemmatization instead of stemming is performed. For example, in the excerpt: "... all forms of restrictions on freedom of expression, threats ...", the indexing terms restriction-2,

on-1, of+1, expression+2 are added to the context of freedom noted $\mathcal{C}(\text{freedom})$. The whole set of collected contexts for a word is used as a query in order to find its distributional neighbors. According to an IR similarity measure, the nearest words of this query (those whose contexts are closest) are returned.

We tested some of the most classical similarity measures used in IR: Hellinger (Escoffier, 1978; Domengès and Volle, 1979), TF-IDF/cosinus, and Okapi-BM-25 (Robertson et al., 1998). The last model can be seen as a variation of TF-IDF that better takes into account the difference between document sizes. This point is of importance since in our case the documents (namely the set of contexts of a word) are actually of very variable sizes, due to the very variable number of occurrences of each word. The Okapi-BM25 similarity between a word w_i ($\mathcal{C}(w_i)$ being the query), and w_j ($\mathcal{C}(w_j)$ being a document), is given in Eqn 1.

$$\text{similarity}(w_i, w_j) = \sum_{t \in \mathcal{C}(w_i)} \frac{(k_3+1)*qtf}{k_3+qtf} * \frac{tf*(k_1+1)}{tf+k_1*(1-b+b*\frac{dl(\mathcal{C}(w_j))}{dl_{avg}})} * \log \frac{n-df(t)+0.5}{df(t)+0.5} \quad (1)$$

qtf is the number of occurrences of the word t in the query ($\mathcal{C}(w_i)$), dl is the size of $\mathcal{C}(w_j)$), dl_{avg} the average size of all contexts, n is the number of documents (that means in our case the number of considered words/thesaurus entries). $df(t)$ is the number of contexts ($\mathcal{C}(\cdot)$) containing t. Finally, k_1, k_3 and b are some constants, with default values $k_1 = 2$, $k_3 = 1000$ and $b = 0.75$. Details of these classical IR models are not given here but can be found in (Manning et al., 2008).

In the following experiments, the context of an occurrence is defined by the two words before and after the occurrence, and we also use an adjusted version of Okapi-BM25 similarity that enhances the influence of the document size and gives more importance to the most discriminating context words by setting $b = 1$ and putting the IDF squared to give more importance to the most discriminating context words.

3.2 Experimental setup

For the sake of comparison, we use in our experiments the data and baselines provided by Ferret (2013). The corpus used to build the distributional thesaurus is AQUAINT-2. It is a collection of articles from press containing about 380 million words. The thesaurus entries are all the nouns in the corpus with a frequency > 10. That represents 25,000 entries (i.e. unique nouns), denoted by n in the remaining. The corpus is labeled in parts of speech by TreeTagger (Schmid, 1994). In this way, we can identify the names that form the thesaurus entries and thus compare to existing work. However this information is not used to build the thesaurus, ensuring the portability of the method to other languages, similarly to (Freitag et al., 2005).

To evaluate the built thesauri, WordNet 3.0 synonyms (Miller, 1990) and Moby (Ward, 1996) are used as references, either separately, or jointly. These two resources exhibit quite different and complementary characteristics: on the one hand, WordNet indicates strong paradigmatic links between words (synonyms or quasi-synonyms). On the other hand, Moby groups words sharing more extended syntagmatic and paradigmatic relations, including synonymy, hyper/hypo-nymy, meronymy, but also many more complex types such as the composition of co-hyponymy and hyponymy (*abolition – annulment*, *cataclysm – debacle*) or hypernymy and co-hyponymy (*abyss – rift*, *algorithm – routine*). As a result, WordNet provides lists of 3 neighbors on average for the 10,473 names of the corpus it covers, while Moby provides lists of 50 neighbors on average for 9,216 names. When combined, the two resources provide a reference of 38 neighbors on average for 12,243 names. It is this combination of WordNet and Moby that will be used as the main reference in all evaluations of this article. Some results restricted to WordNet or Moby only as reference are also given in some cases to illustrate the impact of our methods on semantic simlarity versus semantic relatedness relations.

Through this intrinsic evaluation framework, the semantic neighbors of about half of the entries of our thesauri are evaluated, which can be considered as a very large evaluation set compared to classical benchmarks such as WordSim 353 for instance (Gabrilovich and Markovitch, 2007). This kind of intrinsic evaluation is of course limited by the relations that are present in the resources used as gold standards, often restricted to "classical" relation types such as synonymy or hypernymy. In our case, this limitation

Reference	Method	MAP	R-Prec	P@1	P@5	P@10	P@50	P@100
WordNet + Moby	Ferret 2013 *base*	5.6	7.7	22.5	14.1	10.8	5.3	3.8
	Ferret 2013 *best rerank*	6.1	8.4	24.8	15.4	11.7	5.7	3.8
	Hellinger	2.45	2.89	9.73	6.28	5.31	4.12	3.30
	TF-IDF	5.40	7.28	21.73	13.74	9.59	5.17	3.49
	Okapi-BM25	6.72	8.41	24.82	14.65	10.85	5.16	3.66
	Okapi-BM25 ajusted	8.97	10.94	31.05	18.44	13.76	6.46	4.54
	Ferret 2014 *synt*	7.9	10.7	29.4	18.9	14.6	7.3	5.2
WordNet	Ferret 2013 *base*	9.8	8.2	11.7	5.1	3.4	1.1	0.7
	Ferret 2013 *best rerank*	10.7	9.1	12.8	5.6	3.7	1.2	0.7
	Okapi-BM25 ajusted	14.17	12.22	16.97	7.10	4.47	1.41	0.84
	Ferret 2014 *synt*	13.3	11.5	15.6	6.9	4.5	1.5	0.9
Moby	Ferret 2013 *base*	3.2	6.7	24.1	16.4	13.0	6.6	4.8
	Ferret 2013 *best rerank*	3.5	7.2	26.5	17.9	14.0	6.9	4.8
	Okapi-BM25 ajusted	5.69	9.14	32.18	21.37	16.42	8.02	5.69
	Ferret 2014 *synt*	4.8	9.4	30.6	21.7	17.3	8.9	6.5

Table 1: Performance of IR models for distributional thesaurus building with the references WordNet, Moby and WordNet+Moby

holds true for WordNet's synonyms but can be considered as far less restrictive for the related words of Moby, due to the diversity of their underlying relation types.

3.3 Results

For a given name, our approach by IR models returns a list of names ordered by decreasing similarity. This list is compared to the reference one by computing several classical measures (expressed in % in the following): the precision after k first names, denoted P@k, the Mean Average Precision (MAP) which is the mean of the average precision scores for each query after a reference synonym is found, and the R-precision (precision at R-th position in the ranking of results, where R is the number of relevant names for the query).

Table 1 indicates the performance of different models of IR similarities. For purposes of comparison, we show the results obtained under the same conditions by Ferret (2013), with both a state of the art approach based on using cosine similarity over pointwise mutual information between contexts (referred as *base* in the table), and an improved version by learning as described in section 2 (referred as *best rerank*). We also give the results on the same corpus on an approach based on syntactic co-occurrents (Ferret, 2014 in press), extracted with the Minipar syntactic parser as in (Lin, 1998b).

In these early results, it is worth noting that some IR similarities are quite inefficient, including the TF alone or Hellinger similarity. This is hardly surprising since these similarities use very basic weights that do not enhance the discriminative contexts of words. The similarities that include a notion of IDF get better results in this. Okapi BM25-based similarities offer good results. The standard Okapi version yields performance similar to the state of the art, and the adjusted version even widely outperforms the two systems from Ferret (2013), in particular in terms of overall quality (measured by the MAP). Moreover, the results of this adjusted version are comparable to those obtained with syntactic co-occurrents while it only exploits window-based co-occurents, known to give usually worst results than syntactic co-occurrents, without even lemmatization. This latest version of the system serves as reference for the rest of this article.

4 Reciprocity in the graph of k-NN

Computing all the similarities between all pairs of words produces a weighted graph of neighbors: Each word is connected with certain strength to the n other words. The results above do not reflect this structure. The following sections aim to examine how take advantage of the neighborhood relations embedded in this graph. It must be first noted that some of the IR similarity measures we used are not symmetric, including Okapi-BM25. The similarity between a word w_i, used as query, and another word w_j does not give the same value as the similarity between the query w_j and w_i. Apart from that, even if

the similarity measure it-self is symmetric, nearest neighbor relationships are not.

It seems then reasonable to assume that the reciprocity between two adjacent words (each belonging to the k nearest neighbors of the other) is a sign of confidence on the proximity between these words. Using this information to improve the previous results is discussed in this section. In the following, $\tau_{w_i}(w_j)$ denotes the rank of the word w_j in the list of neighbors of w_i. $\tau_{w_i}(w_j)$ thus varies from 1 to n.

4.1 Distributional neighborhood graph

Reciprocal relationship in distributional neighborhood has already been discussed and used in some work (Ferret, 2013) on distributional semantic, or more generally, on nearest neighbors graphs (Pedronette et al., 2014). In these papers, the reciprocity was considered for giving a new similarity score in a simple way. For a word w_i and its neighbor w_j, the maximal or the minimal rank between $\tau_{w_i}(w_j)$ and $\tau_{w_j}(w_i)$ is taken as the new rank. These two operators have too severe effects as only one rank is taken into consideration to decide the final score. This leads to highly degraded performance as shown later. Many other aggregation operators have however been proposed in other contexts with a behavior may be more appropriate to the task, including fuzzy logic (Detyniecki, 2000). These operators carry some semantic that allow to comprehend their behavior, such as T-norms (fuzzy logic AND) and S-norms (or T-conorms, fuzzy OR).

In this section, we test some of these operators without claiming to be exhaustive. These are defined on $[0, 1]^2$, 1 being the certainty. They are used to generate a new similarity score according to:

$$\text{score}_{w_i}(w_j) = \text{Aggreg}(1 - \tau_{w_i}(w_j)/n, 1 - \tau_{w_j}(w_i)/n) \quad (2)$$

where Aggreg is an aggregation operator. The new scores are then used to produce a new list of nearest neighbors of w_i (the higher the score, the greater proximity is proven). We thus perceive the semantic associated with these operators. For example, if the aggregation function is $\max$, we get the expected behavior of the fuzzy OR associated with this S-norm: w_j will be ranked very close to w_i in the new list if w_j was close to w_i *or* if w_i was close to w_j. For the T-norm $\min$, this happens if w_j is close to w_i *and* w_i is close to w_j.

4.2 Results

Besides the $\min$ and $\max$ aggregation operators, Figure 1 reports the results obtained with the following T-norms (or T-norm families dependent on a parameter γ) used as aggregation function Aggreg:

$\text{T}_{\text{Prob}}(x, y) = x * y$

$\text{T}_{\text{Lukasiewicz}}(x, y) = \max(x + y - 1, 0)$

$\text{T}_{\text{Hamacher}}(x, y) = \frac{x*y}{\gamma+(1-\gamma)*(x+y-x*y)}$; $\gamma \geq 0$

$\text{T}_{\text{Yager}}(x, y) = \max(0, 1 - \sqrt[\gamma]{(1-x)^\gamma + (1-y)^\gamma})$; $\gamma > 0$

We also tested the standard related S-norms, obtained by generalization of the De Morgan's law: $S(x, y) = 1 - T(1 - x, 1 - y)$. For the T-norm families dependent on a parameter, we varied this parameter value in a systematic way. The results reported correspond to the parameter values that maximize the MAP.

All these operators get very different results. Some operators, such as min, max, Lukasiewicz, and others for some γ, induce a threshold effect which degrades the performance: they return a default value generating too much ex aequo among the neighbors, for some values of $\tau_{w_i}(w_j)$ and $\tau_{w_j}(w_i)$. T-norms, focusing on pairs of words symmetrically close to each other, are too restrictive. This is consistent with the conclusions of the work cited: if the reciprocity condition is applied too strictly, it does not improve the nearest neighbor lists over all the words. In contrast, S-norms seem better able to take advantage of the ranking. The improvements are modest in terms of overall quality (MAP), but important at some ranks (e.g. P@10).

Finally, it is important to note that these results depend heavily on the resource used as reference. We tested the aggregation rank with $\text{S}_{\text{Hamacher}}$, $\gamma = 0.95$, on Moby and WordNet references separately. Results are given in Table 2. Because Wordnet is based on a synonymy relationship strong enough (and therefore reciprocal), the performance gains on WordNet are much higher than on Moby.

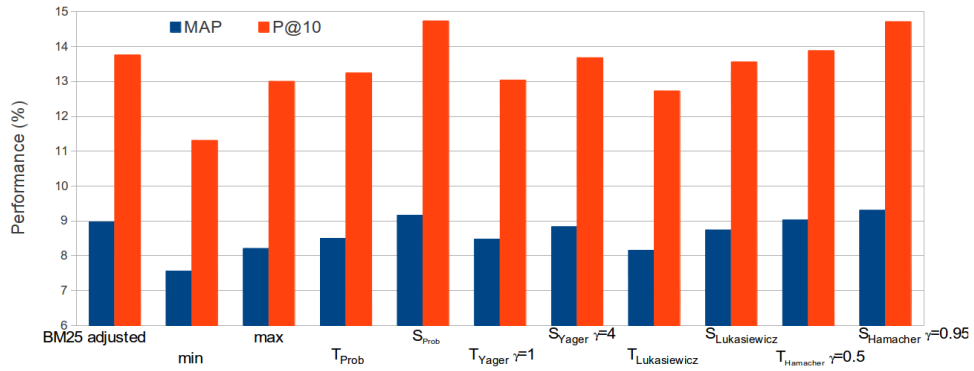


Figure 1: Performance of reciprocal rank aggregation, on the reference WordNet+Moby

Reference	MAP	R-Prec	P@1	P@5	P@10	P@50	P@100
WordNet + Moby	9.30 (+3.75)	11.06 (+2.03)	30.42 (-2.53)	19.29 (+4.58)	14.71 (+6.92)	7.09 (+9.78)	4.86 (+7.07)
WordNet	15.05 (+6.23)	12.81 (+4.81)	17.55 (+3.41)	7.96 (+12.16)	5.07 (+13.30)	1.63 (+15.69)	0.94 (+12.23)
Moby	5.90 (+3.65)	11.86 (+4.14)	*31.77 (-1.27)*	21.65 (+1.34)	17.0 (+3.53)	8.42 (+5.01)	5.92 (+4.12)

Table 2: Performance and gains (%) of reciprocal rank aggregation, relatively to adjusted Okapi-BM25, on the references WordNet and Moby taken separately, with aggregation operator S_{Hamacher}, $\gamma = 0.95$

5 Confidence estimation for a distributional neighborhood list

In the previous section, the rank of w_i in the list of neighbors of w_j is used to improve the ranking of w_j in the list of neighbors of w_i. We can also be interested in a more general way to the relative positions of w_i and w_j in all neighbor lists of all the words. Thereby, we expect to derive a more complete information. As a first step, we define a confidence criterion associated with each list of nearest neighbors, only based on the neighborhood graph.

5.1 Principle

We make the following assumption: the nearest neighbor list of a word w is probably of good quality if the distance (in terms of rank) between w and each of its neighbors w_i, denoted $\delta(w, w_i)$, is consistent with the distance observed between these same words (w, w_i) in other lists. The intuition here is that words supposed to be close should also be found close to the same other words. If k nearest neighbors of w have this property, then we attribute a high confidence to the neighbor list of w.

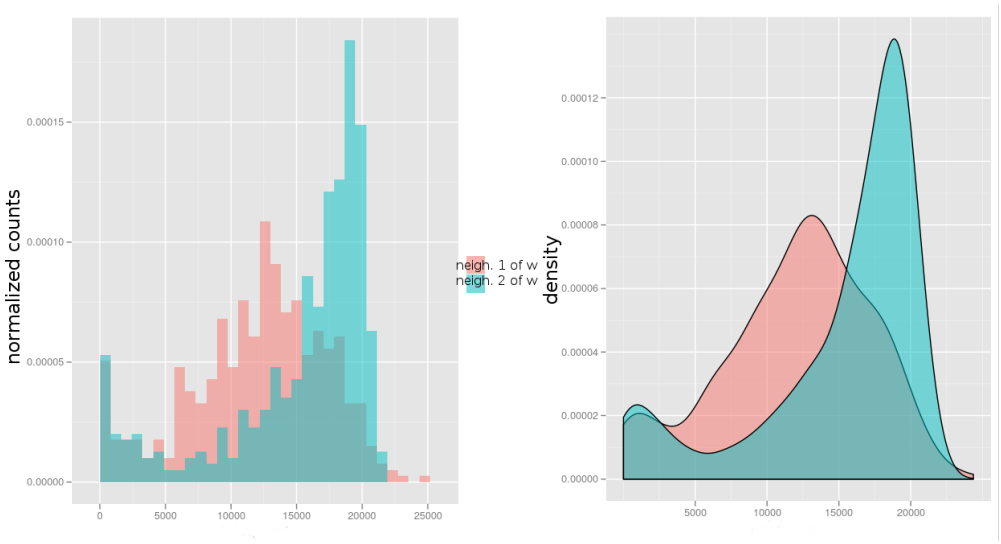
Formally, we define the confidence of the k-nearest neighbor list of w by:

$$Q(w) = \prod_{\{w_i | \tau_w(w_i) \leq k\}} p(\delta(w, w_i) = \tau_w(w_i)) \tag{3}$$

where $p(\delta(w, w_i) = \tau_w(w_i))$ is the probability that w_i is the $\tau_w(w_i)$-th neighbor of w. The problem is then to estimate the probability distribution $p(\delta(w, w_i))$ for each pair of words (w, w_i). To achieve this goal, we use the Parzen windows which is a method for nonparametric density estimation. We describe below how this classic method (Parzen, 1962; Wasserman, 2005) is applied in our context.

5.2 Parzen-window density estimation

Let $x_{ab} = \delta(w_a, w_b)$ be the distance (in terms of ranks) between two words w_a and w_b in a list of neighbors of any given word. Considering the n words of the thesaurus, we have a sample of n realizations assumed *iid*: (x^1_{ab}, x^2_{ab}, ... , x^n_{ab}), which are the observed distances between w_a et w_b in each (complete) neighbor list of each word. These counts can be represented by an histogram as illustrated in Figure 2 (a). Using the Parzen window technique, we can then estimate the probability density of x_{ab} with a kernel



(a) (b)

Figure 2: (a) Example of two distributions of distances x_{ab} and x_{ac} between a word w_a and two of its neighbors w_b and w_c, represented as histograms (blue and red) (b) Same distributions represented by densities estimated with the Parzen-windows method.

density estimator with Eqn 4 where h is a smoothing parameter called the bandwidth, and $K(\dot{)}$ is a kernel that we choose Gaussian. The resulting density is illustrated in Figure 2 (b).

$$\widehat{p_h}(x_{ab}) = \frac{1}{n\,h}\sum_{i=1}^{n} K\left(\frac{x_{ab} - x_{ab}^i}{h}\right) \quad \text{with} \quad K(u) = \frac{1}{\sqrt{2\pi}} e^{\frac{-u^2}{2}} \tag{4}$$

Thus, the resulting probability is a mixture of Gaussians centered on each x_{ab}^i. These methods are known to be sensitive to the bandwidth h, which controls the regularity of the estimation. The problem of choosing h is crucial in density estimation and was widely discussed in the literature. We use Silverman's rule of thumb (Silverman, 1986, page 48, eqn (3.31)) to set its value. Under the assumption of normality of the underlying distribution, this rule provides a simple way to calculate the optimal parameter h when Gaussian functions are used to approximate univariate data (Eqn 5 where $\hat{\sigma}$ is the standard deviation of the samples, and $q_3 - q_1$ is the interquartile range).

$$\hat{h} = 0.9\ \min(\hat{\sigma}, \frac{q_3 - q_1}{1.34})\ n^{-\frac{1}{5}} \tag{5}$$

Once these probabilities have been estimated on each of the k-nearest neighbors of w, we can calculate the confidence score $Q(w)$. The complexity of this estimation for all neighbor lists is $\mathcal{O}(k * n^2)$.

5.3 Using the confidence score

The expected benefit of using the confidence score is to have an a priori indication on the quality of a neighbor list for a given word. Such a score may thus be useful for many applications using thesauri produced by our approach (e.g. for expanding queries in information retrieval tasks). An evaluation of the confidence score through such applications would certainly be the most suitable, but beyond the scope of this article. We use default direct assessment towards the MAP: we measure the correlation between MAP and the confidence score, the idea being that an entry with a neighbor list of low quality matches an entry with low MAP. We use Spearman's correlation ρ and the Kendall's rank correlation coefficient τ, which do not make any assumption about linearity and compare only the order of words classified according to their MAP with the order according to their confidence score. The results of these coefficients are given in Table 3, along with p-value of the associated test of significance. A coefficient

Correlation coefficient	value	statistical significance
Kendall τ	0.37	$p < 10^{-64}$
Spearman ρ	0.51	$p < 10^{-64}$

Table 3: Correlation coefficient values between the MAP and the confidence score, and their statistical significance (p-value).

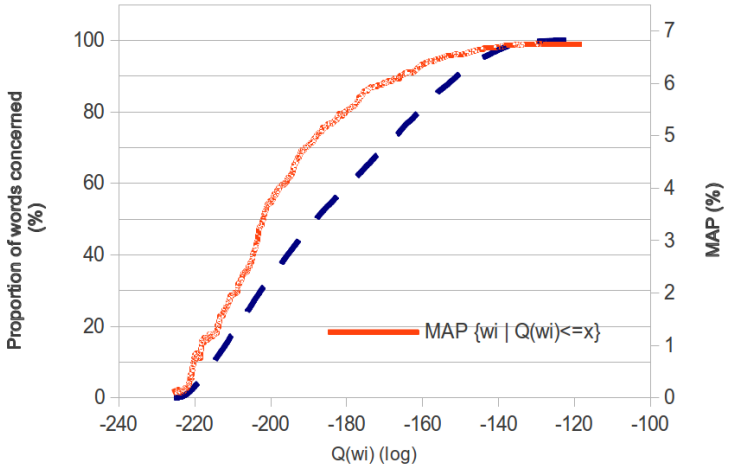


Figure 3: Average MAP computed on words with a confidence score lower than a threshold q (x-axis, log-scale), and cumulative proportion of concerned words.

value of 1 indicates a perfect correlation, 0 no correlation and -1 an inverse correlation. A low p-value, for example < 0.05, indicates a statistically significant result. The confidence scores are obtained with $k = 20$. Other experiments, not reported here, show that this parameter k has little influence on the correlation, for values between 5 and 100.

These measures attest to some statistically significant correlation between our confidence score and the MAP, however this correlation is imperfect and non-linear. We compute the average MAP on neighbor lists with a confidence score lower than a threshold q. Figure 3 represents the average MAP (y-axis) in function of the threshold q (x-axis). It shows that the confidence score is still a good indicator of quality, as the MAP decreases with the confidence score.

The confidence score can be used to improve the performance of aggregation techniques presented in Section 4 by integrating it in the final score:

$$\text{score}_{w_i}(w_j) = Q(w_j) * \text{Aggreg}(1 - \tau_{w_i}(w_j)/n, 1 - \tau_{w_j}(w_i)/n) \quad (6)$$

As shown in Table 4, using this information allows even greater gains than those reported in the previous section (a Wilcoxon test ($p < 0.05$) (Hull, 1993) is performed to ensure that the differences are statistically significant; non-significant ones are shown in italics). In the next section, we propose another use of the confidence scores to improve results more specifically on the head of the lists, that is to say on the neighbors judged closest.

Method	MAP	R-Prec	P@1	P@5	P@10	P@50	P@100
S_{Hamacher} $\gamma = 0.95$	9.61 (+7.20)	11.59 (+5.85)	*30.86 (-0.53)*	19.52 (+5.83)	14.76 (+7.24)	7.03 (+8.88)	4.93 (+8.67)

Table 4: Performance gains (%) by reciprocal rank aggregation using the confidence score, on WordNet+Moby reference.

Target	MAP	R-Prec	P@1	P@5	P@10
all words	9.16 (+2.17)	11.24 (+2.76)	*30.73 (-1.02)*	19.30 (+4.64)	14.37 (+4.44)
the third of words with the lowest $Q(w)$	9.55 (+6.44)	11.81 (+7.99)	31.85 (+2.56)	20.43 (+10.81)	15.46 (+12.37)

Table 5: Performance gains (%) of reranking with the Hungarian algorithm.

6 Local reranking

The previous method gives an overall score to the list, but one can also make use of the individual ranking probabilities $p(\delta(w_i, w_j))$, estimated according to the method of Parzen windows. For a given word w, we have for each of its neighbors w_j the probability of his current rank: $p(\delta(w, w_j)) = \tau_w(w_j)$. For a given neighbor w_j, we can also calculate the probability of any other rank τ: $p(\delta(w, w_j)) = \tau$ with $\tau = 1, 2, ...$ In this section, we propose to rely on these more local information to improve the performance by reranking the k-nearest neighbors.

6.1 Reranking by the Hungarian algorithm

A simple approach would be to reorder the list based on this criterion, from the most probable neighbors to the least ones. But ranking probability estimation for each word is imperfect, and such a reranking strongly degrades the results. We therefore propose instead a method to rerank the k-nearest neighbors on a more local and controlled manner: a word that was not originally in the k-nearest neighbors can not become a k-nearest neighbor, and a word can not be reranked too far from its original rank.

Our problem is expressed by the following matrix $\mathcal{M}_{\text{profit}}$. The rows correspond to words in their original ranks (denoted w_1 to w_k), the columns to new ranks τ at which these words can be assigned, and matrix values are the probabilities of each word w_j to appear at rank τ. Given these probabilities, the goal is to find the most likely permutation of the k-nearest neighbors.

$$\mathcal{M}_{\text{profit}} = \begin{pmatrix} p(\delta(w, w_1) = 1) & \cdots & p(\delta(w, w_1) = k) \\ \vdots & \ddots & \vdots \\ p(\delta(w, w_k) = 1) & \cdots & p(\delta(w, w_k) = k) \end{pmatrix}$$

As pointed out, we want to avoid that an initially very close neighbor was moved far away and vice versa. This constraint is added by multiplying the matrix $\mathcal{M}_{\text{profit}}$ by a penalty matrix $\mathcal{M}_{\text{penalty}}$ (see below) with the Hadamard product (element by element matrix product, denoted $\circ$).

$$\mathcal{M}_{\text{penalty}} = \begin{pmatrix} 1 & \frac{k-1}{k} & \cdots & 0 \\ \frac{k-1}{k} & 1 & \cdots & \frac{1}{k} \\ \vdots & \ddots & \vdots & \\ 0 & \frac{1}{k} & \cdots & 1 \end{pmatrix}$$

We then face a combinatorial optimization problem which can be solved in polynomial time by the Hungarian method (Kuhn and Yaw, 1955, for a description of the algorithm) on the matrix of assignment costs $\mathcal{M}_{\text{profit}} \circ \mathcal{M}_{\text{penalty}}$. This algorithm was originally proposed to optimize the assignment of workers (in our case, the neighbors) on tasks (in our case, ranks), according to the profit generated by each worker for each task (in our case, the probability that a neighbor stands at a given rank). The result of this algorithm therefore indicates a new rank for each word. The algorithm converges to an optimal solution with a complexity $\mathcal{O}(k^3)$ (for reranking the k-nearest neighbors).

6.2 Results

Table 5 presents the performance achieved by our local reranking method compared to the adjusted Okapi-BM25 reference using the same experimental conditions as above. As before, the considered neighborhood is set to $k = 20$. Precisions beyond this threshold are unchanged and thus not reported. We test the effectiveness of the local reranking on all neighbor lists and on a third of lists with the lowest quality scores.

It appears that the reranking on the whole lists does not provide a real gain. However, the gain is substantial on the lists with low confidence score. Moreover, unlike the experiments of section 4, these

gains apply by construction to the heads of lists, which are most likely to be used in practice. This difference between results on the whole set of words and on those with the lowest confidence scores can be explained in two ways. First, the lists with the highest confidence scores correspond largely to the lists with the best MAP, as expected and illustrated in Figure 3. This therefore suggests a priori little room for improvement. Second, regardless of MAP, we can also assume that these lists already have an optimum arrangement of individual probabilities that explains the high confidence score. The reranking thus concerns only few neighbors.

7 Conclusion and future work

The different contributions proposed in this article do not place themselves all at the same level. The thesaurus construction using tools from the IR is not a major conceptual innovation, but this approach seems curiously unexplored although it provides very competitive results while requiring minimum implementations through existing tools from IR.

The various propositions exploiting the neighborhood graph to improve the thesaurus are part of a more original approach where the whole thesaurus is considered. We have specifically examined the aspects of reciprocity and distance, in terms of rank, between two words to offer several contributions. The improvements obtained by aggregation over all neighbors or by the local reranking from confidence scores validate our approach. It should be noted that the gains are small in absolute terms, but, compared to those observed in the field, correspond to significant improvements.

The various aspects of this work open up many prospects of research. For example, many other aggregate functions in addition to those tested in section 4 exist in the literature. Some may even offer the possibility of integrating the confidence score associated with each neighbor, as Choquet's or Sugeno's integrals (Detyniecki, 2000). More generally, it would be interesting to iteratively use improvements of neighbor lists to update the confidence scores, etc., in the spirit for example of what is proposed by Pedronette et al. (2014). A detailled analysis of the impact of these techniques according to the type of semantic relation is still to be performed. Beyond the distributional thesauri construction, the proposed methods to compute confidence scores or reorder lists of neighbors can be applied to other problems where the k-nearest neighbor graphs of are built. Also note that we have only considered a small part of the information carried by the neighborhood graph. We focused on the aspects of reciprocity, but taking into account other aspects of the graph (in particular the transitivity, or more generally its topology), could lead to further improvements.

References

Clémentine Adam, Cécile Fabre, and Philippe Muller. 2013. Évaluer et améliorer une ressource distributionnelle : protocole d'annotation de liens sémantiques en contexte. *TAL*, 54(1):71–97.

Bartosz Broda, Maciej Piasecki, and Stan Szpakowicz. 2009. Rank-Based Transformation in Measuring Semantic Relatedness. In *22nd Canadian Conference on Artificial Intelligence*, pages 187–190.

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.

James R. Curran and Marc Moens. 2002. Improvements in automatic thesaurus extraction. In *Workshop of the ACL Special Interest Group on the Lexicon (SIGLEX)*, pages 59–66, Philadelphia, USA.

Marcin Detyniecki. 2000. *Mathematical aggregation operators and their application to video querying*. Ph.D. thesis, Université de Paris 6.

Dominique Domengès and Michel Volle. 1979. Analyse factorielle sphérique : une exploration. *Annales de l'INSEE*, 35:3–83.

Bernard Escoffier. 1978. Analyse factorielle et distances répondant au principe d'équivalence distributionnelle. *Revue de statistique appliquée*, 26(4):29–37.

Olivier Ferret. 2012. Combining bootstrapping and feature selection for improving a distributional thesaurus. In *20th European Conference on Artificial Intelligence (ECAI 2012)*, pages 336–341, Montpellier, France.

Olivier Ferret. 2013. Identifying bad semantic neighbors for improving distributional thesauri. In *51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 561–571, Sofia, Bulgaria.

Olivier Ferret. 2014 (in press). Typing relations in distributional thesauri. In N. Gala, R. Rapp, and G. Bel, editors, *Advances in Language Production, Cognition and the Lexicon*. Springer.

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

Dayne Freitag, Matthias Blume, John Byrnes, Edmond Chow, Sadik Kapadia, Richard Rohwer, and Zhiqiang Wang. 2005. New experiments in distributional representations of synonymy. In *Ninth Conference on Computational Natural Language Learning (CoNLL)*, pages 25–32, Ann Arbor, Michigan, USA.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 6–12.

Gregory Grefenstette. 1994. *Explorations in automatic thesaurus discovery*. Kluwer Academic Publishers.

Masato Hagiwara, Yasuhiro Ogawa, and Katsuhiko Toyama. 2006. Selection of effective contextual information for automatic synonym acquisition. In *21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006)*, pages 353–360, Sydney, Australia.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *50th Annual Meeting of the Association for Computational Linguistics (ACL'12)*, pages 873–882.

David Hull. 1993. Using Statistical Testing in the Evaluation of Retrieval Experiments. In *Proc. of the 16th Annual ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'93*, Pittsburgh, États-Unis.

Harold W. Kuhn and Bryn Yaw. 1955. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211–240.

Dekang Lin. 1998a. Automatic retrieval and clustering of similar words. In *17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics (ACL-COLING'98)*, pages 768–774, Montréal, Canada.

Dekang Lin. 1998b. Automatic retrieval and clustering of similar words. In *17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics (ACL-COLING'98)*, pages 768–774, Montréal, Canada.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, pages 746–751, Atlanta, Georgia.

George A. Miller. 1990. WordNet: An On-Line Lexical Database. *International Journal of Lexicography*, 3(4).

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

Emanuel Parzen. 1962. On estimation of a probability density function and mode. *Ann. Math. Stat.*, 33:1065–1076.

Daniel Carlos Guimarães Pedronette, Otávio Augusto Bizetto Penatti, and Ricardo da Silva Torres. 2014. Unsupervised manifold learning using reciprocal knn graphs in image re-ranking and rank aggregation tasks. *Image Vision Computing*, 32(2):120–130.

Stephen E. Robertson, Steve Walker, and Micheline Hancock-Beaulieu. 1998. Okapi at TREC-7: Automatic Ad Hoc, Filtering, VLC and Interactive. In *Proc. of the 7th Text Retrieval Conference, TREC-7*, pages 199–210.

Magnus Sahlgren. 2001. Vector-based semantic analysis: Representing word meanings based on random labels. In *ESSLLI 2001 Workshop on Semantic Knowledge Acquisition and Categorisation*, Helsinki, Finland.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*.

Bernard W. Silverman. 1986. *Density estimation for statistics and data analysis*. Monographs on statistics and applied probability. Chapman and Hall Boca Raton, London, Glasgow, Weinheim.

Tim Van de Cruys. 2010. *Mining for Meaning. The Extraction of Lexico-semantic Knowledge from Text*. Ph.D. thesis, University of Groningen, The Netherlands.

Olga Vechtomova and Stephen E. Robertson. 2012. A domain-independent approach to finding related entities. *Information Processing and Management*, 48(4):654–670.

Grady Ward. 1996. Moby thesaurus. Moby Project.

Larry Wasserman. 2005. *All of Statistics: A Concise Course in Statistical Inference*. Springer Texts in Statistics.

Kazuhide Yamamoto and Takeshi Asakura. 2010. Even unassociated features can improve lexical distributional similarity. In *Second Workshop on NLP Challenges in the Information Explosion Era (NLPIX 2010)*, pages 32–39, Beijing, China.

Maayan Zhitomirsky-Geffet and Ido Dagan. 2009. Bootstrapping Distributional Feature Vector Quality. *Computational Linguistics*, 35(3):435–461.

Towards Syntax-aware Compositional Distributional Semantic Models

Lorenzo Ferrone
Department of Enterprise Engineering
University of Rome "Tor Vergata"
Via del Politecnico, 1 00173 Roma
lorenzo.ferrone@gmail.com

Fabio Massimo Zanzotto
Department of Enterprise Engineering
University of Rome "Tor Vergata"
Via del Politecnico, 1 00173 Roma
fabio.massimo.zanzotto@uniroma2.it

Abstract

Compositional Distributional Semantics Models (CDSMs) are traditionally seen as an entire different world with respect to Tree Kernels (TKs). In this paper, we show that under a suitable regime these two approaches can be regarded as the same and, thus, structural information and distributional semantics can successfully cooperate in CSDMs for NLP tasks. Leveraging on distributed trees, we present a novel class of CDSMs that encode both structure and distributional meaning: the distributed smoothed trees (DSTs). By using DSTs to compute the similarity among sentences, we implicitly define the distributed smoothed tree kernels (DSTKs). Experiment with our DSTs show that DSTKs approximate the corresponding smoothed tree kernels (STKs). Thus, DSTs encode both structural and distributional semantics of text fragments as STKs do. Experiments on RTE and STS show that distributional semantics encoded in DSTKs increase performance over structure-only kernels.

1 Introduction

Compositional distributional semantics is a flourishing research area that leverages distributional semantics (see Turney and Pantel (2010), Baroni and Lenci (2010)) to produce meaning of simple phrases and full sentences (hereafter called *text fragments*). The aim is to scale up the success of word-level relatedness detection to longer fragments of text. Determining similarity or relatedness among sentences is useful for many applications, such as multi-document summarization, recognizing textual entailment (Dagan et al., 2013), and semantic textual similarity detection (Agirre et al., 2013).

Compositional distributional semantics models (CDSMs) are functions mapping text fragments to vectors (or higher-order tensors). Functions for simple phrases directly map distributional vectors of words to distributional vectors for the phrases (Mitchell and Lapata, 2008; Baroni and Zamparelli, 2010; Clark et al., 2008; Grefenstette and Sadrzadeh, 2011; Zanzotto et al., 2010). Functions for full sentences are generally defined as recursive functions over the ones for phrases (Socher et al., 2011; Socher et al., 2012; Kalchbrenner and Blunsom, 2013). Distributional vectors for text fragments are then used as inner layers in neural networks, or to compute similarity among text fragments via dot product.

CDSMs generally exploit structured representations t^x of text fragments x to derive their meaning $f(t^x)$, but the structural information, although extremely important, is obfuscated in the final vectors. Structure and meaning can interact in unexpected ways when computing cosine similarity (or dot product) between vectors of two text fragments, as shown for full additive models in (Ferrone and Zanzotto, 2013). Smoothed tree kernels (STK) (Mehdad et al., 2010; Croce et al., 2011) instead realize a clearer interaction between structural information and distributional meaning. STKs are specific realizations of convolution kernels (Haussler, 1999) where the similarity function is recursively (and, thus, compositionally) computed. Distributional vectors are used to represent word meaning in computing the similarity among nodes. STKs, however, are not considered part of the CDSMs family. As usual in kernel machines

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 721–730, Dublin, Ireland, August 23-29 2014.

(Cristianini and Shawe-Taylor, 2000), STKs directly compute the similarity between two text fragments x and y over their tree representations t^x and t^y, that is, $STK(t^x, t^y)$. The function f that maps trees into vectors is only implicitly used, and, thus, $STK(t^x, t^y)$ is not explicitly expressed as the dot product or the cosine between $f(t^x)$ and $f(t^y)$. Such a function f, which is the underlying reproducing function of the kernel (Aronszajn, 1950), is a CDSM since it maps trees to vectors by using distributional meaning. However, the huge dimensionality of $\mathbb{R}^n$ (since it has to represent the set of all possible subtrees) prevents to actually compute the function $f(t)$, which thus can only remain *implicit*.

Distributed tree kernels (DTK) (Zanzotto and Dell'Arciprete, 2012) partially solve the last problem. DTKs approximate standard tree kernels (such as (Collins and Duffy, 2002)) by defining an *explicit* function DT that maps trees to vectors in $\mathbb{R}^m$ where $m \ll n$ and $\mathbb{R}^n$ is the explicit space for tree kernels. DTKs approximate standard tree kernels (TK), that is, $\langle DT(t^x), DT(t^y)\rangle \approx TK(t^x, t^y)$, by approximating the corresponding reproducing function (Aronszajn, 1950). Thus, these distributed trees are small vectors that encode structural information. In DTKs tree nodes u and v (and then also words) are represented by nearly orthonormal vectors, that is, vectors $\vec{u}$ and $\vec{v}$ such that $\langle \vec{u}, \vec{v}\rangle \approx \delta(\vec{u}, \vec{v})$ where δ is the Kroneker's delta. This is in contrast with distributional semantics vectors where $\langle \vec{u}, \vec{v}\rangle$ is allowed to be any value in $[0, 1]$ according to the similarity between the words v and u. Thus, early attempts to include distributional vectors in the DTs failed (Zanzotto and Dell'Arciprete, 2011).

In this paper, leveraging on distributed trees, we present a novel class of CDSMs that encode both structure and distributional meaning: the distributed smoothed trees (DST). DSTs carry structure and distributional meaning on a 2-dimensional tensor (a matrix): one dimension encodes the structure and one dimension encodes the meaning. By using DSTs to compute the similarity among sentences with a generalized dot product (or cosine), we implicitly define the distributed smoothed tree kernels (DSTK) which approximate the corresponding STKs. We present two DSTs along with the two smoothed tree kernels (STKs) that they approximate. We experiment with our DSTs to show that their generalized dot products approximate STKs by directly comparing the produced similarities and by comparing their performances on two tasks: recognizing textual entailment (RTE) and semantic similarity detection (STS). Both experiments show that the dot product on DSTs approximates STKs and, thus, DSTs encode both structural and distributional semantics of text fragments in tractable 2-dimensional tensors. Experiments on STS and RTE show that distributional semantics encoded in DSTs increases performance over structure-only kernels. DSTs are the first positive way of taking into account both structure and distributional meaning in CDSMs.

The rest of the paper is organized as follows. Section 2 introduces the basic notation used in the paper. Section 3 describe our distributed smoothed trees as compositional distributional semantic models that can represent both structural and semantic information. Section 4 reports on the experiments. Finally, Section 5 draws some conclusions.

2 Notation

Before describing the *distributed smoothed trees* (DST) we introduce a formal way to denote constituency-based *lexicalized parse trees*, as DSTs exploit this kind of data structures.

Lexicalized trees are denoted with the letter t and $N(t)$ denotes the set of non terminal nodes of tree t. Each non-terminal node $n \in N(t)$ has a label l_n composed of two parts $l_n = (s_n, w_n)$: s_n is the syntactic label, while w_n is the semantic headword of the tree headed by n, along with its part-of-speech tag. For example, the root node of the tree in Fig.1 has the label S:*booked::v* where S is the syntactic information and *booked::v* is the semantic head of the whole tree. Terminal nodes of trees are treated differently, these nodes represent only words w_n without any additional information, and their labels thus only consist of the word itself (see Fig. 1). The structure of a tree is represented as follows: Given a tree t, $\mathsf{h}(t)$ is its root node and $\mathsf{s}(t)$ is the tree formed from t but considering only the syntactic structure (that is, only the s_n part of the labels), $c_i(n)$ denotes i-th child of a node n. As usual for constituency-based parse trees, pre-terminal nodes are nodes that have a single terminal node as child. Finally, $\vec{s_n} \in \mathbb{R}^m$ and $\vec{w_n} \in \mathbb{R}^k$ represent respectively *distributed* vectors for node labels s_n and *distributional* vectors for words w_n, whereas $\mathbf{T}$ represents the matrix of a tree t encoding structure and distributional meaning. The difference between distributed and distributional vectors is described in the next section.

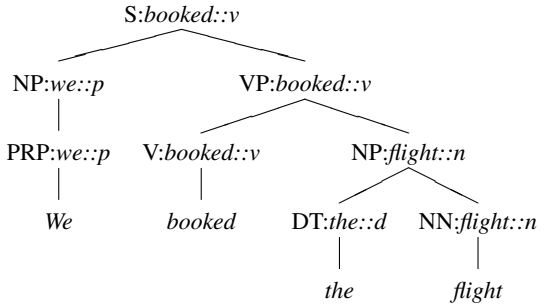


Figure 1: A lexicalized trees

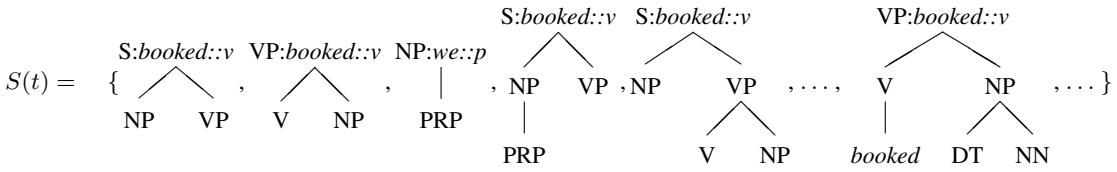


Figure 2: Subtrees of the tree t in Figure 1 (a non-exhaustive list)

3 Distributed Smoothed Trees as Compositional Distributional Semantic Models

We define Distributed Smoothed Trees as recursive functions DST mapping lexicalized trees t to $\mathbb{R}^{m \times k}$ where matrices $\mathbf{T} = DST(t)$ encode both syntactic structures and distributional vectors. DSTs are thus compositional distributional models, as they map lexicalized trees to matrices, and they are defined recursively on distributed vectors for syntactic node labels and distributional vectors for words. In the following we introduce DSTs: Section 3.1 gives a rough idea of the method, Section 3.2 describes how to recursively encode structures in vectors by means of distributed trees (Zanzotto and Dell'Arciprete, 2012), and finally Section 3.3 merges distributed trees and distributional semantic vectors in matrices.

3.1 The method in a glance

We describe here the approach in a few sentences. In line with tree kernels over structures (Collins and Duffy, 2002), we introduce the set $S(t)$ of the subtrees t_i of a given lexicalized tree t. A subtree t_i is in the set $S(t)$ if $\mathsf{s}(t_i)$ is a subtree of $\mathsf{s}(t)$ and, if n is a node in t_i, all the siblings of n in t are in t_i. For each node of t_i we only consider its syntactic label s_n, except for the head $\mathsf{h}(t_i)$ for which we also consider its semantic component w_n. Figure 2 reports a sample for the subtrees of the tree in Fig. 1 The recursive functions DSTs we define compute the following:

$$\mathbf{T} = \sum_{t_i \in S(t)} \mathbf{T}_i$$

where $\mathbf{T}_i$ is the matrix associated to each subtree t_i. The similarity between two text fragments a and b represented as lexicalized trees t^a and t^b can be computed using the Frobenius product between the two matrices $\mathbf{T}^a$ and $\mathbf{T}^b$, that is:

$$\langle \mathbf{T}^a, \mathbf{T}^b \rangle_F = \sum_{\substack{t_i^a \in S(t^a) \\ t_j^b \in S(t^b)}} \langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \quad (1)$$

We want to obtain that the product $\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F$ approximates the dot product between the distributional vectors of the head words ($\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \approx \langle \mathsf{h}(\overrightarrow{t_i^a}), \mathsf{h}(\overrightarrow{t_j^b}) \rangle$) whenever the syntactic structure of the subtrees is the same (that is $\mathsf{s}(t_i^a) = \mathsf{s}(t_j^b)$), and $\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \approx 0$ otherwise. This property is expressed as:

$$\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \approx \delta(\mathsf{s}(t_i^a), \mathsf{s}(t_j^b)) \cdot \langle \mathsf{h}(\overrightarrow{t_i^a}), \mathsf{h}(\overrightarrow{t_j^b}) \rangle \quad (2)$$

3.2 Representing Syntactic Structures with Distributed Trees

Distributed trees (Zanzotto and Dell'Arciprete, 2012) recursively encode syntactic trees t in small vectors by means of a recursive function DT. These DTs preserve structural information as the dot product between the DTs of two trees approximates the classical tree kernels TK as defined by Collins and Duffy (2002), that is, $TK(t^a, t^b) \approx \langle DT(t^a), DT(t^b) \rangle$. To obtain this result, distributed trees $DT(t)$ are defined as follows:

$$DT(t) = \sum_{t_i \in S(t)} \sqrt{\lambda^{|N(t_i)|}} \; \vec{\mathsf{s}}(t_i) \tag{3}$$

where $S(t)$ is again the set of the subtrees of t, $\vec{\mathsf{s}}(t_i)$ are vectors in $\mathbb{R}^m$ corresponding to tree fragment t_i and $\sqrt{\lambda^{|N(t_i)|}}$ is the weight of subtree t_i in the final feature space, with λ being the traditional parameter used to penalize large subtrees and $|N(t_i)|$ being the number of nodes in t_i. The approximation of tree kernels is then given by the fact that $\langle \vec{\mathsf{s}}(t_i), \vec{\mathsf{s}}(t_j) \rangle \approx \delta(\mathsf{s}(t_i), \mathsf{s}(t_j))$. Vectors with this property are called *distributed vectors*. A key feature of the distributed vectors of subtrees $\vec{\mathsf{s}}(t_i)$ is that these vectors are built compositionally from a set $\mathcal{N}$ of *nearly orthonormal random vectors* $\vec{s_n}$, that are associated to each node label. Given a subtree $\mathsf{s}(t_i)$, the related vector is obtained as:

$$\vec{\mathsf{s}}(t_i) = \vec{s_{n_1}} \odot \vec{s_{n_2}} \odot \ldots \odot \vec{s_{n_k}} = \bigodot_{(s_n, w_n) \in N(t_i)} \vec{s_n}$$

where node vectors $\vec{s_{n_i}}$ are ordered according to a depth-first visit of subtree t_i and $\odot$ is a vector composition operation, specifically the *shuffled circular convolution*[1]. This function guarantees that two different subtrees have nearly orthonormal vectors (see (Zanzotto and Dell'Arciprete, 2012) for more details). For example, the fifth tree t_5 of set $S(t)$ in Figure 2 is $\vec{\mathsf{s}}(t_5) = \vec{S} \odot (\vec{NP} \odot (\vec{VP} \odot (\vec{V} \odot \vec{NP})))$. Thus, DTs in Equation 3 can be recursively defined as:

$$DT(t) = \sum_{n \in N(t)} \sigma(n) \tag{4}$$

where $\sigma(n)$ is recursively defined as follows:

$$\sigma(n) = \begin{cases} \sqrt{\lambda}\,(\vec{s_n} \odot \vec{w}) & \text{if } n \text{ is a pre-terminal node} \\ \sqrt{\lambda}\,\vec{s_n} \odot (\bigodot_i (\vec{s_{c_i(n)}} + \sigma(c_i(n)))) & \text{if } n \text{ is an internal node} \end{cases} \tag{5}$$

The vector $\sigma(n)$ encodes all the subtrees that have root in n along with their penalizing weight $\sqrt{\lambda^{|N(t_i)|}}$, that is:

$$\sigma(n) = \sum_{t_i \in S(t) \wedge \mathsf{h}(t_i) = n} \sqrt{\lambda^{|N(t_i)|}} \; \vec{\mathsf{s}}(t_i)$$

This is what we need in order to define our *distributed smoothed trees*.

3.3 Representing distributional meaning and distributed structure with matrices

We now move from distributed trees (encoded as small vectors) to distributed smoothed trees (DST) represented as matrices. DST is a function that maps trees t to matrices $\mathbf{T}$. In analogy with Equation 4, DST is defined as:

$$DST(t) = \sum_{n \in N(t)} S(n)$$

where $S(n)$ is now defined as:

$$S(n) = \sigma(n) \vec{w_n}^{\top}$$

[1] The *shuffled circular convolution* $\odot$ is defined as $\vec{a} \odot \vec{b} = s_1(\vec{a}) * s_2(\vec{b})$ where $*$ is the circular convolution and s_1 and s_2 are two different (but fixed) random permutations of vector elements.

where $\sigma(n)$ is the one defined in Equation 5 and $(\cdot)^\top$ is vector transposition. By combining the two equations, $DST(t)$ is the sum of the matrices described in Equation 1:

$$DST(t) = \sum_{n \in N(t)} \sum_{t_i \in S(t) \wedge \mathsf{h}(t_i)=n} \sqrt{\lambda^{|N(t_i)|}}\ \vec{\mathsf{s}(t_i)} \vec{w_n}^\top = \sum_{t_i \in S(t)} \vec{\mathsf{s}(t_i)} \vec{w_n}^\top$$

where n is $\mathsf{h}(t_i)$ and $\mathbf{T}_i = \vec{\mathsf{s}(t_i)} \vec{w_{\mathsf{h}(t_i)}}^\top$ is the outer product between the distributed vector $\vec{\mathsf{s}(t_i)}$ and the distributional vector $\vec{w_{\mathsf{h}(t_i)}}$. There is an important property of the outer product that applies to the Frobenius product: $\langle \vec{a}\vec{w}^\top, \vec{b}\vec{v}^\top \rangle_F = \langle \vec{a}, \vec{b} \rangle \cdot \langle \vec{w}, \vec{v} \rangle$. Using this property, we have that Equation 2 is satisfied as:

$$\langle \mathbf{T}_i, \mathbf{T}_j \rangle_F = \langle \vec{\mathsf{s}(t_i)}, \vec{\mathsf{s}(t_j)} \rangle \cdot \langle \vec{w_{h(t_i)}}, \vec{w_{h(t_j)}} \rangle \approx \delta(\mathsf{s}(t_i), \mathsf{s}(t_j)) \cdot \langle \vec{w_{h(t_i)}}, \vec{w_{h(t_j)}} \rangle$$

We refer to the Frobenius product of two distributed smoothed trees as *distributed smoothed tree kernel* (DSTK). These DSTKs are approximating the smoothed tree kernels described in the next section. We propose two versions of our DSTKs according to how we produce distributional vectors for words. We have a plain version $DSTK_0$ when we use distributional vectors $\vec{w_n}$ as they are, and a slightly modified version $DSTK_{+1}$ when we use as distributional vectors $\vec{w_n}' = \begin{pmatrix} 1 & \vec{w_n} \end{pmatrix}$.

3.4 The Approximated Smoothed Tree Kernels

The two CDSMs we proposed, that is, the two distributed smoothed tree kernels $DSTK_0$ and $DSTK_{+1}$, are approximating two specific tree kernels belonging to the smoothed tree kernels class (e.g., (Mehdad et al., 2010; Croce et al., 2011)). These two specific smoothed tree kernels recursively compute (but, the recursive formulation is not given here) the following general equation:

$$STK(t^a, t^b) = \sum_{\substack{t_i \in S(t^a) \\ t_j \in S(t^b)}} \omega(t_i, t_j)$$

where $\omega(t_i, t_j)$ is the similarity weight between two subtrees t_i and t_j. $DTSK_0$ and $DSTK_{+1}$ approximate respectively STK_0 and STK_{+1} where the weights are defined as follows:

$$\omega_0(t_i, t_j) = \langle \vec{w_{\mathsf{h}(t_i)}}, \vec{w_{\mathsf{h}(t_j)}} \rangle \cdot \delta(\mathsf{s}(t_i), \mathsf{s}(t_j)) \cdot \sqrt{\lambda^{|N(t_i)|+|N(t_j)|}}$$

$$\omega_{+1}(t_i, t_j) = (\langle \vec{w_{\mathsf{h}(t_i)}}, \vec{w_{\mathsf{h}(t_j)}} \rangle + 1) \cdot \delta(\mathsf{s}(t_i), \mathsf{s}(t_j)) \cdot \sqrt{\lambda^{|N(t_i)|+|N(t_j)|}}$$

STK_{+1} is actually computing a sum between STK_0 and the tree kernel (Collins and Duffy, 2002).

4 Experimental investigation

4.1 Experimental set-up

Generic settings We experimented with two datasets: the Recognizing Textual Entailment datasets (RTE) (Dagan et al., 2006) and the the Semantic Textual Similarity 2013 datasets (STS) (Agirre et al., 2013). The STS task consists of determining the degree of similarity (ranging from 0 to 5) between two sentences. We used the data for core task of the 2013 challenge data. The STS datasets contains 5 datasets: headlines, OnWN, FNWN, SMT and MSRpar, which contains respectively 750, 561, 189, 750 and 1500 pairs. The first four datasets were used for testing, while all the training has been done on the fifth. RTE is instead the task of deciding whether a long text T entails a shorter text, typically a single sentence, called hypothesis H. It has been often seen as a classification task (see (Dagan et al., 2013)). We used four datasets: RTE1, RTE2, RTE3, and RTE5, with the standard split between training and testing. The dev/test distribution for RTE1-3, and RTE5 is respectively 567/800, 800/800, 800/800, and 600/600 T-H pairs. Distributional vectors are derived with DISSECT (Dinu et al., 2013) from a corpus obtained by the concatenation of ukWaC (wacky.sslmit.unibo.it), a mid-2009 dump of

		RTE1	RTE2	RTE3	RTE5	headl	FNWN	OnWN	SMT
STK_0 vs $DSTK_0$	1024	0.86	0.84	0.90	0.84	0.87	0.65	0.95	0.77
	2048	0.87	0.84	0.91	0.84	0.90	0.65	0.96	0.77
STK_{+1} vs $DSTK_{+1}$	1024	0.81	0.77	0.83	0.72	0.88	0.53	0.93	0.66
	2048	0.82	0.78	0.84	0.74	0.91	0.56	0.94	0.67

Table 1: Spearman's correlation between Distributed Smoothed Tree Kernels and Smoothed Tree Kernels

the English Wikipedia (en.wikipedia.org) and the British National Corpus (www.natcorp.ox.ac.uk), for a total of about 2.8 billion words. We collected a 35K-by-35K matrix by counting co-occurrence of the 30K most frequent content lemmas in the corpus (nouns, adjectives and verbs) and all the content lemmas occurring in the datasets within a 3 word window. The raw count vectors were transformed into positive Pointwise Mutual Information scores and reduced to 300 dimensions by Singular Value Decomposition. This setup was picked without tuning, as we found it effective in previous, unrelated experiments. To build our DTSKs and for the two baseline kernels TK and DTK, we used the implementation of the distributed tree kernels[2]. We used: 1024 and 2048 as the dimension of the distributed vectors, the weight λ is set to 0.4 as it is a value generally considered optimal for many applications (see also (Zanzotto and Dell'Arciprete, 2012)). The statistical significance, where reported, is computed according to the sign test.

Direct correlation settings For the *direct correlation* experiments, we used the RTE data sets and the testing sets of the STS dataset (that is, *headlines*, *OnWN*, *FNWN*, *SMT*). We computed the Spearman's correlation between values produced by our $DSTK_0$ and $DSTK_{+1}$ and produced by the standard versions of the smoothed tree kernel, that is, respectively, STK_0 and STK_{+1}. We obtained text fragment pairs by randomly sampling two text fragments in the selected set. For each set, we produced exactly the number of examples in the set, e.g., we produced 567 pairs for RTE1 dev, etc..

Task-based settings For the *task-based* experiments, we compared systems using the standard evaluation measure and the standard split in the respective challenges. As usual in RTE challenges the measure used is the accuracy, as testing sets have the same number of entailment and non-entailment pairs. For STS, we used MSRpar as training, and we used the 4 test sets as testing. We compared systems using the Pearson's correlation as the standard evaluation measure for the challenge[3]. Thus, results can be compared with the results of the challenge.

As classifier and regression learner, we used the java version of LIBSVM (Chang and Lin, 2011). In the two tasks we used in a different way our DSTs (and the related STKs) within the learners. In the following, we refer to instances in RTE or STS as pairs $p = (t^a, t^b)$ where t^a and t^b are the two parse trees for the two sentences a and b for STS and for the text a and the hypothesis b in RTE.

We will indicate with $K(p_1, p_2)$ the final kernel used in the learning algorithm, which takes as input two training instances, while we will use κ to denote either any of our DSTK (that is, $\kappa(x, y) = \langle DST(x), DST(y)\rangle$) or any of the standard smoothed tree kernels (that is, $\kappa(x, y) = STK(x, y)$).

In STS, we encoded only similarity feature between the two sentences. Thus, we used two classes of kernels: (1) the syntactic/semantic class (SS) with the final kernel defined as $K(p_1, p_2) = (\kappa(t^a_1, t^b_1) \cdot \kappa(t^a_2, t^b_2) + 1)^2$; and, (2) the SS class along with token-based similarity (SSTS) where the final kernel is $K(p_1, p_2) = (\kappa(t^a_1, t^b_1) \cdot \kappa(t^a_2, t^b_2) + TS(a_1, b_1) \cdot TS(a_2, b_2) + 1)^2$ where $TS(a, b)$ counts the percent of the common content tokens in a and b.

In RTE, we followed standard approaches (Dagan et al., 2013; Zanzotto et al., 2009), that is, we exploited two models: a model with only a rewrite rule feature space (RR) and a model with the previous space along with a token-level similarity feature (RRTWS). The two models use our DSTs and the standard STKs in the following way as kernel functions: (1) $RR(p_1, p_2) = \kappa(t^a_1, t^a_2) + \kappa(t^b_1, t^b_2)$; (2) $RRTS(p_1, p_2) = \kappa(t^a_1, t^a_2) + \kappa(t^b_1, t^b_2) + (TWS(a_1, b_1) \cdot TS(a_2, b_2) + 1)^2$ where TWS is a weighted token similarity as in Corley and Mihalcea (2005).

[2] http://code.google.com/p/distributed-tree-kernels/

[3] Correlations are obtained with the organizers' script

	SS					SSTS				
	headl	**FNWN**	**OnWN**	**SMT**	**Average**	**headl**	**FNWN**	**OnWN**	**SMT**	**Average**
TS	—	—	—	—	—	0.701	0.311	0.515	0.323	0.462
Add	—	—	—	—	—	0.691	0.268	0.511	0.317	0.446
Mult	—	—	—	—	—	0.291	−0.03	0.228	0.291	0.201
DTK	0.448	0.118	0.162	0.301	0.257	0.698	0.311	0.510	0.329	0.462
TK	0.456	0.145	0.158	0.303	0.265*	0.699	0.316	0.511	0.329	0.463*
$DSTK_0$	0.491	0.155	0.358	0.305	**0.327**†	0.700	0.314	0.519	0.327	0.465
STK_0	0.490	0.159	0.349	0.305	0.325*	0.700	0.314	0.519	0.327	0.465*
$DSTK_{+1}$	0.475	0.138	0.266	0.304	0.295	0.700	0.314	0.519	0.327	**0.465**
STK_{+1}	0.478	0.156	0.259	0.305	0.299*	0.700	0.314	0.519	0.327	0.465*

Table 2: Task-based analysis: Correlation on Semantic Textual Similarity († is different from DTK, TK, $DSTK_{+1}$, and STK_{+1} with a stat.sig. of $p > 0.1$; $*$ the difference between the kernel and its distributed version is not stat.sig.)

We also used two standard and simple CDSMs to compare with: the Additive model (Add) and the Multiplicative model (Mult) as firstly discussed in Mitchell and Lapata (2008). The Additive Model performs a sum of all the distributional vectors of the content words in the text fragment and the Multiplicative model performs an element-wise product among all the content vectors. These are used in the above models as $\kappa(a, b)$.

Finally, to investigate whether our DSTKs behave better than purely structural models, we experimented with the classical tree kernel (TK) (Collins and Duffy, 2002) and the distributed tree kernel (DTK) (Zanzotto and Dell'Arciprete, 2012). Again, these kernels are used in the above models as $\kappa(t_a, t_b)$.

4.2 Results

Table 1 reports the results for the correlation experiments. We report the Spearman's correlations over the different sets (and different dimensions of distributed vectors) between our $DSTK_0$ and the STK_0 (first two rows) and between our $DSTK_{+1}$ and the corresponding STK_{+1} (second two rows) . The correlation is above 0.80 in average for both RTE and STS datasets in the case of $DSTK_0$ and the STK_0. The correlation between $DSTK_{+1}$ and the corresponding STK_{+1} is instead a little bit lower. This depends on the fact that $DSTK_{+1}$ is approximating the sum of two kernels the TK and the STK_0 (as STK_{+1} is the sum of the two kernels). Then, the underlying feature space is bigger with respect to the one of STK_0 and, thus, approximating it is more difficult. The approximation also depends on the size of the distributed vectors. Higher dimensions yield to better approximation: if we increase the distributed vectors dimension from 1024 to 2048 the correlation between $DSTK_{+1}$ and STK_{+1} increases up to 0.80 on RTE and up to 0.77 on STS. This direct analysis of the correlation shows that our CDSM are approximating the corresponding kernel function and there is room of improvement by increasing the size of distributed vectors. Task-based experiments confirm the above trend. Table 2 and Table 3, respectively, report the correlation of different systems on STS and the accuracies of the different systems on RTE. Our CDSMs are compared against baseline systems (Add, $Mult$, TK, and DTK) in order to understand whether in the specific tasks our more complex model is interesting, and against, again, the systems with the corresponding smoothed tree kernels in order to explore whether our DSTKs approximate systems based on STKs. For all this set of experiment we fixed the dimension of the distributed vectors to 1024. Table 2 is organized as follows: columns 2-6 report the correlation of the STS systems based on syntactic/semantic similarity (SS) and columns 7-11 report the accuracies of SS systems along with token-based similarity (SSTS). The first observation for this task is that baseline systems based only on the token similarity (first row) behave extremely well. These results are above many models presented in the 2013 Shared Task (see (Agirre et al., 2013)). This can be disappointing as we cannot appreciate differences among methods in the columns SSTS. But, focusing on the results without this important token-based similarity, we can better understand if our model is capturing both structural and semantic information, that is, if DSTKs behave similarly to STKs. It is also useless to compare results of DSTKs and STKs to the Add baseline model as Add is basically doing a weighted count of the common words

	RR					RRTWS				
	RTE1	**RTE2**	**RTE3**	**RTE5**	**Average**	**RTE1**	**RTE2**	**RTE3**	**RTE5**	**Average**
Add	0.541	0.496	0.507	0.520	0.516	0.560	0.538	0.643	0.578	0.579
Mult	0.495	0.481	0.497	0.528	0.500	0.533	0.563	0.642	0.586	0.581
DTK	0.533	0.515	0.516	0.530	0.523	0.583	0.601	0.643	0.621	0.612
TK	0.561	0.552	0.531	0.54	0.546	0.608	0.627	0.648	0.630	0.628
$DSTK_0$	0.571	0.551	0.547	0.531	$0.550^{\dagger}$	0.628	0.616	0.650	0.625	$0.629^{\dagger}$
STK_0	0.586	0.563	0.538	0.545	0.558^{*}	0.638	0.618	0.648	0.636	0.635^{*}
$DSTK_{+1}$	0.588	0.562	0.555	0.541	$\mathbf{0.561}^{\dagger}$	0.638	0.621	0.646	0.652	$\mathbf{0.639}^{\dagger}$
STK_{+1}	0.586	0.562	0.542	0.546	0.559^{*}	0.638	0.618	0.650	0.636	0.635^{*}

Table 3: Task-based analysis: Accuracy on Recognizing Textual Entailment († is different from DTK and TK wiht a stat.sig. of $p > 0.1$; $*$ the difference between the kernel and its distributed counterpart is not statistically significant.)

that is exactly what the token-based similarity is doing. Add slightly decreases the performance of the token-based similarity. The $Mult$ model instead behaves very poorly. Comparing rows in the SS columns, we can discover that $DSTK_0$ and $DSTK_{+1}$ behave significantly better than DTK and that $DSTK_0$ behave better than the standard TK. Thus, our DSTKs are positively exploitng distributional semantic information along with structural information. Moreover, both $DSTK_0$ and $DSTK_{+1}$ behave similarly to the corresponding models with standard kernels STKs. Results in this task confirm that structural and semantic information are both captured by CDSMs based on DSTs.

Table 3 is organized as follows: columns 2-6 report the accuracy of the RTE systems based on rewrite rules (RR) and columns 7-11 report the accuracies of RR systems along with token similarity (RRTS). Results on RTE are extremely promising as all the models including structural information and distributional semantics have better results than the two baseline models with a statistical significance of 93.7%. For RR models $DTSK_0$, STK_0, $DSTK_{+1}$, and STK_{+1} have an average accuracy 7.9% higher than Add and 11.4% higher than $Mult$ model. For RRTS, the same happens with an average accuracy 9.58% higher than Add and 9.2% higher than the $Mult$. This task is more sensible to syntactic information than STS. As expected (Mehdad et al., 2010), STKs behave also better than tree kernels exploiting only syntactic information. But, more importantly, our CDSMs based on the DSTs are behaving similarly to these smoothed tree kernels, in contrast to what reported in (Zanzotto and Dell'Arciprete, 2011). In (Polajnar et al., 2013), it appears that results of the Zanzotto and Dell'Arciprete (2011)'s method are comparable to the results of STKs for STS, but this is mainly due to the flattening of the performance given by the lexical token similarity feature which is extremely relevant in STS. Even if distributed tree kernels do not approximate well tree kernels with distributed vectors dimension of 1024, our smoothed versions of the distributed tree kernels approximate correctly the corresponding smoothed tree kernels. Their small difference is not statistically significant (less than 70%). The fact that our DSTKs behave significantly better than baseline models in RTE and they approximate the corresponding STKs shows that it is possible to positively exploit structural information in CDSMs.

5 Conclusions and Future Work

Distributed Smoothed Trees (DST) are a novel class of Compositional Distributional Semantics Models (CDSM) that effectively encode structural information and distributional semantics in tractable 2-dimensional tensors, as experiments show. The paper shows that DSTs contribute to close the gap between two apparently different approaches: CDSMs and convolution kernels (Haussler, 1999). This contribute to start a discussion on a deeper understanding of the representation power of structural information of existing CDSMs.

References

[Agirre et al.2013] Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *sem 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational*

*Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

[Aronszajn1950] N. Aronszajn. 1950. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404.

[Baroni and Lenci2010] Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Comput. Linguist.*, 36(4):673–721, December.

[Baroni and Zamparelli2010] Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193, Cambridge, MA, October. Association for Computational Linguistics.

[Chang and Lin2011] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[Clark et al.2008] Stephen Clark, Bob Coecke, and Mehrnoosh Sadrzadeh. 2008. A compositional distributional model of meaning. *Proceedings of the Second Symposium on Quantum Interaction (QI-2008)*, pages 133–140.

[Collins and Duffy2002] Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of ACL02*.

[Corley and Mihalcea2005] Courtney Corley and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In *Proc. of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18. Association for Computational Linguistics, Ann Arbor, Michigan, June.

[Cristianini and Shawe-Taylor2000] Nello Cristianini and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, March.

[Croce et al.2011] Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1034–1046, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Dagan et al.2006] Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In Quionero-Candela et al., editor, *LNAI 3944: MLCW 2005*, pages 177–190. Springer-Verlag, Milan, Italy.

[Dagan et al.2013] Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

[Dinu et al.2013] Georgiana Dinu, Nghia The Pham, and Marco Baroni. 2013. DISSECT: DIStributional SEmantics Composition Toolkit. In *Proceedings of ACL (System Demonstrations)*, pages 31–36, Sofia, Bulgaria.

[Ferrone and Zanzotto2013] Lorenzo Ferrone and Fabio Massimo Zanzotto. 2013. Linear compositional distributional semantics and structural kernels. In *Proceedings of the Joint Symposium of Semantic Processing (JSSP)*, pages –.

[Grefenstette and Sadrzadeh2011] Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1394–1404, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Haussler1999] David Haussler. 1999. Convolution kernels on discrete structures. Technical report, University of California at Santa Cruz.

[Kalchbrenner and Blunsom2013] Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent convolutional neural networks for discourse compositionality. *Proceedings of the 2013 Workshop on Continuous Vector Space Models and their Compositionality*.

[Mehdad et al.2010] Yashar Mehdad, Alessandro Moschitti, and Fabio Massimo Zanzotto. 2010. Syntactic/semantic structures for textual entailment recognition. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 1020–1028, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Mitchell and Lapata2008] Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June. Association for Computational Linguistics.

[Polajnar et al.2013] Tamara Polajnar, Laura Rimell, and Douwe Kiela. 2013. Ucam-core: Incorporating structured distributional similarity into sts. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 85–89, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

[Socher et al.2011] Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems 24*.

[Socher et al.2012] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[Turney and Pantel2010] Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *J. Artif. Intell. Res. (JAIR)*, 37:141–188.

[Zanzotto and Dell'Arciprete2011] Fabio Massimo Zanzotto and Lorenzo Dell'Arciprete. 2011. Distributed structures and distributional meaning. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 10–15, Portland, Oregon, USA, June. Association for Computational Linguistics.

[Zanzotto and Dell'Arciprete2012] F.M. Zanzotto and L. Dell'Arciprete. 2012. Distributed tree kernels. In *Proceedings of International Conference on Machine Learning*, pages 193–200.

[Zanzotto et al.2009] Fabio Massimo Zanzotto, Marco Pennacchiotti, and Alessandro Moschitti. 2009. A machine learning approach to textual entailment recognition. *NATURAL LANGUAGE ENGINEERING*, 15-04:551–582.

[Zanzotto et al.2010] Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, August,.

Low-Dimensional Manifold Distributional Semantic Models

Georgia Athanasopoulou
School of Electronic & Computer Engineering
T.U.C. Chania, Greece
gathanasopoulou@isc.tuc.gr

Elias Iosif
Athena Research and Innovation Center,
15125 Maroussi, Greece
iosif.elias@gmail.com

Alexandros Potamianos
School of Electrical & Computer Engineering
N.T.U.A, Athens, Greece
apotam@gmail.com

Abstract

Motivated by evidence in psycholinguistics and cognition, we propose a hierarchical distributed semantic model (DSM) that consists of low-dimensional manifolds built on semantic neighborhoods. Each semantic neighborhood is sparsely encoded and mapped into a low-dimensional space. Global operations are decomposed into local operations in multiple sub-spaces; results from these local operations are fused to come up with semantic relatedness estimates. Manifold DSM are constructed starting from a pairwise word-level semantic similarity matrix. The proposed model is evaluated on semantic similarity estimation task significantly improving on the state-of-the-art.

1 Introduction

The estimation of semantic similarity between words, sentences and documents is a fundamental problem for many research disciplines including computational linguistics (Malandrakis et al., 2011), semantic web (Corby et al., 2006), cognitive science and artificial intelligence (Resnik, 2011; Budanitsky and Hirst, 2001). In this paper, we study the geometrical structure of the lexical space in order to extract semantic relations among words. In (Karlgren et al., 2008), the high-dimensional lexical space is assumed to consist of manifolds of very low dimensionality that are embedded in this high dimensional space. The manifold hypothesis is compatible with evidence from psycholinguistics and cognitive science. In (Tenenbaum et al., 2011), the question *"How does the mind work?"* is answered as follows: cognitive organization is based on domains with similar items connected to each other and lexical information is represented hierarchically, i.e., a domain that consists of similar lexical entries may be represented by a more abstract concept. An example of such a domain is {*blue, red, yellow, pink, ...*} that corresponds by the concept of *color*. An inspiring analysis about the geometry of thought, as well as cognitive evidence for the low-dimensional manifold assumption can be found in (Gardenfors, 2000), e.g., the domain of color is argued to be cognitively represented as an one-dimensional manifold. Following the *low-dimensional manifold* hypothesis we propose to extend distributional semantic models (DSMs) into a hierarchical model of *domains* (or concepts) that contain semantically similar words. Global operations on the lexical space are decomposed into local operations on the low-dimensional domain sub-manifolds. Our goal is to exploit this hierarchical low-rank model to estimate relations between words, such as semantic similarity.

There has been much research interest on devising data-driven approaches for estimating semantic similarity between words. DSMs (Baroni and Lenci, 2010) are based on the distributional hypothesis of meaning (Harris, 1954) assuming that semantic similarity between words is a function of the overlap of their linguistic contexts. DSMs are typically constructed from co-occurrence statistics of word tuples that are extracted on existing corpora or on corpora specifically harvested from the web. In (Iosif and Potamianos, 2013), general-purpose, language-agnostic algorithms were proposed for estimating semantic similarity using no linguistic resources other than a corpus created via web queries. The key idea of this work was the construction of semantic networks and semantic neighborhoods that capture smooth

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 731–740, Dublin, Ireland, August 23-29 2014.

co-occurrence and context similarity statistics. The majority of DSMs adopt high-dimensional representations, while the underlying space geometry is not explicitly taken into consideration during the design of algorithms aimed for performing several semantic tasks.

We propose the construction of a low-dimensional manifold DSM consting of four steps: 1) identify the domains that correspond to the low-dimensional manifolds, 2) run the dimensionality reduction algorithm for each domain, 3) construct a DSM for each domain, and 4) combine the manifold DSMs to come up with global measures of lexical relations. A variety of algorithms can be found in the literature for projecting a set of tokens into low dimensional sub-spaces, given a token similarity or dissimilarity matrix. Depending on the nature of the dataset, these projection algorithms may or may not preserve the local geometries of the original dataset. Most dimensionality reduction algorithms make the implicit assumption that the underlying space is metric, e.g., Multidimensional Scaling (MDS) (Torgerson, 1952) or Principal Component Analysis (PCA) (Jolliffe, 2005) or the ones using non-negative matrix factorization (Tsuge et al., 2001) and typically fail to capture the geometry of manifolds embedded in high dimensional spaces. A variety of dimensionality reduction algorithms have been developed that respect the local geometry. Some examples are the Isomap algorithm (Tenenbaum et al., 2000) that performs the projection based on a weighted neighborhood graph, Local Linear Embedings (LLE) (Roweis and Saul, 2000) that assigns neighbors to each data point, Random Projections (Baraniuk and Wakin, 2009), (Li et al., 2006) that preserves the manifold geometry by executing random linear projections and others (Hessian Eigenmaps (HLLE) (Donoho and Grimes, 2003); Maximum Variance Unfolding (MVU) (Wang, 2011)). The *manifold hypothesis* has also been studied by the representation learning community where the local geometry is disentangled from the global geometry mainly by using neighborhood graphs (Weston et al., 2012) or coding schemes (Yu et al., 2009). For a review see (Bengio et al., 2013).

A fundamental problem with all aforementioned methods when applied to lexical semantic spaces is that they do not account for ambiguous tokens, i.e., word senses. The main assumption of dimensionality reduction and manifold unfolding algorithms is that each token (word) belongs to a single sub-manifold. This in fact is not true for polysemous words, for example the word 'green' could belong both to the domain *colors*, as well as to the domain *plants*. In essence, lexical semantic spaces are manifolds that have singularities: the manifold collapses in the neighborhood of polysemous words that can be thought of *semantic black holes* that can instantaneously transfer you from one domain to another. Our proposed solution to this problem is to *allow words to live in multiple sub-manifolds*.

The algorithms proposed in this paper build on recent research work on distributional semantic models and manifold representational learning. Manifold DSMs can be trained directly from a corpus and do not require a-priori knowledge or any human-annotated resources (just like DSMs). We show that the proposed low-dimensional, sparse and hierarchical manifold representation significantly improves on the state-of-the-art for the problem of semantic similarity estimation.

2 Metrics of Semantic Similarity

Semantic similarity metrics can be broadly divided into the following types: (i) metrics that rely on knowledge resources (e.g., WordNet), and (ii) corpus-based that do not require any external knowledge source. Corpus-based metrics are formalized as Distributional Semantic Models (DSMs) (Baroni and Lenci, 2010) based on the distributional hypothesis of meaning (Harris, 1954). DSMs can be distinguished into (i) unstructured: use bag-of-words model (Iosif and Potamianos, 2010) and (ii) structured: exploitation of syntactic relationships between words (Grefenstette, 1994; Baroni and Lenci, 2010). The vector space model (VSM) constitutes the main implementation for both unstructured and structured DSMs. Cosine similarity constitutes a measurement of word similarity that is widely used on top of the VSM. The similarity between two words is estimated as the cosine of their respective vectors whose elements correspond to corpus-based co-occurrence statistics. In essence, the similarity between words is computed via second-order co-occurrences.

Direct (i.e., first-order) co-occurrences can be also used for the estimation of semantic similarity (Bollegala et al., 2007; Gracia et al., 2006). The exploitation of first-order co-occurrence statistics constitutes the simplest form of unstructured DSMs. A key parameter for such models is the definition of the context in which the words of interest co-occur: from entire documents (Bollegala et al., 2007) to paragraphs

(Véronis, 2004) and sentences (Iosif and Potamianos, 2013). The effect of co-occurrence context for the task of similarity computation between nouns is discussed in (Iosif and Potamianos, 2013). The underlying assumption is that two words that co-occur in a specified context are semantically related.

3 Collapsed Manifold Hypothesis, Low-Dimensionality and Sparsity

The intuition behind this work is that although the lexical semantic space proper is high-dimensional, it is organized in such a way that interesting semantic relations can be exported from manifolds of much lower dimensionality embedded in this high dimensional space (Karlgren et al., 2008). We assume that (at least some of) these sub-manifolds contain semantically similar words (or word senses). For example, a potential sub-manifold in the lexical space could be the one that contains the colors (e.g., *red, blue, green*). But in fact many words, such as *book, green, fruit*, are expected to belong simultaneously in semantically different manifolds because they have multiple meanings.

A simple way to bootstrap the manifold recreation process is to build a domain around each word, i.e., *the semantic neighborhood of each word defines a domain*. For example, in Figure 1 we show the semantic neighborhood of *fruit*. The connections between words indicate high semantic similarity, i.e., this is a pruned semantic similarity graph of all words in the semantic neighborhood of the word 'fruit'. It is clear from this example that in a typical neighborhood there exist word pairs that should be

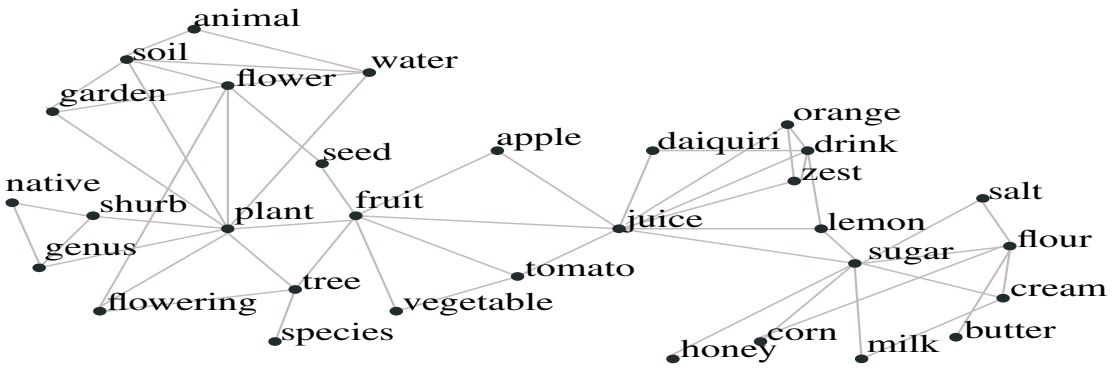


Figure 1: Visualization of the semantic neighborhood of the word 'fruit'.

'connected' to each other because they have close semantic relation, like {*flower*, *plant*} and others that should not be 'connected' because they are semantically apart, like {*garden*, *salt*}. A *sparse encoding* of the semantic similarity relations in a neighborhood is needed in order to achieve (via multi-dimensional scaling) a parsimonious representation with good geometric properties[1].

The graph connectivity or sparseness matrix identifies the word pairs that should be encoded in a neighborhood is defined as $\tilde{\mathbf{S}} \in \{0,1\}^{n \times n}$, where value $\tilde{\mathbf{S}}(i,j) = 1$ indicates that the i^{th}, j^{th} word pair is encoded, while $\tilde{\mathbf{S}}(i,j) = 0$ indicates that the pair is ignored (n is the number of words and $i,j = 1,..,n$ in the neighborhood). We define the degree of sparseness of matrix $\tilde{\mathbf{S}}$ as the percentage of 0's in the matrix.

4 Dimensionality Reduction

In this section, the Sparse Projection (SP) algorithm is described (see also Algorithm 1). SP is the core algorithm for constructing manifold DSMs presented in Section 5. SP is a dimensionality reduction algorithm that projects a set of n words into a vector space of d dimensions. The input to the algorithm is a dissimilarity or semantic distance matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$, where element $\mathbf{P}(i,j)$ encodes the degree of dissimilarity between words w_i and w_j. The output of SP are the d-dimensional coordinate vectors of the n projected words that form a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$. Each row $\mathbf{x}_i \in \mathbb{R}^{1 \times d}$ of matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ corresponds to the coordinates of the i^{th} word w_i. Once $\mathbf{X}$ is estimated the dissimilarity matrix is recomputed and updated to new values, as discussed next. Each paragraph that follows corresponds to a module in Algorithm 1.

[1]Compare for example with Isomap (Tenenbaum et al., 2000) were a short- and long-distance metric is used. When using sparse encoding the long-distance metric is set to a very large fixed number (similarity set to 0). In both cases, the underlying manifold is unfolded and low-dimensional representation with (close to) metric properties are discovered.

Semantic Distance Re-estimation: Given the matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ containing the vector projections of words in the d-dimensional space, the dissimilarity matrix is re-estimated using the Euclidean distance[2]. Let $\hat{\mathbf{P}} \in \mathbb{R}^{n \times n}$ be the matrix with the new dissimilarity scores then the new dissimilarity score between words w_i and w_j is simply: $\hat{\mathbf{P}}(i,j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2$, where x_i, x_j are the vectors corresponding to words w_i, w_j respectively, $i, j = 1, .., n$ and $\|.\|_2$ is the Euclidean norm.

Connectivity Graph and Sparsity: As discussed in Section 3, given a set of words only a small subset of lexical relations should be explicitly encoded between pairs of these words. Therefore, the SP algorithm should only take into account strongly related word pairs and ignore the rest. This is the main difference between our approach compared to the generic MDS algorithm proposed in (Torgerson, 1952). In order to apply the sparseness constraint, we first construct the connectivity matrix $\tilde{\mathbf{S}} \in \{0,1\}^{n \times n}$. Word pairs (w_i, w_j) with small similarity values (or equivalently large semantic distance) are penalized: zero values are assigned to their corresponding position (i,j) in $\tilde{\mathbf{S}}$ matrix. In essence, the matrix $\tilde{\mathbf{S}}$ is obtained by hard $\{0,1\}$ thresholding on the dissimilarity matrix $\mathbf{P}$: all values that are under a threshold are set to 0, while all values equal or greater to the threshold are set to 1. Let n be the number of words under investigation, then the number of word pairs is $p = \frac{n \cdot (n-1)}{2}$. The degree of sparseness is defined as the number of unordered word pairs (w_i, w_j), $i \neq j$ where $\tilde{\mathbf{S}}(i,j) = 0$ normalized over the total number of pairs p [3].

Error Criterion: The algorithm employs a local and a global error criterion defined as follows:

1. The local error corresponds to the projection error for each individual word w_i $\mathbf{e} \in \mathbb{R}^{n \times 1}$, where $i = 1...n$ and is defined as the sum of the dissimilarity matrix errors before and after projection computed only for the words that are 'connected' to w_i, as follows:

$$\mathbf{e}_i = \sum_{j=1}^{n} \tilde{\mathbf{S}}(i,j) \cdot \left(\hat{\mathbf{P}}(i,j) - \mathbf{P}(i,j) \right)^2 \quad (1)$$

2. The global error of the projection is simply the sum over local errors for all words: $e_{tot} = \sum_{i=1}^{n} \mathbf{e}_i$

Algorithm 1 Sparse projection (SP)

Require: $\mathbf{v}$ // Vocabulary: vector of n words
Require: $\mathbf{P}$ // $n \times n$ dissimilarity matrix
$\tilde{\mathbf{S}} \leftarrow$ ComputeConnectivityMatrix($\mathbf{S}$)
for each word $w_i \in \mathbf{v}$ **do**
  $\mathbf{X}_i \leftarrow$ RandomInitialization($\mathbf{X}_i$)
end for
$k = 0$ // Iteration counter: initialization
$e_{tot}^k = \inf$ // Global error: initialization
repeat
  $k = k + 1$
  for each word $w_i \in \mathbf{v}$ **do**
    for each direction z **do**
      $\mathbf{X} \leftarrow$ MoveWordToDirection(w_i, z)
      $\mathbf{e}_i^z \leftarrow$ ComputeLocalError($\tilde{\mathbf{S}}$,$\mathbf{P}$,$\mathbf{X}$,i)
    end for
    $\hat{z}_i \leftarrow$ FindDirectionOfMinLocalError($\mathbf{e}_i^z$)
    $\mathbf{X}$ = MoveWordToDirection($w_i, \hat{z}_i$)
  end for
  $e_{tot}^k \leftarrow$ UpdateGlobalError($\tilde{\mathbf{S}}$,$\mathbf{P}$,$\mathbf{X}$)
until $e_{tot}^{k-1} < e_{tot}^k$ // Stopping condition
$\hat{\mathbf{P}} \leftarrow$ SemanticDistanceReestimation($\mathbf{X}$)
$\tilde{\mathbf{P}} \leftarrow$ SparseDistanceNormalizedRanges($\hat{\mathbf{P}}$,$\tilde{\mathbf{S}}$)
return $\mathbf{X}$ // $n \times d$ matrix with coordinates;
return $\tilde{\mathbf{S}}$ // $n \times n$ matrix with connections;
return $\hat{\mathbf{P}}$ // $n \times n$ updated dissimilarity matrix;
return $\tilde{\mathbf{P}}$ // $n \times n$ sparse-normalized distances;

Random Walk SP: In function MoveWordToDirection($\cdot$) of Algorithm 1, the pseudo-variable *direction* z refers to a standard set of perturbations of each word in the d-dimensional space. For example, if the dimension of the projection is $d = 2$ then the coordinates of each word are modeled as (k_1, k_2), where $k_1, k_2 \in \mathbb{R}$. A potential set of perturbations are the following: $(k_1, k_2 + s)$, $(k_1, k_2 - s)$, $(k_1 + s, k_2)$ and $(k_1 - s, k_2)$, where s is the perturbation step parameter of the algorithm. For coordinates systems normalized in $[0,1]^d$ we chose a value of s equal to 0.1. Good convergence properties to global maxima have been experimentally shown for this algorithm for multiple runs on (noisy) randomly generated data.

[2] Other metrics, e.g., cosine similarity, have also been tested out but results are not shown here due to lack of space. Euclidean distance performed somewhat better that cosine similarity for the semantic similarity estimation task.

[3] The SP algorithm with 0% degree of sparseness is equivalent to the MDS algorithm.

Sparse Semantic Distance Normalized Ranges: This function normalizes all the distance scores of $\hat{\mathbf{P}}$ in a range of values, [0 r_1], where $r_1 \in \mathbb{R}_+$ is an arbitrary positive constant and also it imposes the sparsity constraint as follows: if $\tilde{\mathbf{S}}(i,j) = 0$ then $\tilde{\mathbf{P}}(i,j) = r_1$. If $\tilde{\mathbf{S}}(i,j) = 1$ then $\tilde{\mathbf{P}}(i,j) = r_2 \cdot \frac{\hat{\mathbf{P}}(i,j)}{r_3}$, where r_3 is the maximum distance over all 'connected' pairs, i.e. $r_3 \triangleq \max\{\hat{\mathbf{P}} \odot \tilde{\mathbf{S}}\}$, with $\odot$ denoting the Hadamard product, and $r_2 \in \mathbb{R}_+$ can be either equal to r_1 or slightly smaller than r_1. The assignment of $r_2 < r_1$ aims to differentiate the 'unconnected' pairs from the 'connected' but dissimilar ones [4].

5 Low-Dimensional Manifold DSMs

The end-to-end low-dimensional manifold DSM (LDMS) system is depicted in Figure 2. Note that $\mathrm{v}_1, \mathrm{v}_2, ..., \mathrm{v}_{|\mathrm{V}|} \in \mathrm{V}$ are the domains or sub-manifolds of the LDMS, for each domain v_i a separate DSM is built. V is the set of domains (concept vocabulary) and $|\mathrm{V}|$ denotes to the cardinality of V. The input

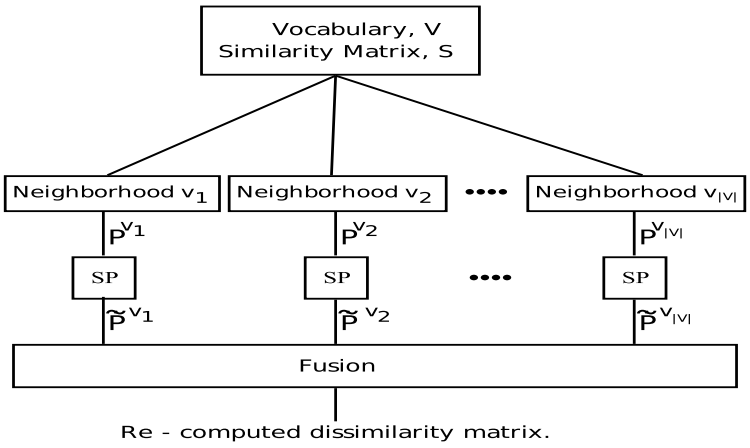


Figure 2: LDMS system.

to LDMS is a (global) similarity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$, where n is the total number of tokens (words) in the LDMS model. Note that $\mathbf{S}$ can be estimated using any of the baseline semantic similarity metrics[5] presented in Section 2. Since the SP algorithm uses as input a dissimilarity or semantic distance matrix, the pairwise word similarity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$ is transformed to a semantic distance (or dissimilarity) matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$ as: $\mathbf{P}(i,j) = c_1 \cdot e^{-c_2 \cdot \mathbf{S}(i,j)}$ where $c_1, c_2 \in \mathbb{R}$ are constants and the i, j indexes run from 1 to n. In this work, we used $c_1 = c_2 = 20$. The transformation defined by (5) was selected in order to non-linearly scale and increase the relative distance of dissimilar words compared to similar ones[6].

The steps followed by the LDMS system are the following:

1. **Domain Selection**: The domains $\mathrm{v}_1, \mathrm{v}_2, ..., \mathrm{v}_{|\mathrm{V}|}$ are created as follows: for each word w_i in our model we create a corresponding domain v_i that consists of all the words that are semantically similar to w_i, i.e., the ith domain is the semantic neighborhood of word w_i. Thus in our model the vocabulary size is equal to the domain set cardinality, i.e., $n = |\mathrm{V}|$. Domain v_i is created by selecting the top N most semantically similar words to w_i based on the (global) similarity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$. We have experimented with various domain sizes N ranging between 20 and 200 neighbors; note that each word in the LDMS may belong to multiple domains.
2. **Sparse Projections on Domains**: Following the selection of domain $\mathrm{v}_i \in \mathrm{V}$ the (local) dissimilarity matrix for each domain $\mathbf{P}^{\mathrm{v}_i} \in \mathbb{R}^{N \times N}$ is defined as a submatrix of $\mathbf{P} \in \mathbb{R}^{n \times n}$. Then, the SP algorithm is applied to each domain separately, resulting in $i = 1, .., |\mathrm{V}|$ re-estimated bounded semantic distance matrices $\tilde{\mathbf{P}}^{\mathrm{v}_i}$.
3. **Fusion**: To reach a decision on the strength of the semantic relation between words w_i and w_j the semantic distance matrices from each domain $\tilde{\mathbf{P}}^{\mathrm{v}_i}$ must be combined. Only domains were both words w_i and w_j appear are relevant in this fusion process. This procedure is described next.

[4] We experimented with various values for r_1 and r_2 achieving comparable performance; we selected $r_2 \approx 0.9 r_1$ that had slightly better performance. The value of r_1 can be chosen arbitrary, the results reported here were obtained for $r_1 = 20$ and $r_2 = 18$.

[5] Here, the Google-based Semantic Relatedness was employed using a corpus of web-harvested document snippets.

[6] Similar nonlinear scaling function from similarity to distance can be found in the literature, e.g., (Borg, 2005)

5.1 Fusion

Motivation: Given a set of words $L = \{w_1, w_2, ...w_n\}$ we assume that their corresponding set of word senses[7] is $M = \{s_{11}, s_{12}, .., s_{1n_1}, .., .., s_{n1}, s_{n2}, .., s_{nn_n}\}$. The set of senses is defined as $M = \cup_{i=1}^{n} M_i$, where $M_i = \{s_{i1}, s_{i2}, ..., s_{in_i}\}$ is the set of senses for word w_i. Let $S(.)$ be a metric of semantic similarity, e.g., the metric defined in Section 2, which is symmetric, i.e., $S(x, y) \equiv S(y, x)$. The notations $S_w(.)$ and $S_s(.)$ are used in order to distinguish the similarity at word and sense level, respectively. According to the maximum sense similarity assumption (Resnik, 1995), the similarity between w_i and w_j, $S_w(w_i, w_j)$, is defined as the pairwise maximum similarity between their corresponding senses $S_s(s_{ik}, s_{jl})$:

$$S_w(w_i, w_j) \equiv S_s(s_{ik}, s_{jl}), \quad \text{where} \quad (k, l) = \underset{(p \in M_i, r \in M_j)}{\operatorname{argmax}} S_s(s_{ip}, s_{jr}).$$

Note that the maximum pairwise similarity metric (or equivalently the *minimum pairwise distance metric*) is also known as the "common sense" set similarity (or distance) employed by human cognition when evaluating the similarity (or distance) between two sets.

Fusion of local dissimilarity scores: Next we describe a domain fusion model that follows the minimum pairwise distance (dissimilarity) principle motivated by human cognition. The steps for the recomputation of the (global) dissimilarity between words w_i and w_j are:

1. Search for all the domains where w_i and w_j co-exist.
2. Let $U \subset V$ be the subset of domains from the previous step. The distances between words w_i and w_j are retrieved from domain dissimilarity matrices $\tilde{\mathbf{P}}^u$ for all $u \in U$. The distances are stored into vector $\mathbf{d} \in \mathbb{R}^{|U| \times 1}$.
3. Motivated by the maximum sense similarity assumption (see above) the dissimilarity between w_i and w_j is defined as[8]:
$$\hat{\mathbf{P}}(i, j) = \min_{k=1..|U|} \{\mathbf{d}_k\} \tag{2}$$
4. If words w_i and w_j do not co-exist in any domain then r_1 is assigned as their dissimilarity score, where r_1 is the upper bound of $\tilde{\mathbf{P}}^u$ matrices as defined in the previous section.

For example, let one pair of words (w_1, w_2) co-exists in $|U| = 3$ different domains with corresponding local distances $\mathbf{d} = [9\ 20\ 11]$ then the global distance of (w_1, w_2) is 9.

6 Evaluation

In this section, we evaluate the performance of the proposed approach with respect to the task of similarity judgment between nouns. Results are reported with respect to several domain/neighborhood sizes, sparse percentages and domain dimensions.

The performance of similarity metrics were evaluated against human ratings from three standard datasets of noun pairs, namely *WS353* (Finkelstein et al., 2001), *RG* (Rubenstein and Goodenough, 1965) *MC* (Miller and Charles, 1991). The first and the second datasets consist of the subset of 272 and 57 pairs, respectively, that are also included in SemCor3[9] corpus, while the third dataset consists of 28 noun pairs. The Pearson's correlation coefficient was selected as evaluation metric to compare estimated similarities against the ground truth.

The similarity matrix computed using the Google-based Semantic Relatedness (Gracia et al., 2006) was used as baseline, as well as to bootstrap the LDMS global similarity matrix $\mathbf{S}$, for a list of 8752 nouns extracted from the SemCor3 corpus [10]. The performance of the proposed LDMS approach is presented in Table 1. In addition, the performance of other *unsupervised* similarity estimation algorithms are reported for comparison purposes: 1) SEMNET is an alternative implementation of unstructured DSMs based on the idea of semantic neighborhoods and networks (Iosif and Potamianos, 2013) 2) WikiRelate! includes various taxonomy-based metrics that are typically applied to the WordNet hierarchy; the basic

[7] This is a simplification. In reality, some of the word senses will be the same, so strictly speaking this is not a set definition.

[8] Other fusion methods have also been evaluated, e.g., (weighted) average. Results are omitted here due to lack of space. Minimum pairwise distance fusion outperformed other fusion schemes.

[9] `http://www.cse.unt.edu/~rada/downloads.html`

[10] The baseline similarity matrix and the 8752 nouns are public available in:
`http://www.telecom.tuc.gr/~iosife/downloads.html`

idea behind WikiRelate! is to adapt these metrics to a hierarchy extracted from the links between the pages of the English Wikipedia (Strube and Ponzetto, 2006) . 3) TypeDM is a structured DSM (Baroni and Lenci, 2010), 4) AAHKPS1 constitutes an unstructured paradigm of DSM development using four billion web documents that were acquired via crawling (Agirre et al., 2009), 5) Moreover, two well-established dimensionality reduction algorithms (Isomap and LLE) that support the manifold hypothesis, were applied to the task of semantic similarity computation [11]. LDMS, Isomap and LLE were given as input the matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$, where $n = 8752$ is the number of words in our models. Isomap and LLE used dimensionality reduction down to $d = 5$ and neighborhood size equal to $N = 120$. SEMNET was run for neighborhood size equal to $N = 100$. While LDMS run for dimensionality down to $d = 5$, domain/neighborhood size equal to $N = 140$ and degree of sparseness 90%. The proposed LDMS system surpassed the performance of the baseline system for all three datasets, as well as the performance of the other corpus-based approaches for the WS353 and MC datasets. The dimensionality reduction algorithms (Isomap - LLE) are shown to perform poorly for this particular task.

Datasets	Algorithm							
	Baseline	SEMNET	WikiRelate!	TypeDM	AAHKPS1	Isomap	LLE	**LDMS**
WS353	0.61	0.64	0.48	-	-	0.14	0.04	0.69
RG	0.81	0.87	0.53	0.82	-	0.04	0	0.86
MC	0.85	0.91	0.45	-	0.89	-0.04	-0.04	0.94

Table 1: Performance of various algorithms for the task of similarity judgment.

The performance (Pearson correlation) of the LDMS approach is shown in Figures 3a, 3b and 4a as a function of neighborhood size and degree of sparseness. Results are presented for all three datasets: WS353, MC, and RG. The baseline performance is also plotted (dotted line). For all three datasets, we see a clear relationship between neighborhood size, degree of sparseness and performance. Sparse representations achieve peak performance for larger neighborhood sizes. High degree of sparseness between 80 and 90% achieves the best results for domain/neighborhood sizes between 100 and 140. The figures show that there is potential for even better performance by fine-tuning the LDMS parameters.

The performance of LDMS is shown in Figure 4b as a function of the projection dimension d. The degree of sparseness is fixed at 80% and the domain/neighborhood size is equal to 100 for all experiments. It is observed that the performance for all three datasets remains relatively constant when at least $d = 3$ is used. In fact results are slightly better for $d = 3$ than for higher dimensions but the differences in performance are not significant. The results suggest that even *a 3D sub-space is adequate for accurately representing the semantics of each underlying domain.*

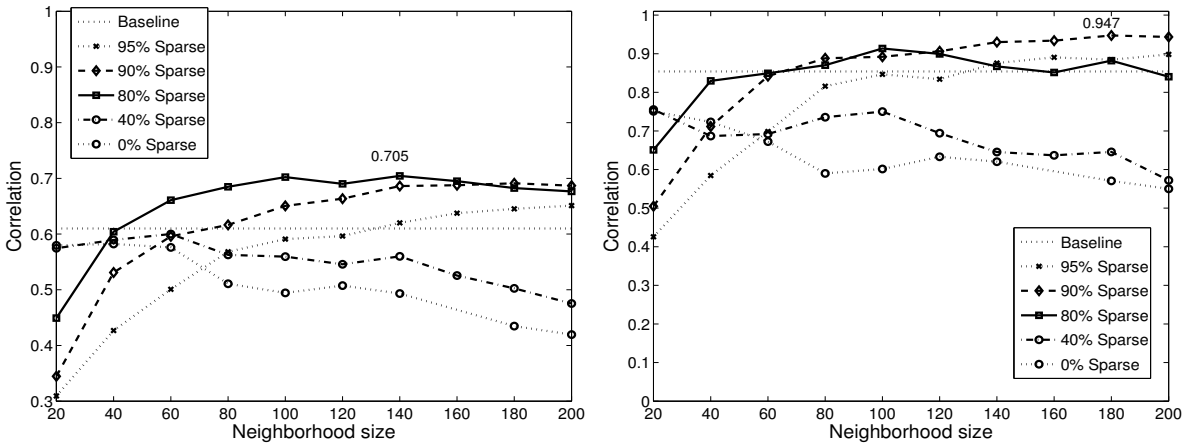


Figure 3: Performance as a function of domain size N and sparseness percentage for the (a) WS353 dataset and (b) MC dataset.

[11] LDMS is not directly comparable with Isomap-LLE algorithms because it represents only the domains in low-dimensional spaces and not the whole dataset.

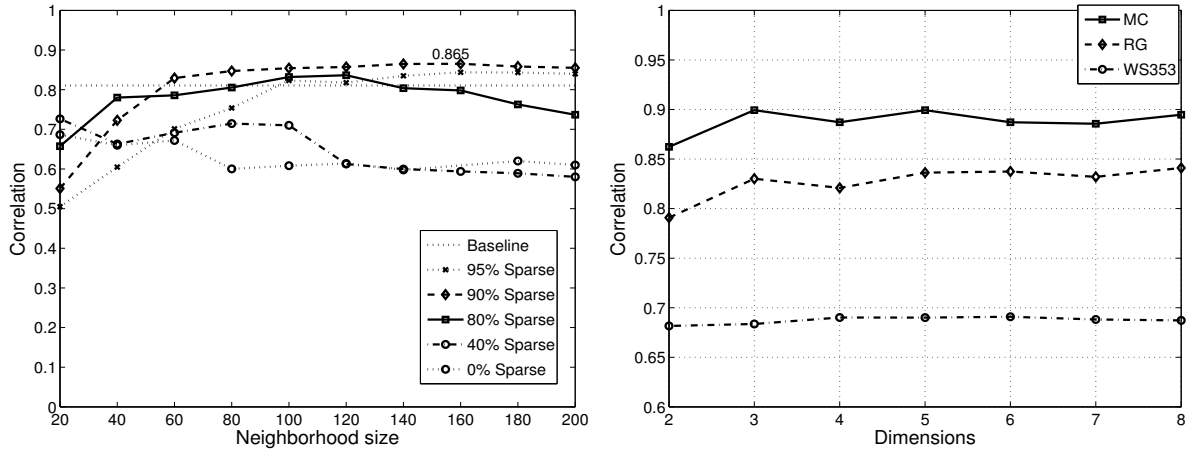


Figure 4: Performance for the (a) RG dataset as a function of domain size N and sparseness percentage and (b) WS353, MC, RG datasets as a function of projection dimension d.

7 Conclusions

In this work, we proposed a novel, hierarchical DSM that was applied to semantic relation estimation task obtaining very good results. The proposed representation consists of low-dimensional manifolds that are derived from sparse projections of semantic neighborhoods. The core idea of low dimensional subspaces was motivated by cognitive models of conceptual spaces. The validity of this motivation was experimentally verified via the estimation of semantic similarity between nouns. The proposed approach was found to be (at least) competitive with other state-of-the-art DSM approaches that adopt flat feature representations and do not explicitly include the sparsity and dimensionality as a key design parameter.

The poor performance of Isomap and LLE can be attributed to the nature of the specific application, i.e., word semantics. A key characteristic of this application is the ambiguity of word senses. These algorithms assume only one sense for each word (i.e., a word is represented as a single point in a high-dimensional space). Although the disambiguation task is not explicitly addressed, LDMS approach handles the ambiguity of words by isolating each word's senses in different domains.

Our initial intuition regarding the semantic fragmentation of lexical neighborhoods due to singularities introduced by word senses was supported by the high performance when large (i.e., 80% - 90%) degree of sparseness was imposed. The hypothesis of low-dimensional representation was validated by the finding that as little as three dimensions are adequate for representing domain/neighborhood semantics. It was also observed that the parameters of the LDMS model, i.e., number of dimensions, neighborhoodsize and degree of sparseness, are interrelated: very sparse projections achieve best results with very low dimensionality when large neighborhood sizes are used.

This is only a first step toward using ensembles of low-dimensional DSMs for semantic relation estimation. As future work we plan to further investigate the creation of domains based on more complex geometric properties of the underlying space (Kreyszig, 2007). A more formal investigation of the relation between sparseness, dimensionality and performance is also needed. Finally, creating multi-level hierarchical representations that are consistent with cognitive organization is an important challenge that can further improve manifold DSM performance.

Acknowledgments

This work has been partially funded by two projects supported by the EU Seventh Framework Programme (FP7): 1) PortDial, grant number 296170 and 2) SpeDial, grant number 611396.

References

E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, and A. Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies*, pages 19–27. Association for Computational Linguistics.

R. G Baraniuk and M. B Wakin. 2009. Random projections of smooth manifolds. *Foundations of computational mathematics*, 9(1):51–77.

M. Baroni and A. Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

Y. Bengio, A. Courville, and P. Vincent. 2013. Representation learning: A review and new perspectives.

D. Bollegala, Y. Matsuo, and M. Ishizuka. 2007. Measuring semantic similarity between words using web search engines. In *Proc. of International Conference on World Wide Web*, pages 757–766, Banff, Alberta, Canada.

Ingwer Borg. 2005. *Modern multidimensional scaling: Theory and applications*. Springer.

A. Budanitsky and G. Hirst. 2001. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources*.

O. Corby, R. Dieng-Kuntz, F. Gandon, and C. Faron-Zucker. 2006. Searching the semantic web: Approximate query processing based on ontologies. *Intelligent Systems, IEEE*, 21(1):20–27.

D. L Donoho and C. Grimes. 2003. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596.

L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM.

P. Gardenfors. 2000. Conceptual spaces: The geometry of thought. *Cambridge, Massachusetts: USA. ISBN*, 262071991.

J. Gracia, R. Trillo, M. Espinoza, and E. Mena. 2006. Querying the web: A multiontology disambiguation method. In *Proc. of International Conference on Web Engineering*, pages 241–248, Palo Alto, California, USA.

G. Grefenstette. 1994. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, Norwell, MA, USA.

Z. Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

E. Iosif and A. Potamianos. 2010. Unsupervised semantic similarity computation between terms using web documents. *Knowledge and Data Engineering, IEEE Transactions on*, 22(11):1637–1647.

E. Iosif and A. Potamianos. 2013. Similarity computation using semantic networks created from web-harvested data. Natural Language Engineering (DOI: 10.1017/S1351324913000144).

I. Jolliffe. 2005. *Principal component analysis*. Wiley Online Library.

J. Karlgren, A. Holst, and M. Sahlgren. 2008. Filaments of meaning in word space. In *Advances in Information Retrieval*, pages 531–538. Springer.

E. Kreyszig. 2007. *Introductory functional analysis with applications*. Wiley. com.

P. Li, T. J Hastie, and K. W Church. 2006. Very sparse random projections. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 287–296. ACM.

N. Malandrakis, A. Potamianos, E. Iosif, and S. S Narayanan. 2011. Kernel models for affective lexicon creation. In *INTERSPEECH*, pages 2977–2980.

G. A Miller and W. G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28.

P. Resnik. 1995. Using information content to evaluate semantic similarity in a taxanomy. In *Proc. of International Joint Conference for Artificial Intelligence*, pages 448–453.

P. Resnik. 2011. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *arXiv preprint arXiv:1105.5444*.

S. T Roweis and L. K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326.

H. Rubenstein and J. B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Michael Strube and Simone Paolo Ponzetto. 2006. Wikirelate! computing semantic relatedness using wikipedia. In *AAAI*, pages 1419–1424.

J. B Tenenbaum, V. De Silva, and J. C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323.

J. B Tenenbaum, C. Kemp, T. L Griffiths, and N. D Goodman. 2011. How to grow a mind: Statistics, structure, and abstraction. *science*, 331(6022):1279–1285.

Warren S Torgerson. 1952. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419.

S. Tsuge, M. Shishibori, S. Kuroiwa, and K. Kita. 2001. Dimensionality reduction using non-negative matrix factorization for information retrieval. In *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, volume 2, pages 960–965 vol.2.

J. Véronis. 2004. Hyperlex: Lexical cartography for information retrieval. *Computer Speech and Language*, 18(3):223–252.

Jianzhong Wang. 2011. Maximum variance unfolding. In *Geometric Structure of High-Dimensional Data and Dimensionality Reduction*, pages 181–202. Springer.

J. Weston, F. Ratle, H. Mobahi, and R. Collobert. 2012. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer.

K. Yu, T. Zhang, and Y. Gong. 2009. Nonlinear learning using local coordinate coding. In *Advances in Neural Information Processing Systems*, pages 2223–2231.

An Entity-Centric Coreference Resolution System for Person Entities with Rich Linguistic Information

Marcos Garcia and **Pablo Gamallo**
Centro Singular de Investigación en Tecnoloxías da Información (CiTIUS)
University of Santiago de Compostela
{marcos.garcia.gonzalez, pablo.gamallo}@usc.es

Abstract

This paper presents a first version of LinkPeople, an entity-centric system for coreference resolution of person entities. The approach combines (i) a multi-pass architecture which takes advantage of entity features at document-level with (ii) a set of linguistically-motivated constraints and rules which allows the system to restrict the candidates of a given mention. The paper includes evaluations and error analysis of LinkPeople in 3 different languages, achieving promising results (more than 81% F1 in different metrics). Both the system and the corpora are freely distributed.

1 Introduction

Coreference Resolution (CR) is a crucial task for several Natural Language Processing (NLP) applications such as Text Summarization, Machine Translation or Information Extraction (IE).

Specially for IE, person entities are those which more effort have deserved from different perspectives. Evaluations such as the Knowledge Base Population (KBP) Slot Filling Task (in the Text Analysis Conference)[1] and the Person Attribute Extraction (in the Web People Search Evaluation Campaign, WePS)[2], tasks such as Personal Name Matching (Cohen et al., 2003), or different works on Relation Extraction of person entities (Mann, 2002; Garcia and Gamallo, 2013) are some examples of their importance.

Recently, entity-centric models for coreference resolution, which use features from all the mentions of an entity, have shown better performance than pair-mention systems, which carry out coreference resolution on single pairs of mentions (Lee et al., 2013).[3] Furthermore, the use of linguistic information such as syntax or semantic knowledge has proved to be essential for high-precision CR (Ng and Cardie, 2002; Ponzetto and Strube, 2006; Uryupina, 2007).

This paper presents the first version of LinkPeople, an open-source system for CR of person entities. LinkPeople is inspired by the Stanford Deterministic Coreference Resolution System (Raghunathan et al., 2010; Lee et al., 2013), using a multi-pass architecture which applies a battery of modules sorted from high-precision to high-recall.

Moreover, the system presented in this paper adds new sieves based on linguistic knowledge, for both cataphoric and anaphoric mentions: It includes a high-precision module which finds cataphoric mentions of Noun Phrases (NP) and personal and elliptical pronouns. The inclusion of this module is based on the claim that definite NPs are not primarily anaphoric (Vieira and Poesio, 2000). In addition, LinkPeople applies a set of syntactic constraints on the pronominal CR module, increasing its precision by blocking links which do not satisfy the constraints (Mitkov, 1998; Palomar et al., 2001; Chaves and Rino, 2007).

The system was evaluated in three languages (Portuguese, Spanish and Galician) with promising results (F1 $\approx$ 83%, with BLANC score). Both LinkPeople and the corpora are freely distributed.[4]

[1] http://www.nist.gov/tac/data/index.html

[2] http://nlp.uned.es/weps/weps-3

[3] In this paper, a *mention* is every instance of reference to a person. An *entity* is the group of all the mentions referring to the same person in the text (Recasens and Martí, 2010).

[4] http://gramatica.usc.es/~marcos/coling14.tar.bz2

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 741–752, Dublin, Ireland, August 23-29 2014.

Apart from this Introduction, Section 2 contains some related work. The architecture of the system is presented in Section 3 while its evaluation is shown in Section 4. Finally, the results of an error analysis are presented in Section 5, and some conclusions and further work are pointed out in Section 6.

2 Related Work

Coreference (and anaphora) resolution is one of the older topics in NLP, so it has been the subject of many works. Two main distinctions can be stated in coreference resolution systems: (i) mention-pair *vs* entity-centric approaches and (ii) machine learning-based *vs* rule-based models.

On the one hand, mention-pair systems classify two mentions in a text as coreferent or not, by using a feature vector obtained from this pair of mentions. On the other hand, entity-centric approaches determine if a mention (or a partial entity) belongs to another partial entity, using features from other mentions of the same (partial) entities.[5]

Machine learning classifiers for CR often use annotated corpora for training supervised models. Supervised models rely on these data in order to learn preferences and constraints (McCarthy and Lehnert, 1995; Soon et al., 2001; Ng and Cardie, 2002; Sapena et al., 2013), while unsupervised models apply clustering approaches to the coreference resolution problem (Haghighi and Klein, 2007; Ng, 2008).

Rule-based strategies make use of sets of rules and heuristics for finding the best element to link each mention to (Lappin and Leass, 1994; Baldwin, 1997; Mitkov, 1998; Bontcheva et al., 2002; Raghunathan et al., 2010; Lee et al., 2013). This last system is based on a multi-pass approach which first solves the *easy* links, then increasing the recall with more rules. Stoyanov and Eisner (2012) presented *EasyFirst*, which uses annotated corpora in order to know whether coreference links are easy or hard.

Concerning the languages LinkPeople deals with, some studies addressed pronominal CR in Portuguese (Paraboni, 1997; Chaves and Rino, 2007; Cuevas and Paraboni, 2008). Coelho and Carvalho (2005) adapted the Lappin and Leass (1994) algorithm for this language, while de Souza et al. (2008) presented a supervised approach for solving the coreference between NPs.

For Spanish, Palomar et al. (2001) presented a set of constraints and preferences for pronominal anaphora resolution. Recasens and Hovy (2009) analyzed the impact of several features for CR, then implemented in Recasens and Hovy (2010). The availability of a large coreference annotated corpus for Spanish (Recasens and Martí, 2010) also allowed other supervised systems being adapted for this language (Recasens et al., 2010).

To the best of our knowledge, there are no specific systems for coreference or anaphora resolution for Galician language.

Other related areas such as the above mentioned personal name matching perform coreference resolution of personal names by linking variants referring to the same person (Cohen et al., 2003).

The system presented in this paper uses a similar approach than Lee et al. (2013), adapting —and adding— some modules for person entities, and enriching others with linguistic-based heuristics such as cataphoric analysis and syntactic constraints.

3 Architecture of LinkPeople

LinkPeople is based on two main principles: (i) an entity-centric approach and (ii) a multi-pass architecture. On the one hand, the entity-centric approach allows the system to use all the features of an entity when a mention is evaluated. On the other hand, the multi-pass model dynamically enriches an entity (with new features) in every iteration. Thus, latter passes take advantage of the information provided by the previous coreference resolution modules.

Figure 1 shows a text with coreference annotation of person entities. It will be used to show how the system works. The input of LinkPeople needs to be pre-processed by NLP tools which provide PoS-tags, Named Entity Recognition (NER) and dependency analysis. In our experiments, FreeLing (Padró and Stanilovsky, 2012; Garcia and Gamallo, 2010) was used for tokenizing, lemmatizing and PoS-tagging. NER labeling for Spanish and Portuguese was also added by FreeLing (Carreras et al., 2003; Gamallo

[5]Partial entities are sets of mentions of the same entity.

Who was $_{1}$[the singer of the Beatles]$_{1}$. $_{2}$[The musician John Winston Ono Lennon]$_{1}$ was one of the founders of the Beatles. With $_{3}$[Paul McCartney]$_{2}$, $_{4}$[he]$_{1}$ formed a songwriting partnership. $_{5}$[Lennon]$_{1}$ was born at Liverpool Hospital to $_{6}$[Julia]$_{3}$ and $_{7}$[Alfred Lennon]$_{4}$. $_{8/9}$[$_{10}$[His]$_{1}$ parents]$_{3/4}$ named $_{11}$[him]$_{1}$ $_{12}$[John Winston Lennon]$_{1}$. $_{13}$[Lennon]$_{1}$ revealed a rebellious nature and acerbic wit. $_{14}$[The musician]$_{1}$ was murdered in 1980.

Figure 1: Example of a text with coreference annotation of person entities. Mentions appear inside brackets. Numbers on the left are mention ids, while entity ids appear in the right side.

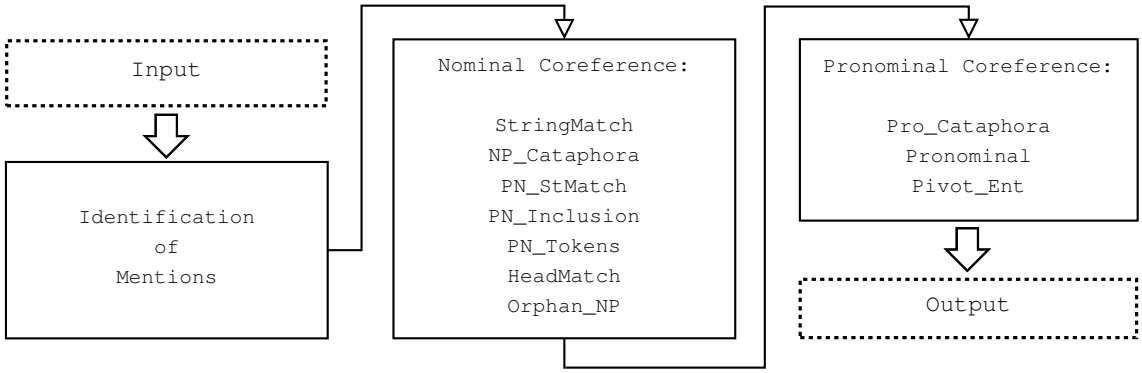


Figure 2: Architecture of the system.

and Garcia, 2011), while the named entities in Galician were classified by the system presented in Garcia et al. (2012). Finally, dependency information for the three languages was added by DepPattern (Gamallo and González López, 2011).

3.1 Coreference Resolution Modules

Figure 2 summarizes the architecture of the system, which starts by identifying the mentions. Then, a battery of nominal and pronominal CR modules is applied. Modules with high-precision are applied first, while other modules increase recall by taking advantage of the previously extracted features.

In the first stage, a specific pass identifies the mentions referring to a person entity, using the information provided by the PoS-tagger and the NER as well as applying basic approaches for NP and elliptical pronoun identification: First, personal names (and noun phrases including personal names) are identified. Then, it seeks for definite NPs whose head may refer to a person (e.g., "the singer"). Finally, this module selects singular possessives and applies basic rules for identifying relative, personal and elliptical pronouns (in sentence-initial position, after adverbial phrases and after preposition phrases) (Ferrández and Peral, 2000). At this step, each mention belongs to a different entity. Each entity contains the gender, number, head of a noun phrase, head of a Proper Noun (PN) and full proper noun as features. Once the mentions are identified, the coreference resolution modules are sequentially executed.

In order to perform CR, each module applies the following strategy (except for some exceptional rules, explained below): mentions are traversed from the beginning of the text and each one is *selected* if (i) it is not the first mention of the text and (ii) it is the first mention of its entity. Once a mention is selected, it looks backwards for *candidates* in order to find an appropriate antecedent (in the experiments, using the whole text). If an antecedent is found, mentions are merged together in the same entity. Then, the next selected mention is evaluated.

Besides the identification of mentions, current version of LinkPeople contains the following modules:

StringMatch (StM): this pass performs strict matching of the whole string of both mentions (the selected one and the candidate). In the example (Figure 1), mentions 13 and 5 are linked in this step.

NP_Cataphora (NP_C): this module verifies if the first mention —in the first paragraph— is an NP without a personal name. If so, it is considered a cataphoric mention, and the system checks if the next sentence contains a personal name as a Subject. In this case, these mentions are linked if they agree in

gender and number. Mentions 1 and 2 in the example meet these requirements, so they merge. Note that, at the end of this pass, this entity has as NP heads the words 'singer' and 'musician', and 'John Winston Ono Lennon' as the PN. This module also matches fixed synonym structures through dependency paths, such as "Person_A, also known as Person_B".

PN_StMatch (PN_St): in this stage, the system looks for mentions which share the whole PN, even if their heads are different (or if one of them does not have head). "The musician John Lennon" and "John Lennon" (not in Figure 1) would be an example.

PN_Inclusion (PN_I): here, the system verifies if the full PN of the selected mention (in the entity) includes the proper noun of the candidate mention (also in the entity), or vice-versa. In the example, mention 5 is linked to mention 2 in this step. Note that mention 7 is not linked to mention 5, because the full PN of mention 5 is now "John Winston Ono Lennon", not compatible with "Alfred Lennon". Also, mention 13 is not selected here because it is not the first mention of its entity.

PN_Tokens (PN_T): this module splits the full PN of a partial entity in their tokens, and verifies if the full PN of the candidate contains all the tokens in the same order, or vice-versa (except for some *stop words*, such as "Sr.", "Jr.", etc.). As the pair "John Winston Ono Lennon" - "John Winston Lennon" are compatible, mentions 12 and 5 are merged.

HeadMatch (HM): in this step, the system checks if the selected mention and the candidate one share the heads (or the heads of their entities). In Figure 1, mention 14 is linked to mention 13.

Orphan_NP (Orph): the last module of nominal CR applies a pronominal-based rule to orphan noun phrases. Here, a definite NP is marked as orphan if it is still a singleton and it does not contain a personal name. Thus, an orphan NP is linked to the previous PN with gender and number agreement. In the example, the mentions 8/9 are linked to 7 and 6.

Pro_Cataphora (Pro_C): similar to NP_Cataphora, this module verifies if a text starts with a personal (or elliptical) pronoun. If so, it looks in the following sentence if there are a compatible PN.

Pronominal (PRO): this is the standard module for pronominal CR. For each selected pronoun, it verifies if the candidate nominal mentions satisfy the syntactic (and morpho-syntactic) constraints (inspired by Palomar et al. (2001)). They include a set of constraints for each type of pronoun, which remove a candidate if any of the constraints is violated. Some of them are: an object pronoun (direct or indirect) cannot corefer with its subject (mention 11 *vs* mentions 8/9); a personal pronoun does not corefer with a mention inside a prepositional phrase (mention 4 *vs* mention 3), a possessive cannot corefer with the NP it belongs to (mention 10 *vs* mentions 8/9) or a pronoun prefers a subject NP as its antecedent (mentions 10 and 11 *vs* mentions 6 and 7). This way, in Figure 1 the pronominal mention 4 is linked to mention 2, and mentions 10 and 11 to mention 5. This module only looks in the same and previous sentence for candidates.

Pivot_Ent: this last module is only applied if there are orphan pronouns (not linked to any proper noun/noun phrase) at this step. First, it verifies if the text has a pivot entity, which is the more frequent personal name in a text whose frequency is at least 33% higher than the second person with more occurrences. Then, if there is a pivot entity, all the orphan pronouns are linked to its mention. If not, each orphan pronoun is linked to the previous PN/NP (with no constraint).

4 Evaluation

LinkPeople was tested on three different corpora (for Portuguese, Galician, and Spanish) with coreference annotation of person entities (Garcia and Gamallo, 2014). The annotation follows the SemEval-2010 guidelines. The corpus for Portuguese has about 51k tokens and $\approx$ 4,000 mentions. The Galician one, 42k tokens and $\approx$ 3,500 mentions. The Spanish corpus has over 46k tokens, and $\approx$ 4,500 mentions.

Some of the annotation (gender, number and syntactic labeling) was not manually revised, so it may contain errors (*regular setting*). The tests were carried out using a *gold mention* evaluation (i. e., using

as input the corpora with the mentions already identified). Moreover, no external resources (gender dictionaries of proper nouns, WordNet, etc.) were used (*closed setting*).

In order to compare the results of LinkPeople, four well-known baselines were also evaluated: (i) *Singletons* (Stons), where every mention belongs to a different entity. (ii) *All_in_One* (AOne), where all the mentions belong to the same entity; (iii) *HeadMatch* (HMb), which clusters in the same entity mentions sharing the head and classify each pronoun as a singleton, and (iv) *HeadMatch_Pro* (HMP), same as the previous one, but linking each pronoun to the previous nominal mention with gender and number agreement.[6]

Five different metrics were taken into account: MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), $CEAF_{entity}$ (Luo, 2005), BLANC (Recasens and Hovy, 2011) and ConLL (Pradhan et al., 2011). They were computed with the scorers used in SemEval-2010[7] (for BLANC) and ConLL 2011[8] (for the other metrics).

Table 1 contains the results of the four baselines and of LinkPeople in the three corpora. The first block of each language includes the results of the baseline models. The central rows show the results of the different modules of LinkPeople (see Figure 2), added incrementally. The first nine rows (StM $>$ PRO) include two default rules in order to classify mentions not covered by the active modules: (i) nominal mentions not analyzed are singletons and (ii) pronouns are linked to the previous nominal mention with gender agreement (except for those pronouns covered by PRO in this model). Furthermore, PRO systems do not restrict the number of previous sentences while looking for antecedents.

The last model (LinkP, the result of all the modules included in LinkPeople) does include a distance restriction in the Pronominal pass (see Section 3.1), so it combines Pronominal with Pivot_Ent modules.

As expected, *Singletons* and *HeadMatch* baselines produce poor results in most languages and metrics (*Singletons* values in MUC are null because this metric do not reward correctly identified singletons). However, *All_In_One* models achieved reasonable results in some scenarios (MUC and B^3). The differences between these values and those from SemEval-2010 are due to the existence (in this work) of just one type of entity. Journalistic and encyclopedic texts are often focused on just one or two persons, (i.e., there is a much lower number of entities in each text), so the precision is higher in *All_In_One* and lower in *Singletons*.

As Recasens and Hovy (2010) shown, *HeadMatch_Pro* baselines obtain good results in the three languages and with every metric ($\approx$ 60% and 67% in F1 BLANC and CoNLL, respectively).

Concerning the different passes of LinkPeople, the performance of the first matching modules depends on the distribution and structure of PNs and NPs in the corpora. In this respect, PN_StMatch works well in all the contexts. However, PN_Inclusion stands out in the Nominal modules, increasing in more than 5% (BLANC and CoNLL) the performance of the previous model. This is due to the high increase in recall together with the high-precision of this module.

It is worth noting that the addition of some modules seems to improve not only recall, but also precision. This is due to the execution of the two default rules: as the system uses more modules, the amount of (partial)entity mergings (usually) grows. Thus, the precision increases because the new mergings restrict incorrect links performed by the two default rules in the previous models.

HeadMatch module is the first one that deals with mentions without PN (except for the rules applied in NP_Cataphora, with low recall). Due to the knowledge provided by previous modules, it also benefits all the models and languages.

The performance of Orphan_NP and Pro_Cataphora also depends on the corpora and on the evaluation metric. The latter involves a 0.2% loss in Spanish with the BLANC score (but increases in 1.1% using CoNLL). However, Orphan_NP allows the system to not classify as singletons some mentions, which in turn helps to increase the performance of Pronominal modules. Similarly, Pro_Cataphora prevents the next sieve from selecting pronominal mentions that are cataphoric.

[6]Due to language differences and format issues, other coreference resolution systems could not be used for comparison (Raghunathan et al., 2010; Sapena et al., 2013).

[7]http://www.lsi.upc.edu/~esapena/downloads/scorer-v1.04.zip

[8]http://conll.cemantix.org/download/reference-coreference-scorers.v7.tar.gz

Lang	Model	MUC			B^3			$CEAF_e$			BLANC			CoNLL
		R	P	F1	R	P	F1	R	P	F1	R	P	F1	F1
Port.	Stons	-	-	-	15.0	100	26.1	65.3	10.9	18.7	50.0	29.0	36.7	14.9
	AOne	93.8	85.5	**89.4**	94.8	47.5	63.3	11.9	78.1	20.7	50.0	21.0	29.1	57.8
	HMb	26.5	93.9	41.3	22.2	97.9	36.2	72.3	16.1	26.4	53.6	78.5	44.2	34.6
	HMP	76.0	91.2	82.9	46.0	85.8	59.9	76.7	49.2	59.9	68.5	80.0	68.1	67.6
	StM	69.8	91.5	79.2	38.8	88.7	54.0	78.1	40.5	53.3	64.7	79.2	62.9	62.2
	NP_C	70.4	91.4	79.6	39.2	88.5	54.3	78.3	41.5	54.3	64.7	79.2	62.9	62.7
	PN_St	72.8	91.9	81.3	40.9	88.3	55.9	79.3	44.7	57.2	65.0	79.2	63.4	64.8
	PN_I	77.1	92.5	84.1	50.5	87.5	64.0	81.9	52.7	64.1	71.1	81.0	71.2	70.8
	PN_T	77.3	92.5	84.2	50.8	87.5	64.3	82.0	53.0	64.4	71.1	81.0	71.3	71.0
	HM	79.7	92.3	85.6	53.6	85.5	65.9	81.3	58.3	67.9	71.5	80.7	71.7	73.1
	Orph	83.4	91.8	87.4	58.1	82.7	68.3	81.4	70.2	75.4	71.6	80.3	71.9	77.0
	ProC	83.4	91.8	87.4	58.1	82.7	68.3	81.4	70.3	**75.5**	71.6	80.3	72.0	77.0
	PRO	81.8	91.7	86.4	59.1	83.9	69.3	82.7	66.5	73.7	76.0	83.7	76.7	76.5
	LinkP	82.7	92,7	87.4	65.8	84.5	**74.0**	84.4	67.9	75.2	83.6	85.4	**84.2**	**78.9**
Gal.	Stons	-	-	-	14.6	100	25.4	71.7	11.0	19.1	50.0	28.4	36.3	14.8
	AOne	96.6	86.0	91.0	97.1	53.9	69.3	9.0	82.7	16.2	50.0	21.6	30.1	58.8
	HMb	21.1	90.5	34.2	20.2	97.5	33.5	74.1	14.3	24.0	51.3	74.7	39.1	30.6
	HMP	81.9	89.8	85.7	44.1	83.6	57.7	70.0	53.5	60.6	61.3	76.5	57.9	68.0
	StM	77.1	90.6	83.3	36.5	86.7	51.4	75.1	45.5	56.6	58.9	76.9	53.7	63.8
	NP_C	77.6	90.7	83.6	37.2	86.7	52.1	75.2	46.2	57.3	59.2	77.0	54.3	64.3
	PN_St	79.0	90.9	84.6	39.1	86.2	53.8	75.6	48.8	59.3	59.7	77.0	55.1	65.9
	PN_I	83.1	91.5	87.1	46.7	85.3	60.4	76.7	57.8	66.0	62.5	77.5	59.5	71.1
	PN_T	83.3	91.5	87.2	48.2	85.3	61.6	76.9	58.6	66.5	63.2	77.9	60.5	71.8
	HM	84.6	91.6	87.9	49.8	84.4	62.6	76.8	62.0	68.6	63.4	77.5	60.8	73.1
	Orph	84.7	91.3	87.9	49.9	83.9	62.6	76.8	63.2	69.4	63.3	77.3	60.8	73.3
	ProC	84.7	91.3	87.9	49.1	83.9	62.6	76.8	63.2	69.4	63.3	77.3	60.8	73.3
	PRO	86.9	92.5	89.6	60.7	86.8	71.4	82.8	72.2	77.1	73.6	82.0	73.9	79.4
	LinkP	89.0	94.6	**91.7**	72.9	88.4	**79.9**	87.6	76.6	**81.7**	82.7	85.8	**83.4**	**84.4**
Spa.	Stons	-	-	-	10.9	100	19.7	69.5	8.7	15.4	50.0	29.4	37.0	11.7
	AOne	91.7	88.4	**90.0**	92.6	51.3	66.0	6.4	83.0	11.9	50.0	20.6	29.2	55.9
	HMb	20.7	94.2	34.0	15.4	98.0	26.6	75.4	11.9	20.6	51.3	74.6	39.9	27.0
	HMP	78.2	90.7	84.0	35.3	81.2	49.2	72.9	51.5	60.4	59.3	74.7	55.5	64.5
	StM	73.9	90.7	81.4	30.1	83.7	44.3	73.9	41.6	53.3	58.6	75.6	54.1	59.7
	NP_C	74.1	90.7	81.5	30.2	83.7	44.4	73.9	42.0	53.6	58.6	75.6	54.1	59.8
	PN_St	75.4	91.0	82.5	31.2	83.1	45.4	73.8	44.1	55.2	58.6	75.4	54.3	61.0
	PN_I	78.8	91.7	84.8	39.3	82.2	53.1	75.9	52.8	62.3	62.0	76.7	59.6	66.7
	PN_T	79.0	91.7	84.9	40.0	82.1	53.8	76.0	53.3	62.7	62.6	76.3	60.5	67.1
	HM	80.5	92.0	85.9	41.7	80.9	55.1	75.6	57.3	65.2	63.1	75.0	61.4	68.7
	Orph	81.1	91.9	86.1	42.3	80.5	55.5	75.4	59.8	66.7	63.2	75.0	61.6	69.4
	ProC	82.3	91.9	86.8	43.2	79.6	56.0	74.6	64.1	68.9	63.0	74.7	61.4	70.6
	PRO	82.6	92.4	87.2	46.0	80.8	58.7	77.5	65.8	71.2	66.8	77.9	66.2	72.4
	LinkP	84.1	94.1	88.8	62.9	84.8	**72.2**	83.4	71.0	**76.7**	81.7	84.9	**82.6**	**79.2**

Table 1: Results of LinkPeople compared to the baselines in Portuguese (Port.), Galician (Gal.) and Spanish (Spa.). *LinkP* contains the results of the execution of the whole system.

The standard pronominal resolution module also increases the accuracy of all the systems (with the only exception in Portuguese language with the CoNLL score, which also had a high increase with the Orphan_NP module).

Finally, one of the main contributions to the performance of LinkPeople is the combination of the Pronominal module with the Pivot_Ent one. This combination reduces the scope of the Pronominal module, thus strengthening the impact of syntactic constraints. Furthermore, Pivot_Ent looks for a prominent person entity in each text, and links the orphan pronouns to this entity. In the three languages, the improvement is noticeably better with the BLANC score.

Last row of each language shows the current results of LinkPeople in the three corpora, with macro-average values of $\approx 83\%$ and $\approx 81\%$ with BLANC and CoNLL scores, respectively.

5 Error Analysis

In order to determine the major classes of errors produced by the system, 150 errors (50 for each language) were randomly selected from the output of LinkPeople. Each error was analyzed in order to find

its source, and was classified according to its typology. This section shows the different error typologies together with some examples, sorted by their frequency in the corpora (first percentage in parenthesis is the average frequency, while the other three correspond to Portuguese, Galician and Spanish values, respectively).[9] They are real examples of incorrectly analyzed mentions (or pairs of mentions belonging to the same entity), with some simplifications due to space reasons:

5.1 Missing links between Noun Phrases and/or Proper Nouns (46%: 58% / 32% / 48%)

This category includes some error typologies that differ in the type of knowledge and analysis required by the system in order to accurately link two mentions:

Synonym heads (35.3%: 48% / 32% / 26%): The most frequent type of missing links was produced by mentions of the same entity whose heads are synonyms:

Mention A: "El *joven*" (the *young*)
Mention B: "el *muchacho*" (the *boy*)

External (real-world) knowledge (6%: 0% / 0% / 18%): This class includes mentions of the same entity which do not share the lexical features, usually because they refer to well-known entities in the real world:

Mention A: "la presidenta" (the president)
Mention B: "Cristina Kirchner"

Here, the noun phrase "the president" is used to refer "Cristina Kirchner", but the mentions are not linked because the system does not take advantage of resources that define Cristina Kirchner as a *president*.

Semantic knowledge (2.7%: 4% / 0 % / 4%): Lack of other type of semantic knowledge, such as hyponym-hypernym pairs, also involves missing links like the following:

Mention A: "o escocês" (the scotish)
Mention B: "o britânico" (the british)

Head modifiers (1.3%: 4% / 0 % / 0%): Internal modifiers of some heads may also produce missing links, as in the following example, where a mention does not contain the modifier *adjunto* (vice):

Mention A: "o *ministro* (the *minister*)
Mention B: "o *ministro-adjunto*" (the *vice-minister*)

Spelling differences (0.7%: 2% / 0% / 0%): Some personal names are spelled differently in the same text:

Mention A: "André *Villas-Boas*"
Mention B: "André *Villas Boas*"

5.2 Errors due to incorrect predicted (syntactic and morpho-syntactic) analysis (15.3%: 2% / 22% / 22%)

Since the corpora do not have PoS-tagging and dependency labels fully revised, some of these errors involve missing and spurious links between mentions.

Errors in syntactic constraints (10.7%: 0% / 16% / 16%): Direct and indirect object pronouns incorrectly labeled are not covered by some of the syntactic constraints, thus involving an incorrect link between a pronoun and its subject noun phrase.

[9] The results of 0% in some languages and categories do not mean that these languages cannot have those error typologies, but they did not appear due to the small number of errors evaluated.

Incorrect gender (2.7%: 2% / 4% / 2%): The gender of some nouns and adjectives also can be wrongly labeled, so other mentions may be incorrectly linked, or involve a missing link. For instance, the word *atleta* (sportsperson, which can be both masculine or feminine), labeled as masculine blocked a link to the feminine pronoun *ela* (she) in Galician.

Incorrect head (2%: 0% / 2% / 4%): Errors in PoS-tagging (usually between nouns and adjectives) also produce wrong dependency analysis, which in turn involve incorrect extractions of the NP heads:

> Mention: "el jugador alemán" (the german player)
> Extracted Head: *alemán (*german, instead of *jugador/player*)

5.3 Missing links due to long distance pronominal anaphora (11.3%: 14% / 18% / 2%)

This kind of errors arises when the distance between a pronoun and its nominal antecedent is outside the scope of a rule (in our case, between two and four sentences, depending on the module), and the antecedent is not the pivot entity.

5.4 Errors due to quoted speech coreference (10%: 10% / 14% / 6%)

Another category of errors includes mentions inside quoted speech. These mentions can refer to the speaker (first person) or to a third person in the quoted speech:

First person (4.7%: 6% / 6% / 2%): The 1st person of the quoted speech should be linked to the speaker instead of to a previous entity (note that the elliptical pronoun might also be a 3rd person pronoun):

> "Si $\emptyset_{1^{st}}$ tuviera que redactar [...]", resumió $\text{Lezcano}_{\text{Speaker}}$.
> "If [$I_{1^{st}}$] had to write [...]", $\text{Lezcano}_{\text{Speaker}}$ summarized.

Third person (5.3%: 4% / 8% / 4%): 3rd persons of a quoted speech should not be linked to the speaker:

> $\text{Gustavo}_{\text{Speaker}}$: "Cuando $\text{yo}_{1^{st}}$ me fui, $\text{él}_{3^{rd}}$ dejó Boca."
> $\text{Gustavo}_{\text{Speaker}}$: "When $I_{1^{st}}$ quit, $\text{he}_{3^{rd}}$ left Boca."

5.5 Spurious links in plural mentions (5.3%: 4% / 4% / 8%)

Coreference of plural mentions was performed through basic links to the previous entities, producing incorrect classifications. Also, some plural mentions include entities with different genders (e.g., *amigos* —friends— may refer to feminine and masculine entities, but the grammatical gender of the word is masculine in the three analyzed languages):

> $_1$[Hulk]$_1$, $_2$[Moutinho]$_2$ e $_3$[Álvaro Pereira]$_3$ na lista de compra de $_4$[Villas-Boas]$_4$ [...]. $_{5/6/7}$[O trio do F.C. Porto]$_{2/3/*4}$ [...].
>
> $_1$[Hulk]$_1$, $_2$[Moutinho]$_2$ and $_3$[Álvaro Pereira]$_3$ in the shopping list of $_4$[Villas-Boas]$_4$ [...]. $_{5/6/7}$[The F.C. Porto trio]$_{2/3/*4}$ [...].

In this example, the plural mention (*O trio do F.C. Porto*) is linked to the previous nominal mentions with gender agreement, so an incorrect link between mentions 7 and 4 is done.

5.6 Errors due to incorrect gender agreement (4.7%: 4% / 4% / 6%)

Some nominal phrases referring to the same entity may have different gender, thus causing wrong links:

> Mention A: la víctima (the victim: feminine)
> Mention B: el muchacho (the boy: masculine)

5.7 Errors produced by constraints and Pivot_Ent modules (4.6%: 6% / 0% / 8%)

The syntactic constraints, although precise, may restrict some correct links. This can involve (i) an incorrect discourse analysis or (ii) the application of Pivot_Ent, linking the mention to the most frequent entity, which might be incorrect:

$_1$[El escritor]$_1$ tuvo que visitar a $_2$[Martín]$_2$ en el hotel. Según $_3\emptyset_{*1}$ dijo [...]
$_1$[The writer]$_1$ had to visit $_2$[Martín]$_2$ in the hotel. As $_3$[he]$_{*1}$ said [...]

Here, the elliptical subject of *dijo* (said) is *Martín*, but the link is blocked due to a syntactic constraint: the antecedent of the (subject) elliptical pronoun should be a subject. Thus, the system incorrectly links mention 3 to mention 1.

5.8 Spurious links between Noun Phrases sharing the same head (1.3%: 0% / 4% / 0%)

In the same text, different entities can share their heads in some mentions, which may involve errors in coreference links, depending on their position and on their features. Thus, the NP "the president" may be linked to two different persons like "the president of the Academia" and "the president of the Government".

5.9 Spurious links produced by errors in previous modules (0.7%: 0% / 2% / 0%)

First modules also produce some incorrect clusters which involve errors in further modules. For instance, in the Galician corpus, NP_Cataphora incorrectly linked the noun phrase *o alcalde* (the mayor) to the proper noun "Dorribo". Then, HeadMatch merged "Dorribo" with *o alcalde Orozco*, creating an incorrect entity that contains two different persons (Dorribo and Orozco).

5.10 Errors due to fixed language structures (0.7%: 2% / 0% / 0%)

Other minor errors include some fixed structures such as the following cataphoric possessive:

Por $_1$[sua]$_1$ parte, $_2$[Cristina]$_{*2}$ [...]
For $_1$[her]$_1$ part, $_2$[Cristina]$_{*2}$ [...]

The results of the error analysis bring interesting information to further work. Thus, including some kind of semantic knowledge (synonyms), improving pronominal coreference resolution and implementing specific rules for quoted speech might solve many of the most frequent errors made by LinkPeople.

6 Conclusions and Further Work

This paper presents the first version of LinkPeople, an open-source entity-centric approach for coreference resolution of person entities which applies a battery of deterministic modules enriched with precise linguistic information.

The system was evaluated in three different languages (Portuguese, Galician and Spanish), clearly surpassing some powerful baselines and achieving promising results.

The addition of rules focused on cataphoric coreference as well as pronominal constraints based on syntactic and discourse restrictions increases the performance of similar approaches with lack of this kind of knowledge.

Current work explores better nominal (Elsner and Charniak, 2010) and pronominal constraints and dedicated handling of plural mentions. In further work, the implementation of an inheritance constraint is planned, which could prevent the merging of partial entities if their mentions were blocked by previous modules. Moreover, the extension of the system for solving the coreference of other types of entities is also planned.

Acknowledgments

This work has been supported by the HPCPLN project – Ref: EM13/041 (Galician Government) and by the Celtic – Ref: 2012-CE138 and Plastic – Ref: 2013-CE298 projects (Feder-Interconnecta).

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Workshop on Linguistic Coreference at the International Language Resources and Evaluation Conference (LREC 1998)*, volume 1, pages 563–566.

Breck Baldwin. 1997. CogNIAC: high precision coreference with limited knowledge and linguistic resources. In *Proceedings of a Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, pages 38–45. Association for Computational Linguistics.

Kalina Bontcheva, Marin Dimitrov, Diana Maynard, Valentin Tablan, and Hamish Cunningham. 2002. Shallow Methods for Named Entity Coreference Resolution. In *Proceedings of the Workshop on Chaines de références et résolveurs d'anaphores at Traitement Automatique des Langues Naturelles (TALN 2002)*.

Xavier Carreras, Lluís Márquez, and Lluís Padró. 2003. A simple named entity extractor using AdaBoost. In *Proceedings of the 7th Conference on Computational Natural Language Learning (CoNLL 2003): Shared Task*, volume 4, pages 152–155. Association for Computational Linguistics.

Amanda Rocha Chaves and Lucia Helena Machado Rino. 2007. A resolução de pronomes anafóricos do português com base em heurísticas que apontam o antecedente. In *Proceedings of VI Congresso de Pós-Graduação da UFSCar*, volume 2, pages 1272–1273, São Carlos, São Paulo.

Thiago Thomes Coelho and Ariadne Maria Brito Rizzoni Carvalho. 2005. Uma adaptação do algoritmo de Lappin e Leass para resolução de anáforas em português. In *III Workshop em Tecnologia da Informação e da Linguagem Humana–TIL. Proceedings of XXV Congresso da SBC*, pages 2069–2078.

William W. Cohen, Pradeep Ravikumar, and Stepehn G. Fienberg. 2003. A Comparison of String Distance Metrics for Name-Matching Tasks. In *Proceedings of the IJCAI 2003 Workshop on Information Integration on the Web*, pages 73–78.

Ramon Ré Moya Cuevas and Invandré Paraboni. 2008. A machine learning approach to portuguese pronoun resolution. In *Advances in Artificial Intelligence (IBERAMIA 2008)*, pages 262–271. Springer-Verlag.

José Guilherme C. de Souza, Patrícia Gonçalves, and Renata Vieira. 2008. Learning Coreference Resolution for Portuguese Texts. In *Computational Processing of the Portuguese Language (PROPOR 2008)*, pages 153–162. Springer-Verlag.

Micha Elsner and Eugene Charniak. 2010. The same-head heuristic for coreference. In *Proceedings of the 48th Association for Computational Linguistics Conference Short Papers (ACL 2010)*, pages 33–37. Association for Computational Linguistics.

Antonio Ferrández and Jesús Peral. 2000. A computational approach to zero-pronouns in Spanish. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (ACL 2000)*, pages 166–172. Association for Computational Linguistics.

Pablo Gamallo and Marcos Garcia. 2011. A resource-based method for named entity extraction and classification. In *Progress in Artificial Intelligence (LNCS/LNAI)*, volume 7026/2011, pages 610–623, Berlin. Springer-Verlag.

Pablo Gamallo and Isaac González López. 2011. A Grammatical Formalism Based on Patterns of Part-of-Speech Tags. *International Journal of Corpus Linguistics*, 16(1):45–71.

Marcos Garcia and Pablo Gamallo. 2010. Análise Morfossintáctica para Português Europeu e Galego: Problemas, Soluções e Avaliação. *Linguamática*, 2(2):59–67.

Marcos Garcia and Pablo Gamallo. 2013. Exploring the Effectiveness of Linguistic Knowledge for Biographical Relation Extraction. *Natural Language Engineering*. Available on CJO 2013 doi:10.1017/S1351324913000314.

Marcos Garcia and Pablo Gamallo. 2014. Multilingual corpora with coreferential annotation of person entities. In *Proceedings of the 9th edition of the Language Resources and Evaluation Conference (LREC 2014)*, pages 3229–3233. European Language and Resources Association.

Marcos Garcia, Iria Gayo, and Isaac González López. 2012. Identificação e Classificação de Entidades Mencionadas em Galego. *Estudos de Lingüística Galega*, 4:13–25.

Aria Haghighi and Dan Klein. 2007. Unsupervised coreference resolution in a nonparametric bayesian model. In *Proceedings of the 45th Annual Meeting on Association for Computational Linguistics (ACL 2007)*, volume 45, pages 848–855. Association for Computational Linguistics.

Shalom Lappin and Herbert J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational linguistics*, 20(4):535–561.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Xiaoqiang Luo. 2005. On Coreference Resolution Performance Metrics. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, pages 25–32. Association for Computational Linguistics.

Gideon S. Mann. 2002. Fine-grained proper noun ontologies for question answering. In *Proceedings of the 2002 workshop on Building and using semantic networks*, volume 11. Association for Computational Linguistics.

Joseph McCarthy and Wendy G. Lehnert. 1995. Using decision trees for coreference resolution. In *Proceedings of the 14th International Conference on Artificial Intelligence*, pages 1050–1055.

Ruslan Mitkov. 1998. Robust pronoun resolution with limited knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL/COLING 1998)*, volume 2, pages 869–875. Association for Computational Linguistics.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 104–111. Association for Computational Linguistics.

Vincent Ng. 2008. Unsupervised models for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 640–649. Association for Computational Linguistics.

Lluís Padró and Evgeny Stanilovsky. 2012. FreeLing 3.0: Towards Wider Multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Turkey. European Language and Resources Association.

Manuel Palomar, Antonio Ferrández, Lidia Moreno, Patricio Martínez-Barco, Jesús Peral, Maximiliano Saiz-Noeda, and Rafael Muñoz. 2001. An algorithm for anaphora resolution in Spanish texts. *Computational Linguistics*, 27(4):545–567.

Ivandré Paraboni. 1997. Uma arquitetura para a resolução de referências pronominais possessivas no processamento de textos em língua portuguesa. Master's thesis, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre.

Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT/NAACL 2006)*, pages 192–199. Association for Computational Linguistics.

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 Shared Task: Modeling Unrestricted Coreference in OntoNotes. In *Proceedings of the 15th Conference on Computational Natural Language Learning (CoNLL 2011): Shared Task*, pages 1–27. Association for Computational Linguistics.

Kathik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 492–501. Association for Computational Linguistics.

Marta Recasens and Eduard Hovy. 2009. A deeper look into features for coreference resolution. In *Anaphora Processing and Applications*, pages 29–42. Springer-Verlag.

Marta Recasens and Eduard Hovy. 2010. Coreference resolution across corpora: Languages, coding schemes, and preprocessing information. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 1423–1432. Association for Computational Linguistics.

Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand Index for Coreference Evaluation. *Natural Language Engineering*, 17(4):485–510.

Marta Recasens and M. Antònia Martí. 2010. AnCora-CO: Coreferentially annotated corpora for Spanish and Catalan. *Language Resources and Evaluation*, 44.4:315–345.

Marta Recasens, Lluís Màrquez, Emili Sapena, M. Antònia Martí, Mariona Taulé, Véronique Hoste, Massimo Poesio, and Yannick Versley. 2010. SemEval-2010 Task 1: Coreference resolution in multiple languages. In *Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval'10)*, pages 1–8. Association for Computational Linguistics.

Emili Sapena, Lluís Padró, and Jordi Turmo. 2013. A Constraint-Based Hypergraph Partitioning Approach to Coreference Resolution. *Computational Linguistics*, 39(4).

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544.

Veselin Stoyanov and Jason Eisner. 2012. Easy-first coreference resolution. In *Proceedings of the International Conference on Computational Linguistics (COLING 2012)*, pages 2519–2534.

Olga Uryupina. 2007. *Knowledge acquisition for coreference resolution*. Ph.D. thesis, Universität des Saarlandes.

Renata Vieira and Massimo Poesio. 2000. An empirically based system for processing definite descriptions. *Computational Linguistics*, 26(4):539–593.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of Message Understanding Conference 6 (MUC-6)*, pages 45–52. Association for Computational Linguistics.

Unsupervised Multiword Segmentation of Large Corpora using Prediction-Driven Decomposition of *n*-grams

Julian Brooke[*†] **Vivian Tsang**[†] **Graeme Hirst**[*] **Fraser Shein**[*†]

[*]Department of Computer Science
University of Toronto
jbrooke@cs.toronto.edu
gh@cs.toronto.edu

[†]Quillsoft Ltd.
Toronto, Canada
vtsang@quillsoft.ca
fshein@quillsoft.ca

Abstract

We present a new, efficient unsupervised approach to the segmentation of corpora into multiword units. Our method involves initial decomposition of common *n*-grams into segments which maximize within-segment predictability of words, and then further refinement of these segments into a multiword lexicon. Evaluating in four large, distinct corpora, we show that this method creates segments which correspond well to known multiword expressions; our model is particularly strong with regards to longer (3+ word) multiword units, which are often ignored or minimized in relevant work.

1 Introduction

Identification of multiword units in language is an active but increasingly fragmented area of research, a problem which can limit the ability of others to make use of units beyond the level of the word as input to other applications. General research on word association metrics (Church and Hanks, 1990; Smadja, 1993; Schone and Jurafsky, 2001; Evert, 2004; Pecina, 2010), though increasingly comprehensive in its scope, has mostly failed to identify a single best choice, leading some to argue that the variety of multiword phenomena must be tackled individually. For instance, there is a body of research focusing specifically on collocations that are (to some degree) non-compositional, i.e. multiword expressions (Sag et al., 2002; Baldwin and Kim, 2010), with individual projects often limited to a particular set of syntactic patterns, e.g. verb-noun combinations (Fazly et al., 2009). A major issue with approaches involving statistical association is that they rarely address expressions larger than 2 words (Heid, 2007); in corpus linguistics, larger sequences referred to as *lexical bundles* are extracted using an *n*-gram frequency cutoff (Biber et al., 2004), but the frequency threshold is typically set very high so that only a very limited set is extracted. Another drawback, common to almost all these methods, is that they rarely offer an explicit segmentation of a text into multiword units, which would be preferable for downstream uses such as probabilistic distributional semantics. An exception is the Bayesian approach of Newman et al. (2012), but their method does not scale well (see Section 2). Our own long-term motivation is to identify a wide variety of multiword units for assisting language learning, since correct use of collocations is known to pose a particular challenge to learners (Chen and Baker, 2010).

Here, we present a multiword unit segmenter[1] with the following key features:

- It is entirely unsupervised.
- It offers both segmentation of the input corpus and a lexicon which can be used to segment new corpora.
- It is scalable to very large corpora, and works for a variety of corpora.
- It is language independent.

[1]The software is available at http:/www.cs.toronto.edu/~jbrooke/ngram_decomp_seg.py .

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 753–761, Dublin, Ireland, August 23-29 2014.

- It does not inherently limit possible units with respect to part-of-speech or length.
- It has a bare minimum of parameters, and can be used off-the-shelf: in particular, it does not require the choice of an arbitrary cutoff for some uninterpretable statistical metric.
- It does, however, include a parameter fixing the minimum number of times that a valid multiword unit will appear in the corpus, which ensures sufficient usage examples for relevant applications.

Our method involves three major steps: extraction of common n-grams, initial segmentation of the corpus, and a refinement of the resulting lexicon (and, by extension, the initial segmentation). The latter two steps are carried out using a simple but novel heuristic based on maximizing word prediction within multiword segments. Importantly, our method requires just a few iterations through the corpus, and in practice these iterations can be parallelized. Evaluating with an existing set of multiword units from WordNet in four large corpora from distinct genres, we show that our initial segmentation offers extremely good subsumption of known collocations, and after lexicon refinement the model offers a good trade-off between subsumption and exact matches. We also evaluate a sample of our multiword vocabulary using crowdsourcing, and offer a qualitative analysis.

2 Related Work

In computational linguistics, there is a large body of research that proposes and/or evaluates lexical association measures for the creation of multiword lexicons (Church and Hanks, 1990; Smadja, 1993; Schone and Jurafsky, 2001; Evert, 2004): there are many more measures than can be addressed here—work by Pecina (2010) considered 82 variations—but popular choices include the t-test, log likelihood, and pointwise mutual information (PMI). In order to build lexicons using these methods, particular syntactic patterns and thresholds for the metrics are typically chosen. Many of the statistical metrics do not generalize at all beyond two words, but PMI (Church and Hanks, 1990), the log ratio of the joint probability to the product of the marginal probabilities, is a prominent exception. Other measures specifically designed to address collocations of larger than two words include the c-value (Frantzi et al., 2000), a metric designed for term extraction which weights term frequency by the log length of the n-gram while penalizing n-grams that appear in frequent larger ones, and mutual expectation (Dias et al., 1999), which produces a normalized statistic that reflects how much a candidate phrase resists the omission of any particular word. Another approach is to simply to combine known $n-1$ collocations to form n-length collocations (Seretan, 2011), but this is based on the assumption that all longer collocations are built up from shorter ones—idioms, for instance, do not usually work in that way.

An approach used in corpus linguistics which does handle naturally longer sequences is the study of *lexical bundles* (Biber et al., 2004), which are simply n-grams that occur above a certain frequency threshold. This includes larger phrasal chunks that would be missed by traditional collocation extraction, and so research in this area has tended to focus on how particular phrases (e.g. *if you look at*) are indiciative of particular genres (e.g. university lectures). In order to get very reliable phrases, the threshold is typically set high enough (Biber et al. use 40 occurrences in 1 million words) to filter out the vast majority of expressions in the process.

With respect to the features of our model, the work closest to ours is probably that of Newman et al. (2012). Like us, they offer an unsupervised solution, in their case a generative Dirichlet Process model which jointly creates a segmentation of the corpus and a multiword term vocabulary. Their method, however, requires full Gibbs sampling with thousands of iterations through the corpus (Newman et al. report using 5000), an approach which is simply not tractable for the large corpora that we address in this paper (which are roughly 1000 times larger than theirs). Though the model is general, their focus is limited to term extraction, and for larger terms they compare only with the c-value approach of Frantzi et al. (2000). Other closely related work includes general tools available for creating multiword lexicons using association measures or otherwise exploring the collocational behavior of words (Kilgarriff and Tugwell, 2001; Araujo et al., 2011; Kulkarni and Finlayson, 2011; Pedersen et al., 2011). Other related but distinct tasks include syntactic chunking (Abney, 1991) and word segmentation for Asian languages, in particular Chinese (Emerson, 2005).

3 Method

3.1 Prediction-based segmentation

Our full method consists of multiple independent steps, but it is based on one central and relatively simple idea that we will introduce first. Given a sequence of words, $w_1 \dots w_n$, and statistics (i.e. n-gram counts) about the use of these words in a corpus, we first define $p(w_i|w_{j,k})$ as the conditional probability of some word w_i appearing with some contextual subsequence $w_j \dots w_{i-1}, w_{i+1} \dots w_k, 1 \leq j \leq i \leq k \leq n$. In the case $i = j = k$, this is simply the marginal probability, $p(w_i)$. We then define the word predictability of some w_i in the context $w_{1,n}$ as the log of the maximal conditional probability of the word across all possible choices of j and k:

$$pred(w_i, w_{1,n}) = \max_{j,k} \log p(w_i|w_{j,k})$$

We can define predictability for the entire sequence then as:

$$pred(w_{1,n}) = \sum_{i=1}^{n} pred(w_i, w_{1,n})$$

Now we consider the case where we have a set of possible segmentations S of the sequence, where each segmentation $s \in S$ can be viewed as a (possibly empty) set of segment boundaries $\langle s_0, s_1, \dots, s_m \rangle$. Among the available options, our optimal segmentation is:

$$\arg\max_{s \in S} \sum_{i=0}^{m-1} pred(w_{s_i, s_{i+1}-1})$$

That is, we will prefer the segmentation which maximizes the overall predictability of each word in the sequence, under the restriction that we only predict words using the context within their segments. This reflects our basic assumption that words within a good segment, i.e. a multiword unit, are (much) more predictive of each other than words outside a unit. Note that if our probabilities are calculated from the full set of n-gram counts for the corpus being segmented and the set of possible segmentations S is not constrained, a segmentation with a smaller number of breaks will generally be preferred over one with more breaks. However, in practice we will be greatly constraining S and also using probabilities based on only a subset of all the information in the corpus.

3.2 Extraction of n-grams

In order to carry out a segmentation of the corpus using this method, we first need to extract statistics in the form of n-gram counts. Given a minimum occurrence threshold, this can be done efficiently even for large corpora in an iterative fashion until all n-grams have been extracted. For all our experiments here, we limit ourselves to n-grams that appear at least once in 10 million tokens, and we did not collect n-grams for $n > 10$ (which are almost always the result of duplication of texts in the corpus). For the purposes of calculating conditional probabilities given surrounding context in our predictive segmentation, we collected both standard n-grams as well as (for $n \geq 3$) skip n-grams with a missing word (e.g. *basic* * *processes* where the asterisk indicates that any word could appear in that slot). Here we use lower-cased unlemmatized tokens, excluding punctuation, though for languages with more inflectional morphology than English, lemmatization would be advised.

3.3 Initial segmentation

Given these n-gram statistics, our initial segmentation proceeds as follows: For each sentence in the corpus, we identify all maximum length n-grams in the sentence, i.e. all those n-grams for $n \geq 2$ where there is no larger n-gram which contains them while still being above our threshold of occurrence. These n-grams represent the upper bound of our segmentation: we will never break into segments larger than these. However, there are many overlaps among these n-grams (in fact, with a low threshold the vast majority of n-grams overlap with at least one other), and for proper segmentation we need to resolve

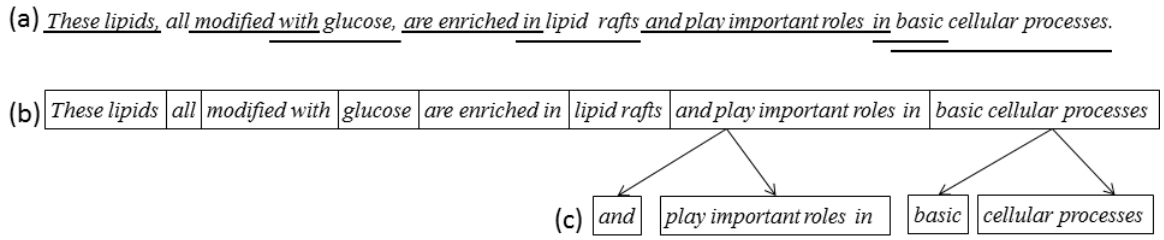


Figure 1: Three-step procedure for n-gram decomposition into multiword units. a) shows the maximal n-grams identified in the sentence, b) is the segmentation after the initial pass of the corpora, and c) shows further decomposition of segments after a pass through the lexicon resulting from b).

all overlaps between these maximal n-grams by inserting at least one break. For this we apply our prediction-based decomposition technique. In our discussion in Section 3.1, we did not consider how the possible segmentations were selected, but now we can be explicit: the set S consists of all possible segmentations which minimally resolve all n-gram overlaps. By minimally resolve, we mean that the removal of any breakpoint from our set would result in an unresolved overlap: in short, there are no extraneous breaks, and therefore no cases where a possible set of breaks is a subset of another possible set. Figure 1a shows a real example: if we just consider the last three maximal n-grams, there are two possible minimal breaks: a single break between *in* and *basic* or two breaks, one between *roles* and *in* and one between *basic* and *cellular*.

Rather than optimizing over all possible breaks over the whole sentence, which is computationally problematic, we simplify the algorithm somewhat by moving sequentially through each n-gram overlap in the sentence, taking any previous breaks as given while considering only the minimum breaks necessary to resolve any overlaps that directly influence the segmentation of the two overlapping spans under consideration, which is to say any other overlapping spans which contain at least one word also contained in at least one of overlapping spans under consideration. For example, in Figure 1a we first deal independently with each of the first two overlaps (the spans *modified with glucose* and *are enriched in lipid rafts*, and then we consider the final two overlaps together: The result is shown in Figure 1b. In development, we tested including more context (i.e. considering second-order influence) and found no benefit. Since we do not consider breaks other than those required to resolve overlapping n-grams, these segments tend to be long. This is by design; our intention is that these segments will subsume as many multiword units as possible, and therefore will be amenable to refinement by further decomposition in the next step.

3.4 Lexicon decomposition

Based on the initial segmentation of the entire corpus, we extract a tentative lexicon, with corresponding counts. Then, in order from longest to shortest, we consider decomposition of each entry. First, using our prediction-based decomposition method, we find the best decomposition of the entry into two parts; note that we only need to consider one break per lexicon entry, since breaks in the (smaller) parts will be considered independently later in the process. If the count in our lexicon is below the occurrence threshold, we always carry out this split, which means we remove the entry from the lexicon and (after all n-grams of that length have been processed, so as to avoid ordering effects) add its counts to the counts of n-grams of its best decomposition. If the count is above the threshold, we preserve the full entry (for entries of length 3 or greater) only if the following inequality is true for each subsegment $w_{j,k}$ in the full entry $w_{1,n}$:

$$\sum_{i=j}^{k} pred(w_i, w_{1,n}) - pred(w_{j,k}) > \log p(w_{j,k}) - \log p(w_{1,n})$$

That is, the ratio (expressed as a difference of logarithms) between the count of the segment and the full unsegmented entry (in our preliminary lexicon) is lower than the ratio of the predictability (as defined in our discussion of prediction-based decomposition) of the words in the segment with the context of the full entry to the predictability of words with only the context included within the segment (which is just $pred(w_{j,k})$). In other words, we preserve only longer multiword sequences in our lexicon when any decrease in the probability of the full entry relative to its smaller components[2] is fully offset by an increase in the conditional probability of the individual words of that segment when the larger context from the full segment is available. For example, after we have decided on a potential break in the phrase *basic | cellular process* from our example in Figure 1, we compare the (marginal) probability "lost" by including *basic* in a larger phrase, i.e. the ratio of counts of *basic* to *basic cellular process* in our lexicon), to the (conditional) probability "gained" by how much more predictable the segment is in this context; when the segment in question is a single word, as in this case, this is simply $p(basic|cellular\ process)/p(basic)$, and we break only when there is more gain than loss. This restriction could be parameterized for more fine-grained control of the trade-off between larger and smaller segments in specific cases, but in the interest of avoiding nuisance parameters we just use it directly. Once we have decomposed all eligible entries to create a final lexicon, we apply these same decompositions to the segments in our initial segmentation to produce a final segmentation (see Figure 1c).

4 Evaluation

Multiword lexicons are typically evaluated in one of two ways: direct comparison to an existing lexicon, or precision of the top n candidates offered by the model. There are problems with both these methods, since there are no resources that offer a truly comprehensive treatment of multiword units, defined broadly, and the top n candidates from a model for small n may not be a particularly representative sample: in particular, they might not include more common terms, which should be given more weight when one is considering downstream applications. Given the dual output of our model, evaluation using segmentation is another option, except that creating full gold standard segmentations would be a particularly difficult annotation task, since our notion of multiword unit is a broad one.

In light of this, we evaluate by taking the best from these various approaches. Given an existing multiword lexicon, we can evaluate not by comparing our lexicon to it directly, but rather by looking at the extent to which our segmentation preserves these known multiword units. There are several major advantages to this approach: first, it does not require a full lexicon or gold standard segmentation; second, common units are automatically given more weight in the evaluation; third, we can use it for evaluation in very large corpora. Our two main metrics are *subsumption* (Sub), namely the percentage of multiword tokens that are fully contained with a segment, and *exact matches* (Exact), the percentage of multiword tokens which correspond exactly to a segment. Exact matches would seem to be preferable to subsumption, but in practice this is not necessarily the case, since our method often identifies valid compound terms and larger constructions than our reference lexicon contains; for example, WordNet only contains the expression *a lot*, but when appearing as part of a noun phrase our model typically segments this to *a lot of*, which, in our opinion, is a preferable segmentation. To quantify overall performance, we calculate a harmonic mean (Mean) of the two metrics. We also looked specifically at performance for terms of 3 or more words (Mean 3+), which are less studied and more relevant to our interests.

Our second evaluation focuses on the quality of these longer terms with a post hoc annotation of output from our model and the best alternatives. We randomly extracted pairs of segments of three words or more where our model mostly but not entirely overlapped with an alternative model (750 examples per corpus per method), and asked CrowdFlower workers to choose which output seemed to be a better multiword unit in the context; they were shown the entire sentence with the relevant span underlined, and then the two individual chunks separately. To ensure quality, we used our multiword lexicon to

[2]This probability is based on the respective counts in our preliminary lexicon at this step in the process, not the original n-gram probability. One key advantage to doing the initial segmentation first is that words that appear consistently in larger units, an extreme example is the bigram *vector machine* in the term *support vector machine*, already have low or zero probability, and will not appear in the lexicon or be good candidate segments for decomposition. This rather intuitively accomplishes what the c-value metric is modeling by applying negative weights to candidates appearing in larger n-grams.

create gold standard examples (comparing known multiword units to purposely bad segmentations which overlapped with them), and used them to test and filter out unreliable workers: for inclusion in our final set, we required a minimum 90% performance on the test questions. We also limited each contributor to only 250 judgments, so that our results reflected a variety of opinions.

We considered a number of alternatives to our approach, though we limited the comparison to methods which could predict segments greater than 2 words, those that were computationally feasible for large corpora, and those which segment into single words only as a last resort: approaches which prefer single words cannot do well under our evaluation because we have no negative examples, only positive ones. The majority of our alternatives involve ranking all potential n-grams (not just the maximal) with $n \geq 2$ and then greedily segmenting them: big-n prefers longer n-grams (with a backoff to counts); c-value is used for term extraction (Frantzi et al., 2000) and was also compared to by Newman et al. (2012); ME refers to the Mutual Expectation metric (Dias et al., 1999); and PMI uses a standard extension of PMI to more than 2 words. We also tested standard (pairwise) PMI as a metric for recursively joining contiguous units (starting with words) into larger units until no larger units can be formed (PMI join), and a version of our decomposition algorithm which selects the minimal breaks which maximize total word count across segments rather than total word predictability (count decomp); the fact that traditional association metrics are not defined for single words prevents us from using them as alternatives to predictability in our decomposition approach. Finally, we also include an oracle which chooses the correct n-grams when they are available for segmentation, but which still fails for units that are below our threshold.

We evaluated our model in four large English corpora: news articles from the Gigaword corpus (Graff and Cieri, 2003) (4.9 billion tokens), out-of-copyright texts from the Gutenberg Project[3] (1.7 billion tokens), a collection of abstracts from PubMed (2.2 billion tokens)[4], and blogs from the ICWSM 2009 social media corpus (Burton et al., 2009) (1.1 million tokens). Our main comparison lexicon is WordNet 3.0, which contains a good variety of expressions appropriate to the various genres, but we also included multiword terms from the Specialist Lexicon[5] for better coverage of the biomedical domain. One issue with our evaluation is that it assumes all tokens are true instances of the multiword unit in question; we carried out a manual inspection of multiword tokens identified by string match in our development sets (5000 sentences set aside from each of the abstract and blog corpora), and excluded from the evaluation a small set of idiomatic expressions (e.g. *on it*, *do in*) whose literal, non-MWE usage is too common for the expression to be used reliably for evaluation; otherwise, we were satisfied that the vast majority of multiword tokens were true matches. When one multiword token appeared within another, we ignored the smaller of the two; when two overlapped in the text, we ignored both.

5 Results

All the results for the main evaluation are shown in Table 1. First, we observe that our initial segmentation always provides the highest subsumption, and our final lexicon always provides the highest harmonic mean, with a modest drop in subsumption but a huge increase in exact matches. The alternative models fall roughly into two categories: those which have reasonably high subsumption, but few exact matches (PMI rank seems to be the best of these) and those that have many exact matches (sometimes better than either of our models) but are almost completely ineffective for identifying multiword units of length greater than 2 (ME rank and c-value, with ME offering more exact matches): the latter phenomenon is attributable to the predominance of two-word multiword tokens in our evaluation, which means a model can do reasonably well by guessing mostly two-word units. For the corpora with more multiword units of greater length, i.e. the PubMed abstracts and the Gutenberg corpus, our method also provides the most exact matches. Our best results come in the PubMed corpus, probably because the texts are the most uniform, though results are satisfactory in all four corpora tested here, which represent a considerable range of genres.

[3] http://www.gutenberg.org . Here we use the English texts from the 2010 image, with headers and footers filtered out using some simple heuristics.

[4] http://www.ncbi.nlm.nih.gov/pubmed/

[5] http://www.nlm.nih.gov/research/umls/new_users/online_learning/LEX_001.htm

Table 1: Performance in segmenting multiword units of various segmentation methods in 4 large corpora. Sub. = Subsumption (%); Exact = Exact Match (%); Mean = Harmonic mean of Sub and Exact; Mean 3+ = Harmonic mean of Sub and Exact for multiword tokens of at length 3 or more. Bold is best in column for corpus, excluding the oracle.

Method	Gigaword news articles				Gutenberg texts			
	Sub	**Exact**	**Mean**	**Mean 3+**	**Sub**	**Exact**	**Mean**	**Mean 3+**
Oracle	97.1	97.1	97.1	95.5	97.0	97.0	97.0	97.8
big-*n* rank	88.7	28.8	43.5	51.4	84.9	30.1	44.4	57.5
c-value rank	69.1	66.1	67.6	23.3	58.6	57.7	58.2	12.6
ME rank	75.3	**70.0**	72.6	14.4	63.2	61.0	62.1	10.9
PMI rank	90.8	30.0	45.1	53.5	86.9	32.8	47.7	61.2
PMI join	83.1	32.8	47.0	43.7	77.7	32.6	46.0	45.5
Count decomp	75.9	31.3	44.3	47.1	69.2	31.5	43.3	54.2
Prediction decomp, initial	**92.2**	36.4	52.2	**64.4**	**89.3**	38.7	54.0	**71.6**
Prediction decomp, final	85.6	66.4	**75.2**	63.8	78.9	**62.8**	**70.0**	61.6

Method	PubMed abstracts				ICWSM blogs			
	Sub	**Exact**	**Mean**	**Mean 3+**	**Sub**	**Exact**	**Mean**	**Mean 3+**
Oracle	91.9	91.9	91.9	84.0	96.5	96.5	96.5	99.4
big-*n* rank	82.2	40.1	53.9	55.5	86.1	33.3	48.0	60.8
c-value rank	63.2	62.3	62.7	21.7	64.3	62.4	63.3	14.6
ME rank	68.5	65.8	67.1	9.1	69.7	**66.2**	67.9	11.7
PMI rank	87.0	41.4	56.1	58.3	88.4	35.7	50.8	63.4
PMI join	79.8	39.7	53.0	46.8	80.3	35.4	49.1	47.0
Count decomp	71.0	38.4	49.9	50.4	71.5	33.5	45.6	53.9
Prediction decomp, initial	**88.6**	50.3	64.1	67.2	**90.5**	40.3	55.8	**70.9**
Prediction decomp, final	85.2	**73.4**	**78.8**	**69.5**	83.2	64.9	**72.9**	66.9

Table 2: CrowdFlower pairwise preference evaluation, our full model versus a selection of alternatives

Comparison	**Preference for Prediction decomp, final**
Prediction decomp, final vs. ME	57.9%
Prediction decomp, final vs. Multi PMI	71.0%
Prediction decomp, final vs. Prediction decomp, initial	70.5%

For our crowdsourced evaluation, we compared our final model to the best models of each of the two major types from the first round, namely Mutual Expectation and PMI rank, as well as our initial segmentation. The results are given in Table 2. Our full model is consistently preferred over the alternatives. This is not surprising in the case of the high-subsumption, low-accuracy models, since the resulting segments often have extraneous words included: an example is *in spite of my*, which our model correctly segmented to just *in spite of*. Given that the ME ranking rarely produces units larger than 2 words, however, we might have predicted that when it does it would be more precise than our model, but in fact our model was somewhat preferred (a chi-square test confirmed that this result was statistically different from chance, $p < 0.001$). An example of an instance where our model offered a better segmentation is *call for an end to* as compared to *for an end to* from the ME model, though there are also many instances where the ME segmentation is more sensible, e.g. *what difference does it make* as compared to *difference does it make* from our model.

Looking closer at the output and vocabulary of our model across the various genres, we see a wide range of multiword phenomena: in the medical abstracts, for instance, there is a lot of medical jargon (e.g. *daily caloric intake*) but also other larger connective phrases and formulaic language (e.g. *an alternative explanation for*, *readily distinguished from*). The blogs also have (very different) formulaic language of

the sort studied using lexical bundles (e.g. *all I can say is that*, *where else can you*) and lots of idiomatic language (e.g. *reinventing the wheel*, *look on the bright side*). The idioms from the Gutenberg, not surprisingly, tend to be less clichéd and more evocative (e.g. *ghost of a smile*); there are rather stodgy expressions like *far be it from me* and conjunctions we would not see in the other corpora (e.g. *rocks and shoals*, *masters and mistresses*). By contrast, many of the larger expressions in the news articles are from sports and finance (e.g. *investor demand for*, *tied the game with*), with many that would be filtered out using the simple grammatical filters often applied in this space. However, for bigrams in particular, some additional syntactic filtering is clearly warranted.

6 Conclusion

We have presented an efficient but effective method for segmenting a corpus into multiword collocational units, with a particular focus on units of length greater than two. Our evaluation indicates that this method results in high-quality segments that capture a variety of multiword phenomena, and is better in this regard than alternatives based on relevant association measures. This result is consistent across corpora, though we do particularly well with highly stereotyped language such as seen in the biomedical domain.

Future work on improving the model will likely focus on extensions related to syntax, for instance bootstrapped POS filtering and discounting of predictability that can be attributed solely to syntactic patterns. Our method could also be adapted to decompose full syntactic trees rather than sequences of words, offering tractable alternatives to Bayesian approaches that identify recurring tree fragments (Cohn et al., 2009); this would allow us, for instance, to correctly identify constructions with long-distance dependencies or other kinds of variation where relying on the surface form is insufficient (Seretan, 2011).

With regards to applications, we will be investigating how to help learners notice these chunks when reading and then use them appropriately in their own writing; this work will eventually intersect with the well-established areas of grammatical error correction (Leacock et al., 2014) and automated essay scoring (Shermis and Burstein, 2003). As part of this, we will be building distributional lexical representations of these multiword units, which is why our emphasis here was on a highly scalable method. Part of our interest is of course in capturing the semantics of idiomatic phrases, but we note that even in the case when a multiword unit is semantically compositional, it might provide *de facto* word sense disambiguation or be stylistically distinct from its components, i.e. be very specific to a particular genre or sub-genre. Therefore, provided we have enough examples to get reliable distributional statistics, these larger units are likely to provide useful information for various downstream applications.

Acknowledgments

This work was supported by the Natural Sciences and Engineering Research Council of Canada and the MITACS Elevate program. Thanks to our reviewers and also Tong Wang and David Jacob for their input.

References

Steven Abney. 1991. Parsing by chunks. In Robert Berwick, Steven Abney, and Carol Tenny, editors, *Principle-Based Parsing*, pages 257–278. Kluwer Academic Publishers.

Vitor De Araujo, Carlos Ramisch, and Aline Villavicencio. 2011. Fast and flexible MWE candidate generation with the mwetoolkit. In *Proceedings of the Multiword Expression Workshop at ACL 2011 (MWE 2011)*.

Timothy Baldwin and Su Nam Kim. 2010. Multiword expressions. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group, Boca Raton, FL.

Douglas Biber, Susan Conrad, and Viviana Cortes. 2004. *If you look at...*: Lexical bundles in university teaching and textbooks. *Applied Linguistics*, 25:371–405.

Kevin Burton, Akshay Java, and Ian Soboroff. 2009. The ICWSM 2009 Spinn3r Dataset. In *Proceedings of the Third Annual Conference on Weblogs and Social Media (ICWSM '09)*, San Jose, CA.

Yu-Hua Chen and Paul Baker. 2010. Lexical bundles in L1 and L2 academic writing. *Language Learning & Technology*, 14(2):30–49.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL '09)*.

Gaël Dias, Sylvie Guilloré, and José Gabriel Pereira Lopes. 1999. Language independent automatic acquisition of rigid multiword units from unrestricted text corpora. In *Proceedings of Conférence Traitement Automatique des Langues Naturelles (TALN) 1999*.

Thomas Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.

Stefan Evert. 2004. *The statistics of word cooccurences–word pairs and collocatoins*. Ph.D. thesis, University of Stuttgart.

Afsaneh Fazly, Paul Cook, and Suzanne Stevenson. 2009. Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics*, 35(1):61–103.

Katerina Frantzi, Sophia Ananiadou, and Hideki Mima. 2000. Automatic recognition of multi-word terms: the c-value/nc-value method. *International Journal on Digital Libraries*, 3:115–130.

David Graff and Christopher Cieri. 2003. *English Gigaword*. Linguistic Data Consortium, Philadelphia, PA.

Ulrich Heid. 2007. Compuational linguistic aspects of phraseology. In Harald Burger, Dmitrij Dobrovol'skij, Peter Kühn, and Neal R. Norrick, editors, *Phraseology. An international handbook*. Mouton de Gruyter, Berlin.

Adam Kilgarriff and David Tugwell. 2001. Word sketch: Extraction and display of significant collocations for lexicography. In *Proceedings of the ACL Workshop on Collocation: Computational Extraction, Analysis and Exploitation*.

Nidhi Kulkarni and Mark Finlayson. 2011. jMWE: A Java toolkit for detecting multi-word expressions. In *Proceedings of the Multiword Expression Workshop at ACL 2011 (MWE 2011)*.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2014. *Automated Grammatical Error Detection for Language Learners (2nd Edition)*. Morgan & Claypool.

David Newman, Nagendra Koilada, Jey Han Lau, and Timothy Baldwin. 2012. Bayesian text segmentation for index term identification and keyphrase extraction. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING '12)*.

Pavel Pecina. 2010. Lexical association measures and collocation extraction. *Language Resources and Evaluation*, 44:137–158.

Ted Pedersen, Satanjeev Banerjee, Bridget McInnes, Saiyam Kohli, Mahesh Joshi, and Ying Liu. 2011. The Ngram statistics package (text::nsp) : A flexible tool for identifying ngrams, collocations, and word associations. In *Proceedings of the Multiword Expression Workshop at ACL 2011 (MWE 2011)*.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLing '02)*.

Patrick Schone and Dan Jurafsky. 2001. Is knowledge-free induction of multiword unit dictionary headwords a solved problem? In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP '01)*.

Violeta Seretan. 2011. *Syntax-Based Collocation Extraction*. Springer.

Mark D. Shermis and Jill Burstein, editors. 2003. *Automated Essay Scoring: A Cross-Disciplinary Approach*. Lawrence Erlbaum Associates, Mahwah, NJ.

Frank Smadja. 1993. Retrieving collocations from text: Xtract. *Computational Linguistics*, pages 143–177.

docrep: A lightweight and efficient document representation framework

Tim Dawborn and **James R. Curran**
ə-lab, School of Information Technologies
University of Sydney
NSW 2006, Australia
{tim.dawborn,james.r.curran}@sydney.edu.au

Abstract

Modelling linguistic phenomena requires highly structured and complex data representations. Document representation frameworks (DRFs) provide an interface to store and retrieve multiple annotation layers over a document. Researchers face a difficult choice: using a heavy-weight DRF or implement a custom DRF. The cost is substantial, either learning a new complex system, or continually adding features to a home-grown system that risks overrunning its original scope.

We introduce DOCREP, a lightweight and efficient DRF, and compare it against existing DRFs. We discuss our design goals and implementations in C++, Python, and Java. We transform the OntoNotes 5 corpus using DOCREP and UIMA, providing a quantitative comparison, as well as discussing modelling trade-offs. We conclude with qualitative feedback from researchers who have used DOCREP for their own projects. Ultimately, we hope DOCREP is useful for the busy researcher who wants the benefits of a DRF, but has better things to do than to write one.

1 Introduction

Computational Linguistics (CL) is increasingly a data-driven research discipline with researchers using diverse collections of large-scale corpora (Parker et al., 2011). Representing linguistic phenomena can require modelling intricate data structures, both flat and hierarchical, layered over the original text; e.g. tokens, sentences, parts-of-speech, named entities, coreference relations, and trees. The scale and complexity of the data demands efficient representations. A document representation framework (DRF) should support the creation, storage, and retrieval of different annotation layers over collections of heterogeneous documents. DRFs typically store their annotations as stand-off annotations, treating the source document as immutable and annotations "stand-off" with offsets back into the document.

Researchers may choose to use a heavy-weight DRF, for example GATE (Cunningham et al., 2002) or UIMA (Götz and Suhre, 2004), but this can require substantial investment to learn and apply the framework. Alternatively, researchers may "roll-their-own" framework for a particular project. While this is not inherently bad, our experience is that the scope of such smaller DRFs often creeps, without the benefits of the features and stability present in mature DRFs. Moreover, some DRFs are based on object serialisation, restricting the user to a specific language. In sum, while DRFs provide substantial benefits, they can come at an opportunity cost to valuable research time.

DOCREP aims to solve this problem by proving a light-weight DRF that does not get in the way. Using a language-agnostic storage layer enables reuse across different tasks in whatever tools and programming languages are most appropriate. Efficiency is our primary goal, and we emphasise compact serialisation and lazy loading. Our streaming design is informed by the pipeline operation of UNIX commands.

Section 2 compares existing DRFs and annotation schemes. We describe and introduce DOCREP in Section 3, outlining the design goals and the problems it aims to solve. We compare DOCREP to UIMA through a case study in Section 4, converting OntoNotes to both DRFs. Section 5 discusses real world uses of DOCREP within our research group and outlines experiences of its use by NLP researchers. DOCREP

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 762–771, Dublin, Ireland, August 23-29 2014.

will be useful for any researcher who wants rapid development with multi-layered annotation that performs well at scale, but at minimal technical cost.

2 Background

Easily and efficiently storing and retrieving linguistic annotations over corpora is a core issue for data-driven linguistics. A number of attempts to formalise linguistic annotation formats have emerged over the years, including Annotation Graphs (AG) (Bird and Liberman, 1999), the Linguistic Annotation Format (LAF) (Ide and Romary, 2004, 2006), and more recently, the Graph Annotation Framework (GRAF) (Ide and Suderman, 2007). GRAF is a serialisation of the LAF model, using XML stand-off annotations to store layers of annotation. The GRAF representation is sufficiently abstract as to be used as a pivot format between other annotation schemes. Ide and Suderman (2009) use GRAF as an intermediate format to convert annotations between GATE and UIMA. The MASC corpus (Ide et al., 2010) has multiple layers of annotation which are distributed in GRAF. Neumann et al. (2013) provide insight into the effectiveness of GRAF as a format for corpus distribution when they import MASC into an annotation database. These linguistic annotation formalisations provide a useful set of requirements for DRFs. While these abstract formalisations are constructive from a theoretical perspective, they do not take into account the runtime performance of abstract representations, nor their ease of use for programmers.

Several DRFs have been developed and used within the CL community. GATE (Cunningham et al., 2002; Cunningham, 2002) has a focus on the human annotation of textual documents. While it has a large collection of extensions and plugins, it was not designed in a matter than suits web-scale corpus processing. Additionally, GATE is limited to Java, making integration with CL tools written in other languages difficult. UIMA (Götz and Suhre, 2004; Lally et al., 2008) is a Java framework for providing annotations over the abstract definition of documents, providing functionality to link between different views of the same document (e.g. translations of a document). UIMA calls these different views different "subjects of analysis" (SOFA). When UIMA was adopted into the Apache Software Foundation, a C++ version of the UIMA API was developed. However, it appears to lag behind behind the Java API in development effort and usefulness, with many undocumented components, numerous external dependencies, and with substantial missing functionality provided by the Java API. Additionally, the C++ API is written in an non-idiomatic manner, making it harder for developers to use.

Publicly available CL pipelining tools have emerged in recent years, providing a way to perform a wide range of CL processes over documents. The Stanford NLP pipeline[1] is one such example, but is Java only and must be run on a single machine. CURATOR (Clarke et al., 2012) provides a cross-language NLP pipeline using Thrift to provide cross-language communication and RPC. CURATOR requires a server to coordinate the components within the pipeline. Using pipelining functionality within a framework often the inspection of per-component contributions more difficult. We are not aware of any DRFs which use a streaming model to utilise UNIX pipelines, a paradigm CL researchers are already familiar with.

3 The docrep document representation framework

DOCREP (/dɒkrɛp/), a portmanteau of *doc*ument *rep*resentation, is a lightweight, efficient, and modern document representation framework for NLP systems that is designed to be simple to use and intuitive to work with. We use the term lightweight to compare it to the existing document representation systems used within the CL community, the main one being UIMA. The overhead of using DOCREP instead of a flat-file format is minimal, especially in comparison to large bulky frameworks.

Our research group has used DOCREP as its primary data storage format in both research projects and commercial projects since mid-2012. DOCREP has undergone an iterative design process during this time as limitations and issues arose, allowing modelling issues to be ironed out and a set of best practices to be established. These two years of solid use by CL researchers has resulted in a easy to use DRF we believe is suitable for most CL applications and researchers.

DOCREP was designed with streaming in mind, facilitating from the data storage layer upwards the ability for CL applications to utilise parallel processing. This streaming model is a model that many

[1] `http://nlp.stanford.edu/software/corenlp.shtml`

CL researchers are already familiar with from writing UNIX pipelines (Church, 1994; Brew and Moens, 2002), again reducing the overhead required to use DOCREP.

DOCREP is not a new language that researchers need to learn. Instead, it is a serialisation protocol and set of APIs to interact with annotations and documents. Using DOCREP is as simple as importing the package in ones favourite programming language and annotating class definitions appropriately. Neither a separate compilation step nor an external annotation definition file are required.

3.1 Idiomatic APIs

One of the motivations for constructing DOCREP was the lack of a good document representation framework in programming languages other than Java. We have implemented DOCREP APIs in three commonly used programming languages in the CL community: C++, Python, and Java. All of these APIs are open source and publicly available on GitHub,[2] released under the MIT licence. The C++ API is written in C++11, the Python API supports version 2.7 as well as versions $\geq$ 3.3, and the Java API supports versions $\geq$ 6. All three APIs are setup to use the standard build tools for the language.

When implementing these APIs, we aimed to make the interface as similar as possible between the three languages, while still feeling idiomatic within that language. Using the API should feel natural for that language. Figure 1 shows an example set of identical model definitions in C++, Python, and Java. This example defines a `Token` type, a `Sent` type spanning over a series of sequential `Token` annotations, and a `Doc` type. The `Token` and `Sent` types include some annotation attributes. Annotation instances are stored on the document in `Stores`. Apart from the missing implementations of the `Schema` constructors in the C++ example, these are complete and runnable definitions of annotation types in DOCREP. The `Schema` classes in the C++ example are automatically induced via runtime class introspection in the Python and Java APIs; functionality which C++ does not possess.

3.2 Serialisation protocol

We chose to reuse an existing serialisation format for DOCREP. This allows developers to use existing serialisation libraries for processing DOCREP streams in languages we do not provide a DOCREP API for.

One of our design considerations when creating DOCREP was a desire for the protocol to be self-describing. With a self-describing protocol, no external files need to be associated with a serialised stream in order to know how to interpret the serialised data. This requires an efficient serialisation protocol because including the definition of the type system with each document comes at a cost. This is different to UIMA which requires its XML type definition files in order to deserialise the serialised data.

The four main competitors in the web-scale binary serialisation format space are BSON,[3] MessagePack,[4] Protocol Buffers,[5] and Thrift.[6] BSON and MessagePack are similar in their design. They both aim to provide a general purpose data serialisation format for common data types and data structures. BSON is used as the primary data representation within the MongoDB database. Protocol Buffers and Thrift work in a similar manner to one another. Their serialisation protocols are not self describing and require an external file which defines how to interpret the messages on the stream. In this external file, users define the structure of the messages they wish to serialise and deserialise, and use a provided tool to convert this external file into source code for their programming language of choice. Protocol Buffers and Thrift also provide RPC functionality, however this was not needed for our situation. Thrift is used by the CURATOR NLP pipeline (Clarke et al., 2012) to provide both serialisation and RPC functionality between cross-language disjoint components in the pipeline.

After designing the serialisation protocol for DOCREP, we implemented it on top of these binary serialisation formats in order to compare the size of the serialised data and the speed at which it could be compressed. As a simple stand-off annotation task, we chose to use the CoNLL 2003 NER shared task

[2] `https://github.com/schwa-lab/libschwa`
[3] `http://bsonspec.org/`
[4] `http://msgpack.org/`
[5] `http://code.google.com/p/protobuf/`
[6] `http://thrift.apache.org/`

```
struct Token : public dr::Ann {
  dr::Slice<uint64_t> span;
  std::string raw;
  std::string norm;
  class Schema;
};

struct Sent : public dr::Ann {
  dr::Slice<Token *> span;
  bool is_headline;
  class Schema;
};

struct Doc : public dr::Doc {
  dr::Store<Token> tokens;
  dr::Store<Sent> sents;
  class Schema;
};
```

```
struct Token::Schema : public dr::Ann::Schema<Token> {
  DR_FIELD(&Token::span) span;
  DR_FIELD(&Token::raw) raw;
  DR_FIELD(&Token::norm) norm;
  Schema(void);
};

struct Sent::Schema : public dr::Ann::Schema<Sent> {
  DR_POINTER(&Sent::span, &Doc::tokens) tokens;
  DR_FIELD(&Sent::is_headline) is_headline;
  Schema(void);
};

struct Doc::Schema : public dr::Doc::Schema<Doc> {
  DR_STORE(&Doc::tokens) tokens;
  DR_STORE(&Doc::sents) sents;
  Schema(void);
};
```

(a) C++ example

```
class Token(dr.Ann):
  span = dr.Slice()
  raw = dr.Text()
  norm = dr.Text()

class Sent(dr.Ann):
  span = dr.Slice(Token)
  is_headline = dr.Field()

class Doc(dr.Doc):
  tokens = dr.Store(Token)
  sents = dr.Store(Sent)
```

(b) Python example

```
@dr.Ann
public class Token extends AbstractAnn {
  @dr.Field public ByteSlice span;
  @dr.Field public String raw;
  @dr.Field public String norm;
}

@dr.Ann
public class Sent extends AbstractAnn {
  @dr.Pointer public Slice<Token> span;
  @dr.Field public bool isHeadline;
}

@dr.Doc
public class Doc extends AbstractDoc {
  @dr.Store public Store<Token> tokens;
  @dr.Store public Store<Sent> sents;
}
```

(c) Java example

Figure 1: Examples of identical type definitions using the DOCREP API in C++, Python, and Java.

	Self-describing	Uncompressed		DEFLATE		Snappy		LZMA	
		Time	Size	Time	Size	Time	Size	Time	Size
Original data	–	–	31.30	1.0	5.95	0.1	9.81	39	0.39
BSON	✓	2.5	188.42	5.3	30.32	0.6	56.36	441	16.22
MessagePack	✓	1.6	52.15	3.2	16.61	0.3	24.82	61	4.36
Protocol Buffers	×	1.4	51.51	3.5	18.52	0.3	29.31	67	5.13
Thrift	×	1.0	126.12	3.5	20.64	0.4	33.69	224	10.99

Table 1: A comparison of binary serialisation libraries being used as the DOCREP serialisation format. Times are reported in seconds and sizes in MB. MessagePack and BSON include the full type system definition on the stream for each document whereas Protocol Buffers and Thrift do not.

data, randomly sampling around 50 MB worth of sentences from the English training data. The serialisation stores the documents, sentences, and tokens, along with the POS and NER tags for the tokens. The appropriate message specification files were written for Protocol Buffers and Thrift, and the type system was serialised as a header for BSON and MessagePack.

Table 1 shows the results of this experiment. The reported size of the original data is smaller than the sample size as we chose to output it in a more concise textual representation than the data was originally distributed in. BSON performs noticeably worse than the others, in terms of both size and speed. While serialising slightly faster, the size of the serialised data produced by Thrift is more then double the size of both MessagePack and Protocol Buffers, and does not compress quite as well. MessagePack compressed slightly better than Protocol Buffers and was on par in terms of speed, while being self-describing on the stream. The result of this experiment and some similar others lead us to conclude that MessagePack was the best serialisation format for DOCREP to use.

At the time of writing, the Python and Java DOCREP APIs use the official MessagePack libraries for those languages. We implemented our own C++ MessagePack library to facilitate laziness.

3.3 Laziness

The serialisation protocol was designed such that we could make the streaming aspect of DOCREP as efficient as possible. Before each collection of annotation objects appears in the serialised data, the number of bytes used to store the serialised annotations is stored. If the current application is not interested in the particular annotation types that are about to be read in, it can simply skip over the correct number of bytes without having to deserialise the internal MessagePack structure.

All three of our APIs implement this laziness. Only the types of annotations that the application specifies interest in will be deserialised at runtime. The other types of annotations will simply be kept in their serialised format and written back out to the output stream unmodified. This is also true for attributes on annotations that the current application is not interested in. The Python API provides an option to fully instantiate each of the types at runtime, even if you have not defined classes for them. Unknown annotation types will have classes created at runtime based on the schema of the types described in the serialisation protocol.

3.4 Processing tools

We trade-off performance against easy inspection of files. We provide a set of command-line tools for manipulating, filtering, and distributing DOCREP streams. The command-line tools mimic the standard set of UNIX tools used to process textual files as well as some other stream introspection and statistics gathering tools. All of these tools and their uses are documented on the DOCREP website.[7] Our provided toolbox for processing DOCREP streams contains tools for counting, visualising, filtering, ordering, partitioning, and exporting DOCREP streams. Due to space limitations in this paper, we are unable to go into these tools in detail.

Below are two examples of some of the tools in action. The first example filters the documents by a regular expression comparison against their ID attribute, and then outputs the ID of the document with the most number of tokens. The second randomly chooses 10 documents from a stream, passing them to another tool, and then opens the first returned document in the stream visualiser.

```
$ dr grep 'doc.id ~ /x-\d+/' corpus.dr | dr count -s tokens | sort -rn | head -n 1
$ dr sample -n 10 corpus.dr | ./my-tool | dr head -n 1 | dr less
```

3.5 Streaming model

Emphasising the fact that the DOCREP protocol is a streaming protocol, combining multiple DOCREP files together is as simple as concatenating the files together. The DOCREP deserialisers expect an input stream to contain zero or more serialised documents. Being able to easily distribute all documents in a corpus along with their annotation layers as a single file is very attractive.

[7] https://github.com/schwa-lab/libschwa

This kind of streaming model makes distributed processing very easy using a typical work queue model. A distributed pipeline "source" can serve the documents from the DOCREP stream by reading them off the input stream without having to deserialise them (subsection 3.3) and a "sink" can simply concatenate the received documents together to the output stream, again without having to deserialise them. We provide a DOCREP source and sink distributed processing tool along with APIs for easily writing worker clients. The distribution is achieved through ØMQ[8] which allows for both scale-up and scale-out distributed processing out of the box without the need for a separate controller process to manage communication between client processes.

4 Case study: OntoNotes 5

The OntoNotes 5 corpus (Pradhan et al., 2013) is a large corpus of linguistically annotated documents from multiple genres in three different languages. This 5th release covers newswire, broadcast news, broadcast conversation, and web data in English and Chinese, a pivot corpus in English, and newswire data in Arabic. Roughly half of the broadcast conversation data is parallel data, with some of the documents providing tree-to-tree alignments. Of the 15 710 documents in the corpus, 13 109 are in English, 2002 are in Chinese, and 599 are in Arabic.

Each of the documents in the OntoNotes 5 corpus contain multiple layers of syntactic and semantic annotations. It builds upon the Penn Treebank for syntax and PropBank for predicate-argument structure, adding named entities, coreference, and word sense disambiguation layers to some documents.

The annotations in the OntoNotes 5 corpus are provided in two different formats: as a series of flat files (340 MB) per document with each file containing one annotation layer, and as a relational database in the form of a SQL file (5812 MB). Both of these data formats have usability issues. Working with the flat files requires parsing each of the different file formats and aligning the data between the files for the same document. Working with the database requires working out how the tables are related to one another, as well as knowledge of SQL, or having access to an efficient API for querying the database.

To outline the effectiveness of document representation frameworks, and in particular the efficiency of DOCREP, we provide code to convert the OntoNotes 5 corpus into both DOCREP and UIMA representations, comparing the conversion time, resultant size on disk, and ease of doing this conversion. We provide conversion scripts in all three languages for DOCREP and in Java and C++ for UIMA. Additionally, we also provide a verification script, reproducing the original OntoNotes 5 flat files from the document representation form, ensuring that no data was lost in the conversion.

4.1 Modelling decisions

The choices made on how to model the different annotation layers were almost identical in UIMA and DOCREP. The main difference occurs when you have an annotation over a sequential span of other annotations. UIMA has no way to model this directly. The most common way users choose to model this is as a normal `Annotation` subtype with its `begin` offset set to the `begin` offset of the first covered annotation and its `end` offset set to the `end` offset of the last covered annotation. An example of this situation is named entity annotations. In OntoNotes, named entities are represented as annotations over a sequence of token annotations. How this is represented in UIMA is shown in the XML snippet in Figure 2. The main disadvantage in this modelling approach is that there is then no direct representation that the named entity annotation is an annotation over a sequence of token annotations. In DOCREP, named entity annotation is directly modelled as a sequence of token annotations. The DOCREP definition for the named entity type is shown on the right hand side of Figure 2.

DOCREP does not allow for the direct modelling of cross-document information. This occurs in the OntoNotes 5 corpus in the form of the parallel document and parallel tree information. Because DOCREP is a streaming protocol, the documents are thought of as independent from one another and as such, no formal relationships between the documents can be made at the framework level. This parallel document information can still be be stored as metadata on the documents. This situation is dealt with in UIMA by the SOFA.

[8] `http://www.zeromq.org/`

```
<typeDescription>
  <name>
ontonotes5.to_uima.types.NamedEntity
  </name>
  <description/>
  <supertypeName>
    uima.tcas.Annotation
  </supertypeName>
  <features>
    <featureDescription>
      <name>tag</name>
      <description>The NE tag.</description>
      <rangeTypeName>uima.cas.String</rangeTypeName>
    </featureDescription>
    <featureDescription>
      <name>startOffset</name>
      <description>Character offset into the start token.</description>
      <rangeTypeName>uima.cas.Integer</rangeTypeName>
    </featureDescription>
    <featureDescription>
      <name>endOffset</name>
      <description>Character offset into the end token.</description>
      <rangeTypeName>uima.cas.Integer</rangeTypeName>
    </featureDescription>
  </features>
  </typeDescription>
```

```
@dr.Ann
public class NamedEntity extends AbstractAnn {
  @dr.Pointer public Slice<Token> span;
  @dr.Field public String tag;
  @dr.Field public int startOffset;
  @dr.Field public int endOffset;
}
```

Figure 2: Defining the named entity annotation type in UIMA (left) and the DOCREP Java API (top-right).

	UIMA							DOCREP		
	Java XMI	Java XCAS	Java bin	Java cbin	C++ XMI	C++ XCAS	C++ bin	Java –	C++ –	Python –
Conversion time	25	25	25	25	77	77	77	12	12	27
Serialisation time	131	122	2103	76	630	611	695	61	23	32
Size on disk	1894	3252	1257	99	2141	3252	2135	371	371	371

Table 2: A comparison of the resources required to represent the OntoNotes 5 corpus in UIMA and DOCREP. Times are reported in seconds and sizes are reported in MB.

4.2 Empirical results

In these experiments, we first load all of the data into memory from the database for the current document we are processing. This data is stored in an object structure which knows nothing about document representation frameworks. We then convert this object representation into the appropriate UIMA and DOCREP annotations, recording how long the conversion took. The UIMA and DOCREP versions of the documents are then serialised to disk, recording how long the serialisation took and the resultant size on disk. All of these performance experiments were run on the same isolated machine, running 64-bit Ubuntu 12.04, using OpenJDK 1.7, CPython 2.7, and gcc 4.8.

In order to provide a fair comparison between UIMA and DOCREP, we perform the conversion using both the Java and C++ UIMA APIs, as well as using all three DOCREP APIs (Java, C++, and Python). The code to load the data from the database and construct the in-memory object structure was common between the UIMA and DOCREP conversions. For UIMA, we serialise in all available output formats: both the XMI and XCAS XML formats, the binary format (bin), and the compressed binary (cbin) format. The UIMA C++ API does not appear to support output in the compressed binary format.

The result of this conversion process can be seen in Table 2. The first row shows the accumulated time taken to convert all of the documents from their in-memory representation into UIMA and DOCREP annotations. As visible in the table, DOCREP performs this conversion twice as fast as UIMA in Java and six times as fast as UIMA in C++ The second row shows the accumulated time taken to serialise

	Flat files	DOCREP	UIMA XMI	UIMA XCAS	UIMA bin	UIMA cbin	SQL	MySQL -indices	MySQL +indices
Uncompressed	340	371	1894	3252	1257	99	4560	4303	5812
`gzip` (DEFLATE)	52	115	268	330	375	66	646	–	–
`xz` (LZMA)	30	69	144	185	150	65	262	–	–

Table 3: A comparison of the how well each of the annotation serialisation formats compress using standard compression libraries. All sizes are reported in MB.

all of the documents to disk. DOCREP serialises up to 34 times faster than UIMA in Java, depending on the UIMA output format, and up to 30 times faster in C++ Ṫhe third row in this table shows the accumulated serialisation size on disk. Apart from the compressed binary output format in UIMA (cbin), DOCREP serialisation requires up to nine times less space than UIMA Ẇe are unsure why the sizes for the different output formats in UIMA do not match up between the Java and C++ APIs Ẇe are also unsure why the UIMA Java binary serialisation is so slow, especially in comparison to the compressed binary serialisation.

Table 3 shows how well each of the serialisation formats compress using three standard compression libraries. Each of these compression libraries were run with their default settings. The files generated by UIMA as well as the "flat file" files were first placed into a tarball so that the compression algorithms could be run over the whole corpus instead of per document. The "flat files" used were the original OntoNotes 5 flat files containing the annotation layers that were converted. The SQL numbers are using the original OntoNotes 5 SQL file. The MySQL numbers are obtained after loading the original SQL into a MySQL database and obtaining table and index sizes from the `information_schema.tables` table. The MySQL database was not altered from the initial import. Unsurprisingly, the DOCREP binary representation does not compress as well as textual serialisation formats with lots of repetition, such as XML or the original stand-off annotation files. However, under all of these reported situations, apart from the UIMA compressed binary format, our DOCREP representation is two to five times smaller than its UIMA counterpart, and 15 times smaller than the representation in MySQL. The UIMA compressed binary (cbinary) format has already been compressed so it is unsurprising that compressing it further makes little difference.

5 Usability

We have primarily evaluated the usefulness of DOCREP from an efficiency perspective, reporting time and space requirements for a complex corpus conversion. In this section, we provide feedback from NLP researchers in our lab who have been using DOCREP over the past two years for a variety of NLP tasks. As researchers ourselves, we are aware of how valuable research time is. We provide these real-world examples of DOCREP's use to solidifying that DOCREP is a valuable tool for researchers.

Coreference DOCREP *is a great tool for this project as all we want to do is develop a good coreference system; we do not want to have to worry about the storage of data. Having an* API *in Python is super convenient, allowing us to write code that changes frequently as we try new ideas.* Related publication: Webster and Curran (2014)

Event Linking *Some work on Event Linking sought to work with gold annotations on one hand, and knowledge from web-based hyperlinks on the other. For some processes these data sources were to be treated identically, and for some differently.* DOCREP*'s extensibility easily supported this use-case, while providing a consistent polymorphic abstraction that made development straightforward, while incorporating many other layers of annotation such as extracted temporal relations. Separately, describing the relationship between a pair of documents in* DOCREP *was a challenging use-case that required more engineering and fore-thought than most* DOCREP *applications so far.* Related publication: Nothman et al. (2012).

Named Entity Linking *Our approach to* NEL *uses a pipeline of components and we initially wrote our own* DRF *using Python's object serialisation. While this worked well initially, we accrued technical debt as we added features with minimal refactoring. Before too long, a substantial part of our experiment runtime was devoted to dataset loading and storage.* DOCREP *made this easier and using* UNIX *pipelines over structured document objects is a productive workflow.* Related publications: Radford et al. (2012); Pink et al. (2013).

Quote Extraction and Attribution *For this task we performed experiments over four corpora, all with distinct data formats and assumptions. Our early software loaded each format into memory, which was a slow, error-prone, and hard-to-debug process. This approach became completely unusable when we decided to experiment with coreference systems, as it introduced even more unique data formats. Converting everything to* DOCREP *greatly simplified the task, as we could represent everything we needed efficiently, and within one representation system. We also gained a nice speed boost, and were able to write a simple set of tests that examined a given* DOCREP *file for validity, which greatly improved our code quality.* Related publication: O'Keefe et al. (2013).

Slot Filling *Being one of the last stages in an* NLP *pipeline, slot filling utilises all of the document information it can get its hands on. Being able to easily accept annotation layers from prior* NLP *components allows us to focus on slot filling instead of component integration engineering. Having access to a multi-language* API *means we are able to write efficiency-critical code in C++ and the more experimental and dynamic components in Python.*

6 Conclusion

We present a light-weight and easy-to-use document representation framework for the busy NLP researcher who wants to model document structure, but does not want to use a heavy-weight DRF. We provide empirical evidence of the efficiency of DOCREP, and provide insights into its use within our research group over the past two years. We believe NLP other researchers will benefit from DOCREP as they are now able to utilise the usefulness of a DRF without it getting in the way of their research time.

Acknowledgments

We would like to thank the anonymous reviewers for their useful feedback. We would also like to thank Will Radford and Joel Nothman for their contributions to this paper as well as to DOCREP itself over the past years. This work was supported by ARC Discovery grant DP1097291 and the Capital Markets CRC Computable News project.

References

Steven Bird and Mark Liberman. 1999. A formal framework for linguistic annotation. *Speech Communication*, 33:23–60.

Chris Brew and Marc Moens. 2002. Data-intensive linguistics. HCRC Language Technology Group, University of Edinburgh.

Kenneth Ward Church. 1994. Unix™ for poets. *Notes of a course from the European Summer School on Language and Speech Communication, Corpus Based Methods*.

James Clarke, Vivek Srikumar, Mark Sammons, and Dan Roth. 2012. An NLP curator (or: How I learned to stop worrying and love NLP pipelines). In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey.

Hamish Cunningham. 2002. GATE, a general architecture for text engineering. *Computers and the Humanities*, 36:223–254.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: an architecture for development of robust HLT applications. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 168–175. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA.

T. Götz and O. Suhre. 2004. Design and implementation of the UIMA common analysis system. *IBM Systems Journal*, 43(3):476–489.

Nancy Ide, Collin Baker, Christiane Fellbaum, and Rebecca Passonneau. 2010. The manually annotated sub-corpus: A community resource for and by the people. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 68–73. Uppsala, Sweden.

Nancy Ide and Laurent Romary. 2004. International standard for a linguistic annotation framework. *Natural Language Engineering*, 10(3-4):211–225.

Nancy Ide and Laurent Romary. 2006. Representing linguistic corpora and their annotations. In *Proceedings of the Fifth Language Resources and Evaluation Conference LREC*.

Nancy Ide and Keith Suderman. 2007. GrAF: A graph-based format for linguistic annotations. In *Proceedings of the Linguistic Annotation Workshop*, pages 1–8. Association for Computational Linguistics, Prague, Czech Republic.

Nancy Ide and Keith Suderman. 2009. Bridging the Gaps: Interoperability for GrAF, GATE, and UIMA. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 27–34. Association for Computational Linguistics, Suntec, Singapore.

Adam Lally, Karin Verspoor, and Eoric Nyberg. 2008. *Unstructured Information Management Architecture (UIMA) Version 1.0. Standards Specification 5, OASIS*.

Arne Neumann, Nancy Ide, and Manfred Stede. 2013. Importing MASC into the ANNIS linguistic database: A case study of mapping GrAF. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 98–102. Sofia, Bulgaria.

Joel Nothman, Matthew Honnibal, Ben Hachey, and James R. Curran. 2012. Event linking: grounding event reference in a news archive. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 228–232. Jeju, Korea.

Tim O'Keefe, James R. Curran, Peter Ashwell, and Irena Koprinska. 2013. An annotated corpus of quoted opinions in news articles. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 516–520. Association for Computational Linguistics, Sofia, Bulgaria.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English Gigaword Fifth Edition. Technical report, Linguistic Data Consortium, Philadelphia.

Glen Pink, Will Radford, Will Cannings, Andrew Naoum, Joel Nothman, Daniel Tse, and James R. Curran. 2013. SYDNEY_CMCRC at TAC 2013. In *Proceedings of the Text Analysis Conference*. National Institute of Standards and Technology, Gaithersburg, MD USA.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards Robust Linguistic Analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152. Sofia, Bulgaria.

Will Radford, Will Cannings, Andrew Naoum, Joel Nothman, Glen Pink, Daniel Tse, and James R. Curran. 2012. (Almost) Total Recall – SYDNEY_CMCRC at TAC 2012. In *Proceedings of the Text Analysis Conference*. National Institute of Standards and Technology, Gaithersburg, MD USA.

Kellie Webster and James R. Curran. 2014. Low memory incremental coreference resolution. In *Proceedings of COLING 2014*. The COLING 2014 Organizing Committee, Dublin, Ireland. To appear.

Why implementation matters: Evaluation of an open-source constraint grammar parser

Dávid Márk Nemeskey
Institute for Computer Science and Control,
Hungarian Academy of Sciences,
H-1111 Budapest
nemeskey.david@sztaki.mta.hu

Francis M. Tyers
HSL-fakultetet,
UiT Norgga árktalaš universitehta,
9017 Romsa (Norway)
francis.tyers@uit.no

Mans Hulden
Department of Linguistics,
University of Colorado Boulder
80309 Boulder, Colorado
mans.hulden@colorado.edu

Abstract

In recent years, the problem of finite-state constraint grammar (CG) parsing has received renewed attention. Several compilers have been proposed to convert CG rules to finite-state transducers. While these formalisms serve their purpose as proofs of the concept, the performance of the generated transducers lags behind other CG implementations and taggers.

In this paper, we argue that the fault lies with using generic finite-state libraries, and not with the formalisms themselves. We present an open-source implementation that capitalises on the characteristics of CG rule application to improve execution time. On smaller grammars our implementation achieves performance comparable to the current open-source state of the art.

1 Introduction

Constraint grammar (CG), described originally by Karlsson (1990), is a rule-based formalism for various linguistics tasks, including morphological analysis, clause boundary detection and surface syntactic parsing. It has been used in a wide range of application areas, such as morphological disambiguation, grammar checking and machine translation (Bick, 2011). CG owns its popularity to two reasons: first, it achieves high accuracy on free text. Second, it works for languages where the annotated corpora required by statistical parsing methods are not available, but a linguist willing to work on the rules is. The original CG has since been superseded by CG-2 (Tapanainen, 1996) and lately, the free/open-source VISL CG-3 (Bick, 2000; Didriksen, 2011).

Constraint grammar, however, has its drawbacks, one of which is speed. The Apertium machine translation project (Forcada et al., 2011) uses both CG (via VISL CG-3) and n-gram based models for morphological disambiguation, and while CG achieves higher accuracy, the n-gram model runs about ten times faster.

In this paper, we investigate how using finite-state transducers (FST) for CG application can help to bridge the performance gap. In recent years, several methods have been proposed for compiling a CG to FST and applying it on text: Hulden (2011) compiles CG rules to transducers and runs them on the input sentences; Peltonen (2011) converts the sentences into ambiguous automata and attempts to eliminate branches by intersecting them with the rule FSTs; finally, Yli-Jyrä (2011) creates a single FST from the grammar and applies it on featurised input. Unfortunately, none of the authors report exact performance measurements of their systems. Yli-Jyrä published promising numbers for the preprocessing step, but nothing on the overall performance. Peltonen, on the other hand, observed that "VISL CG-3 was 1,500 times faster" than his implementation (Peltonen, 2011).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 772–780, Dublin, Ireland, August 23-29 2014.

We do not attempt here to add a new method to this list; instead, we concentrate on three practical aspects of FST-based CG. First, we report accurate measurements of the real-world performance of one of the methods above. Second, we endeavour to optimise the implementation of the selected method. All three works used *foma*, an open source FST library (Hulden, 2009b; Hulden, 2009a). We show that while *foma* is fast, relying on specialised FST application code instead of a generic library clearly benefits performance. We also demonstrate what further improvements can be achieved by exploiting the peculiarities of CG. Lastly, our research also aims to fill the niche left by the lack of openly accessible finite-state CG implementations.

Section 2 briefly introduces the method we chose to evaluate. In the rest of the paper, we present our optimisations in a way that mirrors the actual development process. We start out with a simple rule engine based on foma, and improve it step-by-step, benchmarking its performance after each modification, instead of a single evaluation chapter. We start in Section 3 by describing our evaluation methodology. Section 4 follows the evolution of the rule engine, as it improves in terms of speed. Section 5 contains a complexity analysis and introduces an idea that theoretically allows us to improve the average- and best-case asymptotic bound. Section 6 demonstrates how memory savings can be derived from the steps taken in section 4. Finally, Section 7 contains our conclusions and lists the problems that remain for future work.

2 The *fomacg* compiler and *fomacg-proc*

We have chosen Hulden's *fomacg* compiler for our study. Our reasons for this are twofold. The transducers generated by *fomacg* were meant to be run on the input directly, but they could also be applied to a finite-state automaton (FSA) representation of the input sentence via FST composition, thereby giving us more space to experiment. Peltonen's method, on the other hand, works only through FST intersection. More importantly, *fomacg* was the only compiler that is openly available.[1]

Here we briefly describe how *fomacg* works; for further details refer to (Hulden, 2011). A CG used for morphological disambiguation takes as input a morphologically analysed text, which consists of *cohorts*: a word with its possible readings. A reading is represented by a lemma and a set of morphosyntactic *tags*. For example, the cohort of the ambiguous Hungarian word *szív* with two readings "heart" and "to suck" would be `^szív/szív<n><sg><nom>/szív<vblex><pres><s3p>$`.[2] The text is tokenised into sentences based on a set of *delimiters*. CG rules operate on a sentence, removing readings from cohorts based on their context. The rules can be divided into priority levels called *section*. Most implementations apply the rules one-by-one in a loop, until no rules can further modify the sentence.

fomacg expects cohorts to be encoded in a different format; the cohort in the example above would be represented as

```
$0$ "<szív>" #BOC#         |
#0# "szív" n sg nom        |
#0# "szív" vblex pres s3p  | #EOC#
```

The rule transducers mark readings for removal by replacing the `#0#` in front of the reading by `#X#`; they act as identity for sentences they cannot be applied to.

fomacg is only a compiler, which reads a CG rule file and emits a *foma* FST for each rule. The actual disambiguator program that applies the transducers to text we implemented ourselves. It reads the morphologically analysed input in the Apertium stream format, converts it into the format expected by *fomacg*, applies the transducers to it, and then converts the result back to the stream format. To emphasise its similarity to *cg-proc*, VISL CG's rule applier, we named our program *fomacg-proc*.

3 Methodology

Apertium includes constraint grammars for several languages.[3] While most of these are wide-coverage grammars, and are being actually used for morphological disambiguation in Apertium, they are also

[1] In the Apertium software repository: `https://svn.code.sf.net/p/apertium/svn/branches/fomacg`

[2] The example is in the Apertium stream format, not in CG-2 style.

[3] `http://wiki.apertium.org/wiki/Constraint_grammar`

too big and complex to be easily used for the early stages of parser development. Therefore, we have written a small Hungarian CG, aimed to fully disambiguate a short Hungarian story, which was used as the development corpus. Since Hungarian is not properly supported by Apertium yet, morphological analysis was carried out by Hunmorph (Trón et al., 2005), and the tags were translated to the Apertium tagset with a transducer in `foma`.

The performance of `fomacg-proc` has been measured against that of VISL CG. The programs were benchmarked with three Apertium CG grammars: the toy Hungarian grammar mentioned earlier, the Breton grammar from the `br-fr` language pair (Tyers, 2010), and the version of the Finnish grammar originally written by Karlsson in the North Sámi–Finnish (`sme-fin`) pair. Seeing that in the early phases, only the Hungarian grammar was used for development, results for the other two languages are reported only for the later steps.

Each grammar was run on a test corpus. For Breton, we used the corpus in the `br-fr` language pair, which consists of 1,161 sentences. There are no Finnish and Hungarian corpora in Apertium; for the former, we used a 1,620-sentence excerpt from the 2013-Nov-14 snapshot of the Finnish Wikipedia, while for the latter, the short test corpus used for grammar development. Since the latter contains a mere 11 sentences, it was repeated 32 times to produce a corpus similar in size to the other two.[4] The Breton and Finnish corpora were tagged by Apertium's morphological analyser tools.

Since VISL CG implements CG-3, and *fomacg* only supports CG-2, a one-to-one comparison with the grammars above was not feasible. Therefore, we extracted the subset of rules from each that compiled under *fomacg*, and carried out the tests on these subsets. Table 1 shows the number of rules in the original and the CG-2 grammars.

Table 1: Grammar sizes with the running time and binary size of the respective VISL-CG grammars

Language	Rules	CG-2 rules	Binary	Time
Hungarian	33	33	8kB	0.284s
Breton	251	226	36kB	0.77s
Finnish	1207	1172	184kB	1.78s

We recorded both initialisation and rule application time for the two programs, via instrumentation in case of *fomacg-proc* and by running the grammar first on an empty file and then on the test corpus in case of *cg-proc*. However, as initialisation is a one-time cost, in the following we are mainly concerned with the time required for applying rules. The tests were conducted on a consumer-grade laptop with a 2.2GHz Core2Duo CPU and 4GB RAM, running Linux.

4 Performance optimisations

Our implementation, much like that of *fomacg* (and indeed, all recent work on finite state CG) is based on the *foma* library. We started out with a naïve implementation that used solely stock *foma* functions. Most of the improvements below stem from the fact that we have replaced these functions with custom versions that run much faster. The final implementation abandons *foma* entirely, but for the data structures. In the future, we plan to discard those as well, making our code self-contained.

The program loads the transducers produced by *fomacg* and applies them to the text. The input is in the Apertium stream format[5] and it is read cohort-by-cohort. A *foma* FST is used to convert each cohort to the format expected by the rule transducers, and to convert the final result back.

To tokenise the text to sentences, we modified *fomacg* to compile the *delimiters* set and emit it as the first FSA in the binary representation of the grammar. *fomacg-proc* reads the input until a cohort matches this set and then sends the accumulated sentence to the rule applier engine.

[4] Although we used the same corpus for development and testing for Hungarian, the experimental setup was the same for VISL-CG and fomacg. While the numbers we acquired for Hungarian are not representative of how a proper Hungarian CG would perform on unseen data, they clearly show which of our steps benefit performance.

[5] `http://wiki.apertium.org/wiki/Apertium_stream_format`

The rules are tested one-by-one, section-by-section, to see if any of them can be applied to the text. Once such a rule is found, the associated FST is executed on the text. As it is possible that a rule that was not applicable to the original text would now run on the modified one, testing is restarted from the first section after each rule application. The process ends when no more applicable rules are found.

4.1 Naïve implementation

The first version of the program used the `apply_down()` *foma* function both for rule application and format conversion. As *fomacg* generated a single FST for a rule, rule testing and execution was done in the same step, by applying the FST. Whether the rule was actually applied or not was decided by comparing the original sentence to the one returned by the function.

The first row in Table 2 shows the running time for the Hungarian grammar. At 6.4s, the naïve implementation runs more than 20 times slower than VISL-CG (see Table 1). Luckily a far cry from the 1,500 reported by Peltonen, but clearly too slow to be of practical use.

4.2 FST composition

Another way to apply a rule is to convert the input sentence into a single-path FSA with the same alphabet as the rules and compose the rule FST on top of it. To check if the rule has actually be applied, the input automaton was intersected with the result. Unfortunately, this method proved to be much slower than the application-based one; composition alone took 28.3 seconds on our corpus, while the intersection pushed it up to 45s. Therefore we decided to abandon this path altogether.

4.3 Deletion of discarded readings

The original transducers replace the `#0#` in front of discarded readings with `#X#`. Our first optimisation comes from the observation that deleting these readings instead would not make the transducers any more complex, but would shorten the resulting sentence, making subsequent tests faster. Moreover, it allows the engine to recognise actual rule application by simply testing the length of the output to the input sentence, an operation slightly faster than byte-for-byte comparison.

Table 2 reports an approximately 8% improvement. While not self-evident, this benefit remained in effect after our subsequent optimisations.

4.4 FSA-based rule testing

Theoretically, further speed-ups could be achieved by separating rule testing and application, using finite-state automata for the former. Automata are faster than transducers for two reasons: first, there is no need to assemble an output; and second, a FSA can be determinised and minimised, while *foma* can only make a FST deterministic by treating it as a FSA with an alphabet of the original input:output pairs, which does not entail determinism in the input.

As the fourth row in table 2 shows, the idea does not immediately translate well to practice. The fault lies with the `apply_down()` function, which, being the only method of running a finite-state machine in *foma*, was designed to support all features of the library. It treats automata as identity transducers, and fails to capitalise on the aforementioned advantages of the former. In order to benefit from FSA-based testing then, a custom function is required.

4.5 Custom FSA/FST application

The `apply_down()` function supports the following features (Hulden, 2009a):

- Conversion of the text to symbols (single- and multi-character)
- Regular transitions and flag diacritics
- Three types of search in the transition matrix (linear, binary and indexed)
- Deterministic and non-deterministic operation
- Iterators (multiple invocations iterate the non-deterministic outputs)

Our use-case makes most of these features surplus to requirements. *fomacg* uses multi-character symbols, but not flag diacritics. To maximise the performance gains, the rule testing automata must be minimal (hence deterministic), so there was no need for non-determinism and iterators. Finally, by modifying *fomacg* to sort the edges of all grammar machines, we could ensure that binary transition search alone suffices.

The custom FSA applier function that implements only the necessary features was employed for both rule testing and finding the delimiter cohort. As a result, running time went down to 1.45 seconds (see table 2), a 75% improvement.

A similar function was written for input-deterministic minimal transducers. While not applicable to the non-deterministic rule FSTs, it could replace `apply_down()` for the conversion between the Apertium and the *fomacg* formats, further reducing the running time to 1.275 seconds.

What we can take home from the last two sections is that when speed is paramount, relying blindly on generic libraries may not only lead to suboptimal performance, but may also produce counterintuitive results.

Conversely, libraries may benefit from including specialised implementations for different use-cases. For example, *foma* has all the information at hand to decide if a FST is deterministic, whether it supports binary search or not, etc. and so, providing specialised functions (even private ones hidden behind `apply_down()`) would improve its performance substantially in certain situations.

4.6 Exploiting CG structure

In this chapter, we review the improvements made available by the characteristics of our CG representation. The first of these is functionality: even though the rule FSTs are non-deterministic, the input-output mapping is one-to-one (Hulden, 2011). It was thus possible to implement the non-deterministic version of the FST runner function described in the last section without the need for an iterator feature, and to use it for rule application. The last usage of the generic `apply_down()` function thus eradicated, the running time dropped to 1.05 seconds (see table 2).

Internally *foma*, similarly to other FST toolkits, represents elements of the Σ alphabet as integers. The conversion of text into tokens in Σ is a step usually taken for granted in the literature, but it contributes to the execution time of an FST to a significant extent. In *foma*, token matching is performed by a trie built from the symbols in the automaton's alphabet. Our custom DFSA runner function (see section 4.5) spends about 60% of its time applying this trie.

The two enhancements below have helped to all but negate the cost of token conversion. The first of these exploits the fact that in the `fomacg` format, symbols are separated by space characters. Instead of passing the input string to each FSM, we split it along the spaces, and pass the resulting string vector to the machines. This is a rather small change, and while the Hungarian grammar benefited almost nothing, the running time of the Breton grammar improved by 40%.

The second enhancement came from the observation that all rule testing automata and rule transducers accept the same CG tags. It is thus possible to generate an automaton whose alphabet is the union of those of the other machines. This automaton could be used to convert the input sentence into a vector of Σ ids, and then this vector could be sent to the other machines, relinquishing the need of repeated conversions.

Both *fomacg* and *fomacg-proc* had to be modified to account for the changes. The former now creates the converter FSA and saves it as the second machine in the binary grammar file. Also, since the ids that correspond to a symbol are unique to each machine, we added a post-processing phase that replaces the ids with the "canonical" ones in the converter FSA. *fomacg-proc* then converts the input to ids using the converter automaton's trie, and sends the vector to the rule machines. The rule machines treat the vector as their input, with a caveat: ids not in the alphabet of the machine in question are replaced by `@IDENTITY_SYMBOL@`, so that they are handled in the same way as before.

Table 2 shows that factoring the symbol conversion out from the individual machines resulted in huge savings: the running time of the Hungarian setup improved by 70% to 0.32 second; the Breton one by 40% to 1.55 seconds.

Table 2: Effects of the optimisations on running time

Version	Hungarian	Breton
Naïve (4.1)	6.4s	–
Composition (4.2)	45s	–
Delete readings (4.3)	5.9s	–
FSA rule testing (4.4)	10s	–
Custom FSA runner (4.5)	1.45s	–
Custom format-FST (4.5)	1.275s	6.8s
Input partitioning (4.6)	1.15s	4s
Custom rule applier (4.6)	1.05s	2.6s
One-time conversion (4.6)	0.32s	1.55s

5 Complexity analysis

Tapanainen (1999) proves that the worst-case time complexity for disambiguating a sentence in his CG-2 parser is $\mathcal{O}(n^3k^2G)$, where n is the length of the sentence, k is the maximum number of readings per word, and the grammar consists of G rules. The explanation is as follows: testing a cohort with a single rule can be done in $\mathcal{O}(nk)$; the whole sentence in $\mathcal{O}(n^2k)$. This process must be repeated for each rule, yielding $\mathcal{O}(n^2kG)$. Finally, in the worst case, a rule only removes a single reading, so it takes $n(k-1)$ rounds to disambiguate the sentence, resulting in the aforementioned bound.

Hulden (2011) showed that if the rules are compiled to transducers, they can be applied to the whole sentence in $\mathcal{O}(nk)$ time, thus decreasing the complexity to $O(n^2k^2G)$, instead of the $O(n^2k)$ suggested by Tapanainen. To be more precise, applying a rule transducer takes $O(nkT)$ time, where the constant T is the size of the FST. While T may be rather large, rule transducers may be factored into bimachines, which removes the constant. Hence, a disambiguating bimachine for one CG rule can be applied to a sentence of nk tokens in $O(nk)$ (linear) time. However, *fomacg* only includes CG rule-to-transducer compilation and does not include bimachine factorization as of yet.

While this work has left the theoretical limit untouched thus far, it improved on three aspects of the complexity. First, unlike *foma*, our specialised FST application functions can take advantage of the properties of automata and bimachines, and actually run them in $\mathcal{O}(nk)$ time. Second, the constant in the $\mathcal{O}$ has been decreased as a result of extensive optimisation. Third, rule testing automata have been introduced which, being minimal, can also be applied in $\mathcal{O}(nk)$ time. Assume that in a round G_a rules can be applied to the sentence and G_u cannot, $G_a + G_u = G$. With minimal automata for rule testing the round finishes with $2G_a + G_u$ machine applications, instead of the $2G$ required by bimachines. The facts that usually $G_a << G$ and that automata can be applied faster than transducers result in a performance improvement over the pure bimachine setup.

5.1 Beyond the $\mathcal{O}(n^2k^2G)$ bound

This section presents an idea that allows the system to theoretically overcome the $\mathcal{O}(n^2k^2G)$ average complexity bound. This section describes the method, and investigates its feasibility; the next section contains the evaluation.

The idea is based on the fact that regular languages are closed under the union operator. If there are two automata, FSA_{G_a} and FSA_{G_b} that test the rules G_a and G_b, respectively, then it follows that their union, $\mathrm{FSA}_{G_{ab}}$, accepts a sentence iff either G_a or G_b is applicable to it. If $\mathrm{FSA}_{G_{ab}}$ is minimised, it runs in $\mathcal{O}(nk)$ time, the same as FSA_{G_a} and FSA_{G_b}.

The union FSA allows us to implement hierarchical rule checking. In this example, testing if any of the two rules match a sentence with only the original automata requires a check with both. Instead, we can apply $\mathrm{FSA}_{G_{ab}}$ first. If neither rule is applicable, the automaton will not accept the sentence, and no further testing is required. If one of the rules is, FSA_{G_a} (or equivalently, FSA_{G_b}) must be run against the sentence to see which. In practice, if we pick two rules from a CG in random, we shall find that the majority of the sentences will not match either, hence the number of tests may be reduced substantially.

There is no need to stop here: we can take two union automata, and merge them again. It is easy to see that if we represent the rule testing automata in a graph, where a node is a FSA, and two nodes are connected iff one was created from the other via union, then we get a binary tree. For a grammar of G rules, a binary tree of $\log G$ levels can be built. Such a tree can confirm with a single test if a sentence does not match any of the rules, or find the matching rule in $\log G + 1$ tests, if one does. Accordingly, in theory this method allows us to improve the average- and best-case complexity bounds of the system to $\mathcal{O}(n^2k^2 \log G)$ and $\mathcal{O}(nk)$, respectively. (Clearly, for grammars with several sections, instead of a single tree that contains all rules, one tree must be built for each section to preserve rule priorities. However, this does not affect the reasoning above).

The bottleneck in this method is memory consumption. The size of the FSA resulting from a non-deterministic union operation is simply the sum of the sizes of the original automata. To achieve the speed-up described above, however, the rule checking automata must be determinised, which may cause them to blow up in size exponentially. Therefore, building a single tree from all rules is not feasible. A compromise solution is to construct a forest of 2–4 level trees, which still fits into the memory and provides similar benefits to a single tree, though to a smaller extent.

5.2 Evaluation

The forest can be assembled in several ways; we experimented with two simple algorithms. Both take as input a list of rule testing automata, which are encapsulated into single-node trees. Before each step, the trees are sorted by the size of the automata in their roots.

The first algorithm, *SmallestFirst*, unifies the two smallest trees in each step, until the root FSA in each tree is above a size limit (1,000 states in our case). The second, *FixedLevel*, aims to create full, balanced binary trees: in a single step, it unifies the smallest tree with the largest, the second smallest with the second largest, etc, and repeats the process until the trees reach a predefined height.

Table 3 lists the running times and memory requirements of the resulting forests. It can be seen that hierarchical rule testing indeed improves performance: even a single level of merging results in 30-42% speedup. However, it is also immediately evident that aside from special cases, the disadvantages overweight the benefits: memory usage and binary size grow exponentially, affecting compilation and grammar loading time as well, and very soon we run into the limits of physical memory. Unless a method is found that reduces memory usage substantially, we have to give up on hierarchical rule testing.

Table 3: Performance and storage requirements of rule testing trees
* State count limit was 500 † Reached limit of physical memory

Language	Algorithm	Initialisation	Disambiguation	Memory	File size
Hungarian	(flat)	0.028s	0.32s	0.5%	60kB
Hungarian	FixedLevel(3)	0.77s	0.235s	2.1%	7.1MB
Hungarian	Smallest First	0.62s	0.234s	1.9%	5.9MB
Breton	(flat)	0.5s	1.55s	5.1%	1.5MB
Breton	FixedLevel(2)	1.8s	1.09s	9.6%	7.4MB
Breton	Smallest First	11.14s	1.05s	28.7%	60MB
Finnish	(flat)	1.5s	22.87s	21.8%	7.2MB
Finnish	FixedLevel(2)	3.64s	13.28s	32.3%	28MB
Finnish	SmallestFirst*	20.75s	9.95s	–†	198MB

6 Memory savings

The use of a single converter automaton has not only resulted in improved performance, but it has also opened a way to decrease the storage space requirements of the grammar as well. The trie that converts the machine's alphabet to integer ids in *foma* takes up space; depending on the number and length of the symbols in bytes, this trie may be responsible for a considerable portion of the memory footprint of an

automaton. Given the number of rules in an average CG grammar, it is easy to see how this trivial subtask may affect the memory consumption of the application, as well as the size of the grammar binary. As the job of token matching has been delegated to the symbol automaton (see section 4.6), we no longer maintain separate tries for all individual FSAs.

Table 4 presents the resulting memory savings. We report numbers for the raw grammars (L1), as well as for two- and three-level condition trees (L2-3). It is not surprising that the raw grammars see the largest improvements; here the tries accounted for 70-80% of the memory usage. As the trees get higher, the number of states and edges grows more rapidly than does the number of tries and the savings become more modest.

Table 4: Improvements in memory usage due to removing the sigma trie. Memory consumption is measured as a percentage of the 4GB system memory

Language	Method	Before	After	Reduction
Hungarian	L1	0.5%	0.1%	80%
Hungarian	L3	2.1%	1.5%	28.57%
Breton	L1	5.1%	1.3%	74.5%
Breton	L2	9.6%	4.4%	54.16%
Finnish	L1	21%	4.1%	80.47%
Finnish	L2	32.3%	8.9%	72.44%

We explored other options as well to reduce the size of rule condition trees. Unfortunately, most methods aimed at FSA compression in the literature are either already implemented in *foma* (e.g. as row-indexed transition matrix, see Kiraz (2001)), or are aimed at automata with a regular structure, such as morphological analysers (Huet, 2003; Huet, 2005; Drobac et al., 2014). Without further support, the approximately 30% saving achieved by our method for a three-level condition tree alone is not enough to redeem hierarchical rule checking.

A task-specific framework, one based on inward deterministic automata has been proposed for CG parsing (Yli-Jyrä, 2011). The paper reports a binary size similar to the original grammar size. However, as the framework breaks away from the practice of direct rule application followed in this paper and in related literature (Hulden, 2011; Peltonen, 2011), closer inspection remains as future work.

7 Conclusions

We set out with the goal of creating a fast constraint grammar parser based on finite-state technology. Our aim was to achieve better performance on the task of morphological disambiguation than the current state-of-the-art parser VISL CG-3. We used the CG grammars available in the Apertium machine translation project.

Our goals were partially fulfilled: while the speed of our parser falls short of that of VISL CG-3 — with the exception of the execution of very small grammars — we have made advances on the state-of-the-art free/open-source FST implementations of CG. We based our system on the *fomacg* compiler, and extended it in several ways. Our parser uses optimised FST application methods instead of the generic *foma* variant used by previous implementations, thereby achieving better performance. Further optimisations, both memory and runtime, were made by exploiting the properties of FSTs generated from a CG. We report real-world performance measurements with and without these optimisations, so their efficacy can be accurately evaluated. A new method for rule testing has also been proposed, which in theory is capable of reducing the worst-case complexity bound of CG application to $\mathcal{O}(n^2k^2 \log G)$. Unfortunately, the method has yet to be proven feasible in practice.

Our main finding is that implementation matters: an FST library which is too generic hinders performance and can even make a theoretically faster algorithm slower in practice. Using bimachines and rule testing automata should have sped up rule application, but only did so after we implemented our own, specialised FST functions. Since *foma* has all necessary information about an FST in place to decide

the right application method, incorporating our functions into it, or other FST libraries, could benefit applications beyond the scope of CG.

Acknowledgements

This research was supported through an Apertium project in the 2013 Google Summer of Code.[6]

References

Eckhard Bick. 2000. *The parsing system" Palavras": Automatic grammatical analysis of Portuguese in a constraint grammar framework*. Aarhus Universitetsforlag.

Eckhard Bick. 2011. Constraint grammar applications. In *Proceedings of the NODALIDA 2011 Workshop: Constraint Grammar Applications*, page iv.

Tino Didriksen. 2011. Constraint grammar manual: 3rd version of the CG formalism variant.

Senka Drobac, Krister Lindén, Tommi Pirinen, and Miikka Silfverberg. 2014. Heuristic hyper-minimization of finite state lexicons. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Mikel L Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M Tyers. 2011. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144.

Gérard Huet. 2003. Automata mista. *Lecture notes in computer science*, pages 359–372.

Gérard Huet. 2005. A functional toolkit for morphological and phonological processing, application to a Sanskrit tagger. *Journal of Functional Programming*, 15(4):573–614.

Mans Hulden. 2009a. *Finite-state machine construction methods and algorithms for phonology and morphology*. Ph.D. thesis.

Mans Hulden. 2009b. Foma: a finite-state compiler and library. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*, pages 29–32. Association for Computational Linguistics.

Mans Hulden. 2011. Constraint grammar parsing with left and right sequential finite transducers. In *Proceedings of the 9th International Workshop on Finite State Methods and Natural Language Processing*, pages 39–47. Association for Computational Linguistics.

Fred Karlsson. 1990. Constraint grammar as a framework for parsing running text. In *Proceedings of the 13th conference on Computational linguistics-Volume 3*, pages 168–173. Association for Computational Linguistics.

George Anton Kiraz. 2001. Compressed storage of sparse finite-state transducers. In *Automata Implementation*, pages 109–121. Springer.

Janne Peltonen. 2011. A finite state constraint grammar parser. In *Proceedings of the NODALIDA 2011 Workshop: Constraint Grammar Applications*, pages 35–40.

Pasi Tapanainen. 1996. *The constraint grammar parser CG-2*. University of Helsinki, Department of General Linguistics.

Pasi Tapanainen. 1999. *Parsing in two frameworks: finite-state and functional dependency grammar*. Ph.D. thesis, University of Helsinki, Department of General Linguistics.

Viktor Trón, András Kornai, György Gyepesi, László Németh, Péter Halácsy, and Dániel Varga. 2005. Hunmorph: open source word analysis. In *Proceedings of the Workshop on Software*, pages 77–85. Association for Computational Linguistics.

Francis M Tyers. 2010. Rule-based Breton to French machine translation. In *Proceedings of the 14th Annual Conference of the European Association of Machine Translation, EAMT10*, pages 174–181.

Anssi Yli-Jyrä. 2011. An efficient constraint grammar parser based on inward deterministic automata. In *Proceedings of the NODALIDA 2011 Workshop: Constraint Grammar Applications*, pages 50–60.

[6] `https://google-melange.appspot.com/gsoc/project/details/google/gsoc2013/davidnemeskey/5764017909923840`

Language for Communication: Language as Rational Inference

Edward Gibson
Massachusetts Institute of Technology
Brain and Cognitive Sciences Department
Cambridge, MA, USA
egibson@mit.edu

Invited Speaker Abstract

Perhaps the most obvious hypothesis for the evolutionary function of human language is for use in communication. Chomsky has famously argued that this is a flawed hypothesis, because of the existence of such phenomena as ambiguity. Furthermore, he argues that the kinds of things that people tend to say are not short and simple, as would be predicted by communication theory. Contrary to Chomsky, my group applies information theory and communication theory from Shannon (1948) in order to attempt to explain the typical usage of language in comprehension and production, together with the structure of languages themselves. First, we show that ambiguity out of context is not only not a problem for an information-theoretic approach to language, it is a feature. Second, we show that language comprehension appears to function as a noisy channel process, in line with communication theory. Given si, the intended sentence, and sp, the perceived sentence we propose that people maximize $P(s_i|s_p)$, which is equivalent to maximizing the product of the prior $P(s_i)$ and the likely noise processes $P(s_i \rightarrow s_p)$.

We show that several predictions of this way of thinking of language are true:

1. the more noise that is needed to edit from one alternative to another leads to lower likelihood that the alternative will be considered;
2. in the noise process, deletions are more likely than insertions;
3. increasing the noise increases the reliance on the prior (semantics); and
4. increasing the likelihood of implausible events decreases the reliance on the prior.

Third, we show that this way of thinking about language leads to a simple re-thinking of the P600 from the ERP literature. The P600 wave was originally proposed to be due to people's sensitivity to syntactic violations, but there have been many instances of problematic data in the literature for this interpretation. We show that the P600 can best be interpreted as sensitivity to an edit in the signal, in order to make it more easily interpretable.

Finally, we discuss how thinking of language as communication can explain aspects of the origin of word order. Some recent evidence suggests that subject-object-verb (SOV) may be the default word order for human language. For example, SOV is the preferred word order in a task where participants gesture event meanings (Goldin-Meadow et al. 2008). Critically, SOV gesture production occurs not only for speakers of SOV languages, but also for speakers of SVO languages, such as English, Chinese, Spanish (Goldin-Meadow et al. 2008) and Italian (Langus and Nespor, 2010). The gesture-production task therefore plausibly reflects default word order independent of native language. However, this leaves open the question of why there are so many SVO languages (41.2% of languages; Dryer, 2005). We propose that the high percentage of SVO languages cross-linguistically is due to communication pressures over a noisy channel. We provide several gesture experiments consistent with this hypothesis, and we speculate how a noisy channel approach might explain several typical word order patterns that occur in the world's languages.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 781–782, Dublin, Ireland, August 23-29 2014.

References

Matthew S. Dryer. 2005. The order of subject, object and verb. In *The World Atlas of Language Structures*, pages 330–333. Oxford University Press, Oxford, UK.

Susan Goldin-Meadow, Wing Chee So, Aslı Özyürek, and Carolyn Mylander. 2008. The natural order of events: How speakers of different languages represent events nonverbally. *Proceedings of the National Academy of Sciences of the United States of America*, 105(27):9163–9168.

Alan Langusa and Marina Nespor. 2010. Cognitive systems struggling for word order. *Cognitive Psychology*, 60(4):291–318.

C.E. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27:623–656.

Soft Cross-lingual Syntax Projection for Dependency Parsing

Zhenghua Li , Min Zhang*, Wenliang Chen
Provincial Key Laboratory for Computer Information Processing Technology
Soochow University
{zhli13,minzhang,wlchen}@suda.edu.cn

Abstract

This paper proposes a simple yet effective framework of soft cross-lingual syntax projection to transfer syntactic structures from source language to target language using monolingual treebanks and large-scale bilingual parallel text. Here, *soft* means that we only project reliable dependencies to compose high-quality target structures. The projected instances are then used as additional training data to improve the performance of supervised parsers. The major issues for this idea are 1) errors from the source-language parser and unsupervised word aligner; 2) intrinsic syntactic non-isomorphism between languages; 3) incomplete parse trees after projection. To handle the first two issues, we propose to use a probabilistic dependency parser trained on the target-language treebank, and prune out unlikely projected dependencies that have low marginal probabilities. To make use of the incomplete projected syntactic structures, we adopt a new learning technique based on *ambiguous labelings*. For a word that has no head words after projection, we enrich the projected structure with all other words as its candidate heads as long as the newly-added dependency does not cross any projected dependencies. In this way, the syntactic structure of a sentence becomes a parse forest (ambiguous labels) instead of a single parse tree. During training, the objective is to maximize the mixed likelihood of manually labeled instances and projected instances with ambiguous labelings. Experimental results on benchmark data show that our method significantly outperforms a strong baseline supervised parser and previous syntax projection methods.

1 Introduction

During the past decade, supervised dependency parsing has made great progress. However, due to the limitation of scale and genre coverage of labeled data, it is very difficult to further improve the performance of supervised parsers. On the other hand, it is very time-consuming and labor-intensive to manually construct treebanks. Therefore, lots of recent work has been devoted to get help from bilingual constraints. The motivation behind are two-fold. First, a difficult syntactic ambiguity in one language may be very easy to resolve in another language. Second, a more accurate parser on one language may help an inferior parser on another language, where the performance difference may be due to the intrinsic complexity of languages or the scale of accessible labeled resources.

Following the above research line, much effort has been done recently to explore bilingual constraints for parsing. Burkett and Klein (2008) propose a reranking based method for joint constituent parsing of bitext, which can make use of structural correspondence features in both languages. Their method needs bilingual treebanks with manually labeled syntactic trees on both sides for training. Huang et al. (2009) compose useful parsing features based on word reordering information in source-language sentences. Chen et al. (2010a) derive bilingual subtree constraints with auto-parsed source-language sentences. During training, both Huang et al. (2009) and Chen et al. (2010a) require bilingual text with target-language gold-standard dependency trees. All above work shows significant performance gain

*Correspondence author

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 783–793, Dublin, Ireland, August 23-29 2014.

over monolingual counterparts. However, one potential disadvantage is that bilingual treebanks and bitext with one-side annotation are difficult to obtain. Therefore, They usually conduct experiments on treebanks with a few thousand sentences. To break this constraint, Chen et al. (2011) extend their work in Chen et al. (2010a) and translate text of monolingual treebanks to obtain bilingual treebanks with a statistical machine translation system.

This paper explores another line of research and aims to boost the state-of-the-art parsing accuracy via syntax projection. Syntax projection typically works as follows. First, we train a parser on source-language treebank, called a source parser. Then, we use the source parser to produce automatic syntactic structures on the source side of bitext. Next, with the help of automatic word alignments, we project the source-side syntactic structures into the target side. Finally, the target-side structures are used as gold-standard to train new parsing models of target language. Previous work on syntax projection mostly focuses on unsupervised grammar induction where no labeled data exists for target language (Hwa et al., 2005; Spreyer and Kuhn, 2009; Ganchev et al., 2009; Liu et al., 2013). Smith and Eisner (2009) propose quasi-synchronous grammar for cross-lingual parser projection and assume the existence of hundreds of target language annotated sentences. Similar to our work in this paper, Jiang et al. (2010) try to explore projected structures to further improve the performance of statistical parsers trained on full-scale monolingual treebanks (see Section 4.4 for performance comparison).

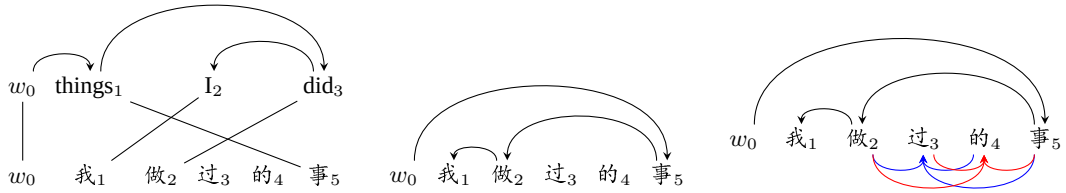
The major issues for syntax projection are 1) errors from the source-language parser and unsupervised word aligner; 2) intrinsic syntactic non-isomorphism between languages; 3) incomplete parse trees after projection. Hwa et al. (2005) propose a simple projection algorithm based on the *direct correspondence assumption* (DCA). They apply post-editing to the projected structures with a set of hand-crafted heuristic rules, in order to handle some typical cross-lingual syntactic divergences. Similarly, Ganchev et al. (2009) manually design several language-specific constrains during projection, and use projected partial structures as soft supervision during training based on posterior regularization (Ganchev et al., 2010). To make use of projected instances with incomplete trees, Spreyer and Kuhn (2009) propose a heuristic method to adapt training procedures of dependency parsing. Instead of directly using incomplete trees to train dependency parsers, Jiang et al. (2010) train a local dependency/non-dependency classifier on projected syntactic structures, and use outputs of the classifier as auxiliary features to help supervised parsers. One potential common drawback of above work is the lack of a systematic way to handle projection errors and incomplete trees.

Different from previous work, this paper proposes a simple yet effective framework of soft syntax projection for dependency parsing, and provides a more elegant and systematic way to handle the above issues. First, we propose to use a probabilistic parser trained on target-language treebank, and prune unlikely projected dependencies which have very low marginal probabilities. Second, we adopt a new learning technique based on ambiguous labelings to make use of projected incomplete trees for training. For a word that has no head words after projection, we enrich the projected structure by adding all possible words as its heads as long as the newly-added dependency does not cross any projected dependencies. In this way, the syntactic structure of a sentence becomes a parse forest (ambiguous labelings) instead of a single parse tree. During training, the objective is to maximize the mixed likelihood of manually labeled instances and projected instances with ambiguous labelings. Experimental results on benchmark data show that our method significantly outperforms a strong baseline supervised parser and previous syntactic projection methods.

2 Syntax Projection

Given an input sentence $\mathbf{x} = w_0 w_1 ... w_n$, a dependency tree is $\mathbf{d} = \{(h, m) : 0 \leq h \leq n, 0 < m \leq n\}$, where (h, m) indicates a directed arc from the *head* word w_h to the *modifier* w_m, and w_0 is an artificial node linking to the root of the sentence.

Syntax projection aims to project the dependency tree $\mathbf{d}^s$ of a source-language sentence $\mathbf{x}^s$ into the dependency structure of its target-language translation $\mathbf{x}$ via word alignments $\mathbf{a}$, where a word alignment $a_i = z$ means the target-side word w_i is aligned into the source-side word w_z^s, as depicted in Figure 1(a) and Figure 1(b). For simplicity, we avoid one-to-many alignments by keeping the one with highest



(a) Source tree and word alignments (b) Projected incomplete tree (c) Forest (ambiguous labelings)

Figure 1: Illustration of syntax projection from English to Chinese with a sentence fragment. The two Chinese auxiliary words, "过$_3$" (past tense marker) and "的$_4$" (relative clause marker), are not aligned to any English words.

marginal probability when the target word is aligned to multiple source words. We first introduce a simple syntax projection approach based on DCA (Hwa et al., 2005), and then propose two extensions to handle parsing and aligning errors and cross-lingual syntactic divergences.

Projection with DCA. If two target words w_i and w_j are aligned to two different source words $w^s_{a_i}$ and $w^s_{a_j}$, and the two words compose a dependency in the source tree $(a_i, a_j) \in \mathbf{d}^s$, then add a dependency (i, j) into the projected syntactic structure. For example, as shown in Figure 1(a), the two Chinese words "做$_2$" and "事$_5$" are aligned to the two English words "did$_3$" and "things$_1$", and the dependency "things$_1$$\curvearrowright$did$_3$" is included in the source tree. Therefore, we project the dependency into the target side and add a dependency "做$_2$$\curvearrowleft$事$_5$" into the projected structure, as shown in Figure 1(b). An obvious drawback of DCA is that it may produce many wrong dependencies due to the errors in the automatic source-language parse trees and word alignments. Even with manual parse trees and word alignments, syntactic divergences between languages can also lead to projection errors.

Pruned with target-side marginals. To overcome the weakness of DCA, we propose to use target-side marginal probabilities to constrain the projection process and prune obviously bad projections. We train a probabilistic parser on an existing target-side treebank. For each projected dependency, we compute its marginal probability with the target parser, and prune it off the projected structure if the probability is below a *pruning threshold* λ_p. Our study shows that dependencies with very low marginal probabilities are mostly wrong (Figure 2).

Supplemented with target-side marginals. To further improve the quality of projected structures, we add dependencies with high marginal probabilities according to the target parser. Specifically, if a target word w_j obtain a head word w_i after projection, and if another word w_k has higher marginal probability than a *supplement threshold* λ_s to be the head word of w_j, then we also add the dependency (k, j) into the projected structure. In other words, we allow one word to have multiple heads so that the projected structure can cover more correct dependencies.

From incomplete tree to forest. Some words in the target sentence may not obtain any head words after projection due to incomplete word alignments or the pruning process, which leads to incomplete parse trees after projection. Also, some words may have multiple head words resulting from the supplement process. To handle these issues, we first convert the projected structures into parse forests, and then propose a generalized training technique based on ambiguous labelings to make use of the projected instances. Specifically, if a word does not have head words after projection, we simply add into the projected structure all possible words as its candidate heads as long as the newly-added dependency does not cross any projected dependencies, as illustrated in Figure 1(c). We introduce three new dependencies to compose candidate heads for the unattached word "过$_3$". Note that it is illegal to add the dependency "我$_1$$\curvearrowright$过$_3$" since it would cross the projected dependency "做$_2$$\curvearrowleft$事$_5$".

3 Dependency Parsing with Ambiguous Labelings

In parsing community, two mainstream methods tackle the dependency parsing problem from different perspectives but achieve comparable accuracy on a variety of languages. Graph-based methods view the problem as finding an optimal tree from a fully-connected directed graph (McDonald et al., 2005; McDonald and Pereira, 2006; Carreras, 2007; Koo and Collins, 2010), while transition-based methods try to find a highest-scoring transition sequence that leads to a legal dependency tree (Yamada and Matsumoto, 2003; Nivre, 2003; Zhang and Nivre, 2011).

3.1 Graph-based Dependency Parser (GParser)

We adopt the graph-based paradigm because it allows us to elegantly derive our CRF-based probabilistic parser, which is required to compute the marginal probabilities of dependencies and likelihood of both manually labeled data and unannotated bitext with ambiguous labelings. The graph-based method factors the score of a dependency tree into scores of small subtrees $\mathbf{p}$.

$$Score(\mathbf{x}, \mathbf{d}; \mathbf{w}) = \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, \mathbf{d}) = \sum_{\mathbf{p} \subseteq \mathbf{d}} Score(\mathbf{x}, \mathbf{p}; \mathbf{w}) \tag{1}$$

We adopt the second-order model of McDonald and Pereira (2006) as our core parsing algorithm,[1] which defines the score of a dependency tree as:

$$Score(\mathbf{x}, \mathbf{d}; \mathbf{w}) = \sum_{\{(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{dep} \cdot \mathbf{f}_{dep}(\mathbf{x}, h, m) + \sum_{\{(h,s),(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{sib} \cdot \mathbf{f}_{sib}(\mathbf{x}, h, s, m) \tag{2}$$

where $\mathbf{f}_{dep}(\mathbf{x}, h, m)$ and $\mathbf{f}_{sib}(\mathbf{x}, h, s, m)$ are feature vectors corresponding to two kinds of subtree; $\mathbf{w}_{dep/sib}$ are the feature weight vectors; the dot product gives the scores contributed by the corresponding subtrees. We adopt the state-of-the-art syntactic features proposed in Bohnet (2010).

3.2 Probabilistic CRF-based GParser

Previous work on dependency parsing mostly adopts linear models and online perceptron training, which lack probabilistic explanations of dependency trees and likelihood of the training data. Instead, we build a log-linear CRF-based probabilistic dependency parser, which defines the probability of a dependency tree as:

$$p(\mathbf{d}|\mathbf{x}; \mathbf{w}) = \frac{exp\{Score(\mathbf{x}, \mathbf{d}; \mathbf{w})\}}{Z(\mathbf{x}; \mathbf{w})}; \quad Z(\mathbf{x}; \mathbf{w}) = \sum_{\mathbf{d}' \in \mathcal{Y}(\mathbf{x})} exp\{Score(\mathbf{x}, \mathbf{d}'; \mathbf{w})\} \tag{3}$$

where $Z(\mathbf{x})$ is the normalization factor and $\mathcal{Y}(\mathbf{x})$ is the set of all legal dependency trees for $\mathbf{x}$.

3.3 Likelihood and Gradient of Training Data with Ambiguous Labelings

Traditional CRF models assume one gold-standard label for each training instance, which means each sentence is labeled with a single parse tree in the case of parsing. To make use of projected instances with ambiguous labelings, we propose to use a generalized training framework which allows a sentence to have multiple parse trees (forest) as its gold-standard reference (Täckström et al., 2013). The goal of the training procedure is to maximize the likelihood of the training data, and the model is updated to improve the probabilities of parse forests, instead of single parse trees. In other words, the model has the flexibility to distribute the probability mass among the parse trees inside the forest, as long as the probability of the forest improves. In this generalized framework, a traditional instance labeled with a single parse tree can be regarded as a special case that the forest contains only one parse tree.

The probability of a sentence $\mathbf{x}$ with ambiguous labelings $\mathcal{F}$ is defined as the sum of probabilities of all parse tree $\mathbf{d}$ contained in the forest $\mathcal{F}$:

$$p(\mathcal{F}|\mathbf{x}; \mathbf{w}) = \sum_{\mathbf{d} \in \mathcal{F}} p(\mathbf{d}|\mathbf{x}; \mathbf{w}) \tag{4}$$

[1] Higher-order models of Carreras (2007) and Koo and Collins (2010) can achieve a little bit higher accuracy, but suffer from higher time cost of $O(n^4)$ and system complexity. Our method is applicable to the third-order model.

	Train	Dev	Test
PTB	39,832	1,346	2416
CTB5	16,091	803	1,910
CTB5X	18,104	352	348
Bitext	0.9M	–	–

Table 1: Data sets (in sentence number).

Suppose the training data set is $\mathcal{D} = \{(\mathbf{x}_i, \mathcal{F}_i)\}_{i=1}^{N}$. Then the log likelihood of $\mathcal{D}$ is:

$$\mathcal{L}(\mathcal{D}; \mathbf{w}) = \sum_{i=1}^{N} \log p(\mathcal{F}_i | \mathbf{x}_i; \mathbf{w}) \tag{5}$$

Then we can derive the partial derivative of the log likelihood with respect to $\mathbf{w}$:

$$\frac{\partial \mathcal{L}(\mathcal{D}; \mathbf{w})}{\partial \mathbf{w}} = \sum_{i=1}^{N} \left(\sum_{\mathbf{d} \in \mathcal{F}_i} \tilde{p}(\mathbf{d} | \mathbf{x}_i, \mathcal{F}_i; \mathbf{w}) \mathbf{f}(\mathbf{x}_i, \mathbf{d}) - \sum_{\mathbf{d} \in \mathcal{Y}(\mathbf{x}_i)} p(\mathbf{d} | \mathbf{x}_i; \mathbf{w}) \mathbf{f}(\mathbf{x}_i, \mathbf{d}) \right) \tag{6}$$

where $\tilde{p}(\mathbf{d} | \mathbf{x}_i, \mathcal{F}_i; \mathbf{w})$ is the probability of $\mathbf{d}$ under the space constrained by the parse forest $\mathcal{F}_i$:

$$\tilde{p}(\mathbf{d} | \mathbf{x}_i, \mathcal{F}_i; \mathbf{w}) = \frac{\exp\{Score(\mathbf{x}_i, \mathbf{d}; \mathbf{w})\}}{Z(\mathbf{x}_i, \mathcal{F}_i; \mathbf{w})}; \quad Z(\mathbf{x}_i, \mathcal{F}_i; \mathbf{w}) = \sum_{\mathbf{d} \in \mathcal{F}_i} \exp\{Score(\mathbf{x}_i, \mathbf{d}; \mathbf{w})\} \tag{7}$$

The first term in Eq. (6) is the model expectations in the search space constrained by $\mathcal{F}_i$, and the second term is the model expectations in the complete search space $\mathcal{Y}(\mathbf{x}_i)$. Since $\mathcal{Y}(\mathbf{x}_i)$ contains exponentially many legal dependency trees, direct calculation of the second term is prohibitive. Instead, we can use the classic Inside-Outside algorithm to efficiently compute the second term within $O(n^3)$ time complexity, where n is the length of the input sentence. Similarly, the first term can be solved by running the Inside-Outside algorithm in the constrained search space $\mathcal{F}_i$.

3.4 Stochastic Gradient Descent (SGD) Training

With the likelihood gradients, we apply L2-norm regularized SGD training to iteratively learn the feature weights $\mathbf{w}$ for our CRF-based baseline and bitext-enhanced parsers. We follow the implementation in CRFsuite.[2] At each step, the algorithm approximates a gradient with a small subset of training examples, and then updates the feature weights. Finkel et al. (2008) show that SGD achieves optimal test performance with far fewer iterations than other optimization routines such as L-BFGS. Moreover, it is very convenient to parallel SGD since computation among examples in the same batch is mutually independent.

Once the feature weights $\mathbf{w}$ are learnt, we can parse the test data and try to find the optimal parse tree with the Viterbi decoding algorithm in $O(n^3)$ parsing time (Eisner, 2000; McDonald and Pereira, 2006).

$$\mathbf{d}^* = \arg\max_{\mathbf{d} \in \mathcal{Y}(\mathbf{x})} p(\mathbf{d} | \mathbf{x}; \mathbf{w}) \tag{8}$$

4 Experiments and Analysis

To verify the effectiveness of our proposed method, we carry out experiments on English-to-Chinese syntax projection, and aim to enhance our baseline Chinese parser with additional training instances projected from automatic English parse trees on bitext. For **monolingual treebanks**, we use Penn English Treebank (PTB) and Penn Chinese Treebank 5.1 (CTB5). For English, we follow the standard practice to split the data into training (sec 02-21), development (sec 22), and test (sec 23). For CTB5, we adopt the data split of (Duan et al., 2007). We convert the original bracketed structures into dependency

[2] http://www.chokkan.org/software/crfsuite/

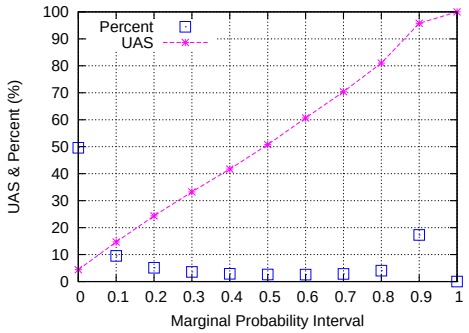


Figure 2: Distribution (Percent) and accuracy (UAS) of dependencies under different marginal probability interval for Chinese baseline parser on CTB5 development set. For example, 0.8 at x-axis means the interval $[0.8, 0.9)$.

structures using Penn2Malt with its default head-finding rules. We build a CRF-based bigram part-of-speech (POS) tagger with the features described in (Li et al., 2012b), and produce POS tags for all train/development/test datasets and bitext (10-way jackknifing for training datasets). The tagging accuracy on test sets is 97.3% on English and 94.0% on Chinese.

To compare with the recent work on syntax projection of Jiang et al. (2010) who use a smaller test dataset, we follow their data split of CTB5 and use gold-standard POS tags during training and test. We refer to this setting as CTB5X.

For **bitext**, we collect a parallel corpus from FBIS news (LDC03E14, 0.25M sentence pairs), United Nations (LDC04E12, 0.62M), IWSLT2008 (0.04M), and PKU-863 (0.2M). After corpus cleaning, we obtain a large-scale bilingual parallel corpus containing 0.9M sentence pairs. We run the unsupervised BerkeleyAligner[3] (Liang et al., 2006) for 4 iterations to obtain word alignments. Besides hard alignments, we also make use of posterior probabilities to simplify one-to-many alignments to one-to-one as discussed in Section 2. Table 1 shows the data statistics.

For training both the baseline and bitext-enhanced parsers, we set the batch size to 100 and run SGD until a maximum iteration number of 50 is met or the change on likelihood of training data becomes too small. Since the number of projected sentences is much more than that of manually labeled instances (0.9M vs. 16K), it is likely that the projected data may overwhelm manually labeled data during training. Therefore, we adopt a simple corpus-weighting strategy. Before each iteration, we randomly sample 50K projected sentences and 15K manually labeled sentences from all training data, and run SGD to train feature weights using the sampled data. To speed up training, we adopt multi-thread implementation of gradient computations in the same batch. It takes about 1 day to train our bitext-enhanced parser for one iteration using a single CPU core, while using 24 CPU cores only needs about 2 hours.

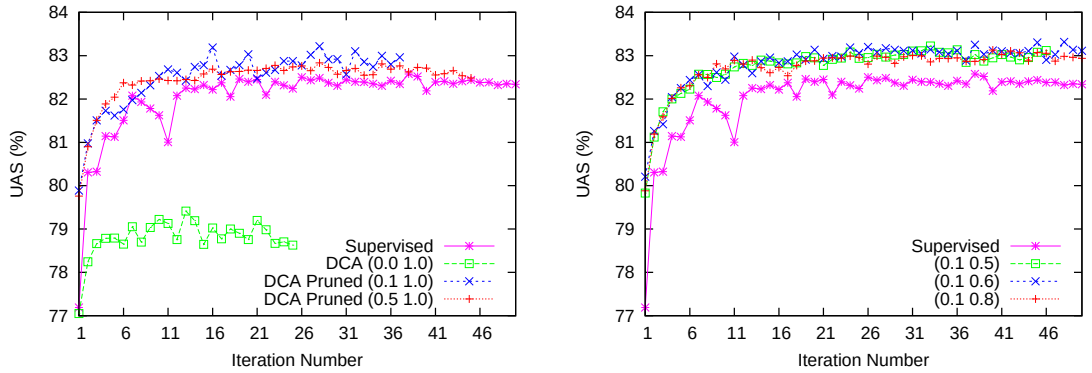
We measure parsing performance using unlabeled attachment score (UAS, percent of words with correct heads), excluding punctuation marks. For significance test, we adopt Dan Bikel's randomized parsing evaluation comparator (Noreen, 1989).[4]

4.1 Analysis on Marginal Probabilities

In order to gain insights for parameter settings of syntax projection, we analyse the distribution and accuracy of dependencies under different marginal probability interval. We train the baseline Chinese parser on CTB5 train set, and use the parser to produce the marginal probabilities of all dependencies for sentences in CTB5 development set. We discard all dependencies that have a marginal probability less than 0.0001 for better illustration. Figure 2 shows the results, where we can see that UAS is roughly proportional to marginal probabilities. In other word, dependencies with higher marginal probabilities are more accurate. For example, dependencies with probabilities under interval $[0.8, 0.9)$ has a 80% chance to be correct. From another aspect, we can see that 50% of dependencies fall in probability

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

[4] http://www.cis.upenn.edu/~dbikel/software.html



(a) Parameters for DCA and DCA Pruned (b) Parameters for DCA Pruned & Supplemented

Figure 3: Performance with different parameter settings of (λ_p λ_s) on CTB5 development set.

interval $[0, 0.1)$, and such dependencies have very low accuracy (4%). These observations are helpful for our parameter selection and methodology study during syntax projection.

4.2 Results of Syntax Projection on Development Dataset

We apply the syntax projection methods described in Section 2 to the bilingual text, and use the projected sentences with ambiguous labelings as additional training instances to train new Chinese parsers based on the framework described in Section 3. Figure 3 shows the UAS curves on development set with different parameters settings. The *pruning threshold* λ_p (see Section 2) balances the quality and coverage of projection. Larger λ_p leads to more accurate but fewer projections. The *supplement threshold* λ_s (see Section 2) balances the size and oracle score of the projected forest. Smaller λ_s can increase the oracle score of the forest by adding more dependencies with lower marginal probabilities, but takes the risk of making the resulted forest too ambiguous and weak to properly supervise the model during training. [5]

The *DCA* method corresponds to the results with $\lambda_p = 0.0$ and $\lambda_s = 1.0$. We can see that DCA largely decreases UAS compared with the baseline CRF-based parser. The reason is that although DCA projects many source-language dependencies to the target side (44% of target-language words obtain head words), it also introduces a lot of noise during projection.

DCA pruned with target-side marginals corresponds to the results with $\lambda_p > 0.0$ and $\lambda_s = 1.0$. Pruning with target-side marginals can clearly improve the projection quality by pruning out bad projections. When $\lambda_p = 0.1$, 31% of target-language words obtain head words, and the model outperforms the baseline parser by 0.6% at peak UAS. When $\lambda_p = 0.5$, the projection ratio decreases to 26% and the improvement is 0.3%. Based on the results, we choose $\lambda_p = 0.1$ in later experiments.

Figure 3(b) presents the results of *DCA pruned & supplemented* with different λ_s. The supplement process adds a small amount of dependencies of high probabilities into the projected forest and therefore increases the oracle score, which provides the model with flexibility to distribute the probability mass to more preferable parse trees. We can see that although the peak UAS does not increase much, the training curve is more smooth and stable than that without supplement. Based on the results, we choose $\lambda_s = 0.6$ in later experiments.

4.3 Final Results and Comparisons on Test Dataset

Table 2 presents the final results on CTB5 test set. For each parser, we choose the parameters corresponding to the iteration number with highest UAS on development set. To further verify the usefulness of syntax projection, we also conduct experiments with self-training, which is known as a typical semi-supervised method. For the standard self-training, we use Chinese-side bitext with self-predicted parse trees produced by the baseline parser as additional training instances, which turns out to be hurtful to parsing performance. This is consistent with earlier results (Spreyer and Kuhn, 2009).

[5] Please note when $\lambda_p + \lambda_s >= 1$, λ_s becomes useless. The reason is that if the probability of a projected dependency (i, j) is larger λ_p, then no other word beside w_i can have a probability larger than λ_s of being the head word of w_j.

	UAS
Baseline Supervised Parser	81.04
Standard Self-training	80.51 (-0.53)
Self-training with Ambiguous Labelings	81.09 (+0.05)
DCA	78.70 (-2.34)
DCA Pruned	81.46 (+0.42 †)
DCA Pruned & Supplemented	81.71 (+0.67 †)

Table 2: UAS on CTB5 test set. † indicate statistical significance at confidence level of $p < 0.01$.

	Supervised	Bitext-enhanced
Jiang et al. (2010)	87.15	87.65 (+0.50)
This work	89.62	90.50 (+0.88 †)

Table 3: UAS on CTB5X test set. † indicate statistical significance at confidence level of $p < 0.01$.

Then, we try a variant of self-training with ambiguous labelings following the practice in Täckström et al. (2013), and use a parse forest composed of dependencies of high probabilities as the syntactic structure of an instance. We can see that ambiguous labelings help traditional self-training, but still have no significant improvement over the baseline parser. Results in Table 2 indicate that our syntax projection method is able to project useful knowledge from source-language parse trees to the target-side forest, and then helps the target parser to learn effective features.

4.4 Comparisons with Previous Results on Syntax Projection on CTB5X

To make comparison with the recent work of Jiang et al. (2010), We rerun the process of syntax projection with CTB5X as the target treebank with the *DCA pruned & supplemented* method ($\lambda_p = 0.1$ and $\lambda_s = 0.6$).[6] Table 3 shows the results. Jiang et al. (2010) employ the second-order MSTParser of McDonald and Pereira (2006) with a basic feature set as their base parser. We can see that our baseline parser is much stronger than theirs. Even though, our approach leads to larger UAS improvement.

This work is different from theirs in a few aspects. First, the purpose of syntax projection in their work is to produce dependency/non-dependency instances which are used to train local classifiers to produce auxiliary features for MSTParser. In contrast, the outputs of syntax projection in our work are partial trees/forests where only reliable dependencies are kept and some words may receive more than one candidate heads. We directly use these partial structures as extra training data to learn model parameters. Second, their work measures the reliability of a projected dependencies only from the perspective of alignment probability, while we adopt a probabilistic parsing model and use target-side marginal probabilities to throw away bad projections, which turns out effective in handling syntactic non-isomorphism and errors in word alignments and source-side parses.

5 Related work

Cross-lingual annotation projection has been applied to many different NLP tasks to help processing resource-poor languages, such as POS tagging (Yarowsky and Ngai, 2001; Naseem et al., 2009; Das and Petrov, 2011) and named entity recognition (NER) (Fu et al., 2011). In another direction, much previous work explores bitext to improve monolingual NER performance based on bilingual constraints (Chen et al., 2010b; Burkett et al., 2010; Li et al., 2012a; Che et al., 2013; Wang et al., 2013).

Based on a universal POS tag set (Petrov et al., 2011), McDonald et al. (2011) propose to train delexicalized parsers on resource-rich language for parsing resource-poor language without use of bitext (Zeman and Resnik, 2008; Cohen et al., 2011; Søgaard, 2011). Täckström et al. (2012) derive cross-lingual clusters from bitext to help delexicalized parser transfer. Naseem et al. (2012) propose selectively sharing to better explore multi-source transfer information.

[6] In the previous draft of this paper, we directly use the projected data with in previous subsection for simplicity, and find that UAS can reach 91.39% (+1.77). The reason is that the CTB5X test is overlapped with CTB5 train. We correct this mistake in this version.

Our idea of training with ambiguous labelings is originally inspired by the work of Täckström et al. (2013) on multilingual parser transfer for unsupervised dependency parsing. They use a delexicalized parser trained on source-language treebank to obtain parse forests for target-language sentences, and re-train a lexicalized target parser using the sentences with ambiguous labelings. Similar ideas of learning with ambiguous labelings are previously explored for classification (Jin and Ghahramani, 2002) and sequence labeling problems (Dredze et al., 2009).

6 Conclusions

This paper proposes a simple yet effective framework of soft cross-lingual syntax projection. We make use of large-scale projected structures as additional training instances to boost performance of supervised parsing models trained on full-set manually labeled treebank. Compared with previous work, we make two innovative contributions: 1) using the marginal probabilities of a target-side supervised parser to control the projection quality with the existence of parsing and aligning errors and cross-lingual syntax divergences; 2) adopting a new learning technique based ambiguous labelings to make use of projected incomplete dependency trees for model training. Experimental results on two Chinese datasets demonstrate the effectiveness of the proposed framework, and show that the bitext-enhanced parser significantly outperforms all baselines, including supervised parsers, semi-supervised parsers based on self-training, and previous syntax projection methods.

Our anonymous reviewers present many great comments, especially on the experimental section. We will improve this work accordingly and release an extended version of this paper at the homepage of the first author. Such extensions include: 1) further exploring source-language parsing probabilities and alignment probabilities to help syntax projection; 2) studying the effect of the scale of source/target treebank and bilingual text.

Acknowledgments

The authors would like to thank Wanxiang Che and Jiang Guo for sharing their bilingual data, and our anonymous reviewers for their critical and insightful comments, which will certainly help our future work. This work was supported by National Natural Science Foundation of China (Grant No. 61373095, 61203314, 61373097).

References

Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of COLING*, pages 89–97.

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

David Burkett, Slav Petrov, John Blitzer, and Dan Klein. 2010. Learning better monolingual models with unannotated bilingual text. In *Proceedings of CoNLL 2010*, pages 46–54.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of EMNLP/CoNLL*, pages 141–150.

Wanxiang Che, Mengqiu Wang, Christopher D. Manning, and Ting Liu. 2013. Named entity recognition with bilingual constraints. In *Proceedings of NAACL 2013*.

Wenliang Chen, Jun'ichi Kazama, and Kentaro Torisawa. 2010a. Bitext dependency parsing with bilingual subtree constraints. In *Proceedings of ACL*, pages 21–29.

Yufeng Chen, Chengqing Zong, and Keh-Yih Su. 2010b. On jointly recognizing and aligning bilingual named entities. In *Proceedings of ACL 2010*.

Wenliang Chen, Jun'ichi Kazama, Min Zhang, Yoshimasa Tsuruoka, Yujie Zhang, Yiou Wang, Kentaro Torisawa, and Haizhou Li. 2011. SMT helps bitext dependency parsing. In *EMNLP*.

Shay B. Cohen, Dipanjan Das, , and Noah A. Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of EMNLP*.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of ACL-HLT 2011*, pages 600–609.

Mark Dredze, Partha Pratim Talukdar, and Koby Crammer. 2009. Sequence learning from data with multiple labels. In *ECML/PKDD Workshop on Learning from Multi-Label Data*.

Xiangyu Duan, Jun Zhao, and Bo Xu. 2007. Probabilistic parsing action models for multi-lingual dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 940–946.

Jason Eisner. 2000. Bilexical grammars and their cubic-time parsing algorithms. In *Advances in Probabilistic and Other Parsing Technologies*, pages 29–62.

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

Ruiji Fu, Bing Qin, and Ting Liu. 2011. Generating chinese named entity data from a parallel corpus. In *Proceedings of IJCNLP 2011*, pages 264–272.

Kuzman Ganchev, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. In *Proceedings of ACL-IJCNLP 2009*, pages 369–377.

Kuzman Ganchev, Jo ao Graca, Jennifer Gillenwater, and Ben Taskar. 2010. Posterior regularization for structured latent variable models. *Journal of Artifical Intellignece Research*.

Liang Huang, Wenbin Jiang, and Qun Liu. 2009. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proceedings of EMNLP*, pages 1222–1231.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Boostrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11(3):311–325.

Wenbin Jiang, , and Qun Liu. 2010. Dependency parsing and projection based on word-pair classification. In *ACL*, pages 897–904.

Rong Jin and Zoubin Ghahramani. 2002. Learning with multiple labels. In *Proceedings of NIPS*.

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

Qi Li, Haibo Li, Heng Ji, Wen Wang, Jing Zeng, and Fei Huang. 2012a. Joint bilingual name tagging for parallel corpora. In *Proceedings of CIKM 2012*.

Zhenghua Li, Min Zhang, Wanxiang Che, and Ting Liu. 2012b. A separately passive-aggressive training algorithm for joint POS tagging and dependency parsing. In *COLING 2012*, pages 1681–1698.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *HLT-NAACL*.

Kai Liu, Yajuan Lü, Wenbin Jiang, and Qun Liu. 2013. Bilingually-guided monolingual dependency grammar induction. In *Proceedings of ACL*.

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

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

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of EMNLP*.

Tahira Naseem, Benjamin Snyder, Jacob Eisentein, and Regina Barzilay. 2009. Multilingual part-of-speech tagging: two unsupervised approaches. *Journal of Artifical Intellignece Research*, 36(1):341–385.

Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *Proceedings of ACL*.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of IWPT*, pages 149–160.

Eric W. Noreen. 1989. *Computer-intensive methods for testing hypotheses: An introduction*. John Wiley & Sons, Inc., New York.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2011. A universal part-of-speech tagset. In *ArXiv:1104.2086*.

David A. Smith and Jason Eisner. 2009. Parser adaptation and projection with quasi-synchronous grammar features. In *Proceedings of EMNLP*, pages 822–831.

Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of ACL 2011*, pages 682–686.

Kathrin Spreyer and Jonas Kuhn. 2009. Data-driven dependency parsing of new languages using incomplete and noisy training data. In *CoNLL*, pages 12–20.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of NAACL-HLT*.

Oscar Täckström, Ryan McDonald, and Joakim Nivre. 2013. Target language adaptation of discriminative transfer parsers. In *Proceedings of NAACL*, pages 1061–1071.

Mengqiu Wang, Wanxiang Che, and Christopher D. Manning. 2013. Joint word alignment and bilingual named entity recognition using dual decomposition. In *Proceedings of ACL 2013*.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT*, pages 195–206.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Proceedings of NAACL 2001*.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of IJCNLP 2008*.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of ACL*, pages 188–193.

Automatic Feature Selection for Agenda-Based Dependency Parsing

Miguel Ballesteros
Natural Language Processing Group
Universitat Pompeu Fabra
Barcelona, Spain
miguel.ballesteros@upf.edu

Bernd Bohnet
School of Computer Science
University of Birmingham
Birmingham, United Kingdom
bohnetb@cs.bham.ac.uk

Abstract

In this paper we present an in-depth study on automatic feature selection for beam-search dependency parsers. The search strategy is inherited from the one implemented in MaltOptimizer, but searches in a much larger set of feature templates that could lead to a higher number of combinations. Our models provide results that are on par with models trained with a larger set of feature templates, and this implies that our models provide faster training and parsing times. Moreover, the results establish the state of the art for some of the languages.

1 Introduction

Finding an optimal and accurate set of feature templates is crucial when training statistical parsers; in fact it is essential when building any machine learning system (Smith, 2011). In dependency parsing, the features are based on the linguistic information that is annotated within the words and the information that is being calculated during the parsing process. Researchers normally tend to include a large set of feature templates in their machine learning models, following the idea that more is always better; however some recent research on feature selection for transition-based parsing (Ballesteros, 2013; Ballesteros and Nivre, 2014) and graph-based parsing (He et al., 2013) have shown that more features are not always better, at least in the case of dependency parsing; models containing more features are always slower in parsing and training time and they do not always provide better results.

This indicates that a smart feature template selection could be the key in the trade-off for finding an accurate and fast feature model for a given parsing model. On the one hand, we want a parser that should provide the best results possible, while on the other hand, we want a parser that should provide the results in the fastest way possible. For practical applications, a fast model is crucial.

In this paper, we report the results of feature selection experiments that we carried out with the intention of obtaining accurate and faster feature models, for the transition-based Mate parser with and without graph-based completion models. The Mate parser is a beam search parser that uses a hash kernel for training, joint part-of-speech tagging, morphological tagging and dependency parsing. As a result of this research, we provide a framework that allows to find an optimal feature template set for the Mate parser (Bohnet et al., 2013). Moreover, our models provide some of the highest results ever reported for a set of treebanks.

The paper is organized as follows. Section 2 describes related work including the used agenda-based dependency parser. This section depicts the feature templates that can be used by a transition-based or a graph-based parser. Section 3 describes the feature selection algorithm that we implemented for our experiments. Section 4 shows the experimental set-up. Section 5 reports the main results of our experiments. Section 6 provides the parsing times and memory requirements. Finally, Section 7 concludes.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 794–805, Dublin, Ireland, August 23-29 2014.

Transition		Condition
LEFT-ARC$_d$	$([\sigma\|i,j],B,\Gamma) \Rightarrow ([\sigma\|j],B,\Gamma[(j,i)\in A,\delta(j,i)=d])$	$i \neq 0$
RIGHT-ARC$_d$	$([\sigma\|i,j],B,\Gamma) \Rightarrow ([\sigma\|i],B,\Gamma[(i,j)\in A,\delta(i,j)=d])$	
SHIFT$_{p,m,l}$	$(\sigma,[i\|\beta],\Gamma) \Rightarrow ([\sigma\|i],\beta,\Gamma[\pi(i)=p,\mu(i)=m,\lambda(i)=l])$	
SWAP	$([\sigma\|i,j],\beta,\Gamma) \Rightarrow ([\sigma\|j],[i\|\beta],\Gamma)$	$0 < i < j$

Figure 1: Transition set for joint morphological and syntactic analysis. The stack Σ is represented as a list with its head to the right (and tail σ) and the buffer B as a list with its head to the left (and tail β).

2 Related Work

2.1 Mate Parser

For our experiments, we used the transition-based parser of Bohnet et al. (2013). This parser performs joint part-of-speech tagging, morphological tagging, and non-projective labeled dependency parsing. The parser employs a number of techniques that lead to very competitive accuracy such as beam-search with early update (Collins and Roark, 2004), a hash kernel that can quickly cope with a large feature set, a graph-based completion model that adds scores for tree parts which a transition-based parser would not be able to consider, cf. (Zhang and Clark, 2008; Bohnet and Kuhn, 2012). The graph-based model takes into account second and third order factors and obtains a score as soon as the tree parts are completed. The parser employs a rich feature set for a transition-based model (Zhang and Nivre, 2011; Bohnet et al., 2013) as well as for a graph-based model. In total, there are 326 different feature templates for the two models. The drawback of such a large feature set is a huge impact on the speed. Important research questions include (1) whether the number of features could be reduced to speed up the parser and (2) whether languages dependent feature sets would be beneficiary.

2.2 Features in transition-based dependency parsing

Every transition-based parser uses two data structures: (1) a buffer that contains at the beginning of the parsing process all words of the sentence that have to be parsed, and (2) a stack.

The Mate parser that we used in our experiment follows Nivre's arc-standard parsing algorithm plus the SWAP transition to build non-projective dependency trees. Figure 1 depicts the transition system formally; the SHIFT transition removes the first node from the buffer and puts it on the stack. The LEFT-ARC$_d$ transition introduces a labeled dependency edge between the top element on the stack and the second element of the stack with the label d. The second top element is removed from the stack. The RIGHT-ARC$_d$ transition introduces a labeled dependency edge between the second element on the stack and the top element with the label d while the top element is removed from the stack. The SWAP transition swaps the position of the topmost nodes of the stack and the buffer.

A classifier selects transitions based on the feature templates that are composed of stack elements, buffer elements, the already created parse, and the transition sequence. For instance, if the parser contains the feature template LEMMA(S_1), it means that it may use the lemma of the word that is in the first position of the stack in any parsing state in order to select the best parsing action.

2.3 Features in graph-based dependency parsing

A Graph-based dependency parser performs an exhaustive search over trees of the words of a sentence. Frequently, dynamic programming techniques are used to find the optimal tree for each span, considering candidate spans by successively building larger spans in a bottom-up fashion. A classifier is used to decide among alternative spans. The typical feature models are based on combinations of edges (as known as, factors). A factor consists either of a single edge, two or three edges; which are called first order, second and third order factors, respectively. The later are employed in more advanced and recent parsers trading off accuracy with complexity, cf. (McDonald et al., 2005b; Carreras, 2007; Koo and Collins, 2010). The features in a graph-based algorithm consist of sets of features drawn from the

vertexes involved in the factors. A feature template of a second order factor is composed of properties drawn from up to all three vertex, e.g., the part-of-speech of the head, the dependent and a child denoted as POS(H)+POS(D)+POS(C). In our experiments, we use in addition to the transition-based model, a completion model that uses graph-based feature templates with up to third order factors to re-score the beam.

2.4 Feature Selection

There has been some recent research on trying to manually find better feature models for dependency parsers, such as Nivre et al. (2006), Hall et al. (2007), Hall (2008), Zhang and Nivre (2011), and Agirre et al. (2011). There is also research on automatic feature selection in the case of transition-based dependency parsing, a good example is MaltOptimizer (Ballesteros and Nivre, 2014) which implements a search for the best feature model that it can find, following acquired previous experience and deep linguistic knowledge (Hall et al., 2007; Nivre and Hall, 2010); Nilsson and Nugues (2010) also tried to search for optimal feature sets in the case of transition-based parsing, starting from a reduced test set using the concept of topological neighbors. Finally, He He et al. (2013) also tried automatic feature selection but for a graph-based parsing algorithm, where they pruned the feature space, removing unused features, in a first-order graph-based dependency parser, providing models that are equally accurate and faster.

Zhang and Nivre (2011) pointed out that two different parsers based on the same algorithm may need different feature templates since other design aspects of a parser might have an influence on the usefulness of feature templates such as the learning technique or the use of beam search.

3 Feature Selection Algorithm

As in MaltOptimizer (Ballesteros and Nivre, 2014), our feature selection algorithm starts with a default feature set that is based on the MaltParser's default feature model for an arc-standard parsing algorithm[1], it first tests whether the features that are in the default model are actually useful, which means that whenever we remove any of the features of the default set, the accuracy is still the same (or better).

After that, one by one, the algorithm tries to add feature templates to the feature set. For each additional feature template a parser is trained for testing and if the accuracy is higher than the accuracy of the previous step plus a Δ (threshold) then the feature in question is added to the feature set. The selection process continues until all features have been tested, and therefore each feature has been either added or rejected. Most of the feature selection is based on the forward selection algorithm shown in Figure 2, although there is also a bit of backward selection from the default set.

Let $F = \{F_1, \ldots, F_n\}$ be the full set of features,
let $M(X)$ be the evaluation metric for feature set X,
and let Δ be the threshold.

```
1   X ← ∅
2   while X ≠ F
3       B ← 0
4       Y ← ∅
5       for each X_i ∈ F \ X
6           if M(X ∪ {X_i}) + Δ > B then
7               B ← M(X ∪ {X_i})
8               Y ← X ∪ {X_i}
9       if M(X) > B then
10          return X
11      else
12          X ← Y
13  return X
```

Figure 2: Algorithm for forward feature selection.

The feature selection algorithm only has the training set as an input, and it splits it into training and development to validate the outcomes of the experiments.[2] After the feature selection, we run the parser model on a held-out test set to measure its performance.

The feature selection is pruned following similar strategies to MaltOptimizer; there are features that are deeply related and the system tries to avoid unnecessary tests when some features happen to be excluded. For instance, the algorithm will not try to select the third position of the buffer for the part-of-speech, if the second position was excluded by the feature selection algorithm.

[1] http://www.maltparser.org/userguide.html

[2] It makes a 80/20 division; 80% for training, 20% for development.

4 Experimental Set-Up

In order to set up the experiments for the feature selection algorithm, we carried out a series of tests based on the parser settings. From these experiments, we obtained the best parser settings, the threshold that provides the best results given a development set, and the best scoring method and some additional configurations, that gave us reliable results and a fast outcome.

We used the following corpora for our experiments. **Chinese:** We used the Penn Chinese Treebank 5.1 (CTB5), converted with the head-finding rules and conversion tools of Zhang and Clark (2008), with the same split as in (Zhang and Clark, 2008) and (Li et al., 2011).[3] **English:** We used the WSJ section of the Penn Treebank, converted with the head-finding rules of Yamada and Matsumoto (2003) and the labeling rules of Nivre (2006).[4] **German:** We used Tiger Treebank (Brants et al., 2002) in the improved dependency conversion by Seeker and Kuhn (2012). **Hungarian:** We used the Szeged Dependency Treebank (Farkas et al., 2012). **Russian:** We used the SynTagRus Treebank (Boguslavsky et al., 2000; Boguslavsky et al., 2002).

4.1 Parser settings

As outlined in Section 3, our feature selection experiments require the training of a large number of parsing models and applying these to the development set.[5] Therefore, we aimed to find a training setup for the parser that provided fast training times while maintaining a realistic training and optimization scenario.

A major factor for the time usage is the beam size. The beam contains the alternative syntactic structures that are considered in the parsing process, and thus it requires more time and memory while it normally provides better results. The parser uses two additional small beams to store the differently tagged syntactic structures and morphological structures, for the joint models. We explored a number of configurations and assessed the parsing performance by carrying out a set of experiments on the Penn Treebank and the training settings of Bohnet et al. (2013);[6] the results are shown in Table 1.

	transition-based model								
beam	1	3	5	8	12	20	30	40	50
LAS	88.00	89.71	90.10	90.19	90.26	90.09	90.29	90.46	90.41
POS	96.88	97.02	97.03	97.00	96.94	96.95	97.02	96.92	97.00
TT	4	7	8	9	11	14	16	20	21
	transition-based and graph-based completion model								
beam	1	3	5	8	12	20	30	40	50
LAS	77.49	88.92	90.13	90.55	90.49	90.62	90.97	90.96	90.75
POS	96.71	96.93	96.97	96.97	96.97	97.05	96.99	97.00	97.04
TT	2	9	11	14	20	32	35	40	48

Table 1: Labeled Accuracy Score (LAS) in percent, Part-of-Speech tag accuracy POS in percent and training time (TT) in milliseconds per sentence. The parser was applied on the development set and trained over the Penn Treebank.

The table provides an overview of this preliminary experiment. The upper part of the table shows the performance when only using the transition-based model. The accuracy improvements are small when the beam-size becomes larger than 5. Even when we compared the results with the results of a beam size of 30, we observed only a small accuracy improvement. Further, we observe with a larger beam size a saturation where the accuracy does not improve and the parsing results show a small variance.

[3] Training: 001–815, 1001–1136. Development: 886–931, 1148–1151. Test: 816–885, 1137–1147.

[4] Training: 02-21. Development: 24. Test: 23.

[5] All this experiments were carried out on a CPU Intel Xeon 3.4 Ghz with 6 cores.

[6] We used 25 training iterations and we took the accuracy scores from the last iteration, we used the join parser, the two best part-of-speech tags and morphological tags. The threshold for the inclusion of part-of-speech tags was set to 0.25 and that of the morphological tagger to 0.1. We selected a beam size for the alternative POS tags and morphological tags of 4.

	English				German				
Δ	LAS	UAS	POS	#	LAS	UAS	POS	MOR	#
0.05	90.17	91.39	97.00	40	90.57	92.81	97.89	90.45	41
0.02	90.24	91.52	97.04	54	90.83	93.00	98.01	90.55	49
0.01	90.17	91.45	96.90	54	90.90	92.95	97.98	90.69	60
0.00	90.43	91.71	97.00	57	90.89	92.98	97.94	90.59	68
-0.01	90.26	91.47	97.06	69	90.92	93.09	98.02	90.72	79
-0.02	90.27	91.52	97.05	77	91.27	93.37	98.17	90.84	93
-0.05	90.49	91.66	97.01	98	91.02	93.11	98.11	90.69	116
$-\infty$	90.37	91.65	96.98	188	90.77	93.00	98.14	89.56	188

Figure 3: Accuracy scores depending on the threshold Δ.

The feature selection starts with a default feature set that includes 20 features (cf. Section 3), and all these features are derived from the default feature models for MaltParser (Nivre et al., 2007)[7]. In total, the feature selection algorithm, for the transition-based model, may select 188 features. In Table 1 we show the training time (TT). We used this table to selected the optimal settings for the beam. After considering the trade-off between accuracy and speed, we selected for the feature selection a beam size of 8, since it obtains 90.19 LAS which is close to the highest accuracy score 90.46 and with this beam size the parser is fast. For a parser trained with all feature templates, the average parsing time per sentence is 9 milliseconds. With 20-60 features, we obtained a parsing time of 2-5 milliseconds per sentence, which is a faster and more optimal setting for the feature selection. Moreover, with a beam size of 40, we get parsing times that ranged depending on the number of features from 12 to 50 milliseconds per sentence, this is impracticable for feature selection experiments.

4.2 Selecting an Optimal Threshold

Feature templates are selected when they provide a higher accuracy compared to the previous feature set plus a threshold Δ. To determine an optimal Δ for the feature selection, we carried out a series of experiments with different Δ values. As a first step, we ran the feature selection algorithm starting from 0.05 and reducing the value stepwise to -0.05 (testing 0.05, 0.02, 0.01, 0.0, -0.01, -0.02, -0.05) with the intention of obtaining accuracy scores for all these settings. Table 3 shows the scores for our experiments on the development set for the English and German treebanks. We obtained an optimal trade-off between score and number of features with a Δ of 0.0. With higher thresholds, such as 0.02 or 0.05, the feature selection algorithm was very restrictive, and resulted in lower accuracy scores. This indicates that there are several features that are not included that could contribute to a higher accuracy; for instance, in the German case, we see that the algorithm only selects 41 features. Moreover, the accuracy for English with a Δ of 0.0 is even higher compared with the results obtained when all features were included (cf. last row: $-\infty$). For German, we see a highest accuracy score with threshold of -0.02. We might get the best accuracy with this threshold when applied to the test set; however, the downside of this threshold is that the algorithm selected 25 more feature templates, which leads to a slower parser.

Figure 4 illustrates the accuracy gain depending on the number of features included. The development set of these graphs consist of 20% of the original training set. A negative Δ leads to the inclusion of more features, which seem to provide even slightly higher results while including much more features. This outcome is not fully supported by the results from the development sets for English where we observed slightly lower results for a Δ of -0.02 compared to 0.0.

To determine the optimal threshold Δ for a language would come with a high computational cost, we carried out these experiments for English and German which show only small differences in accuracy in the threshold range around 0. Therefore, we adopted 0.0 as threshold for our further experiments on other languages as well, cf. Table 4.

[7] http://maltparser.org

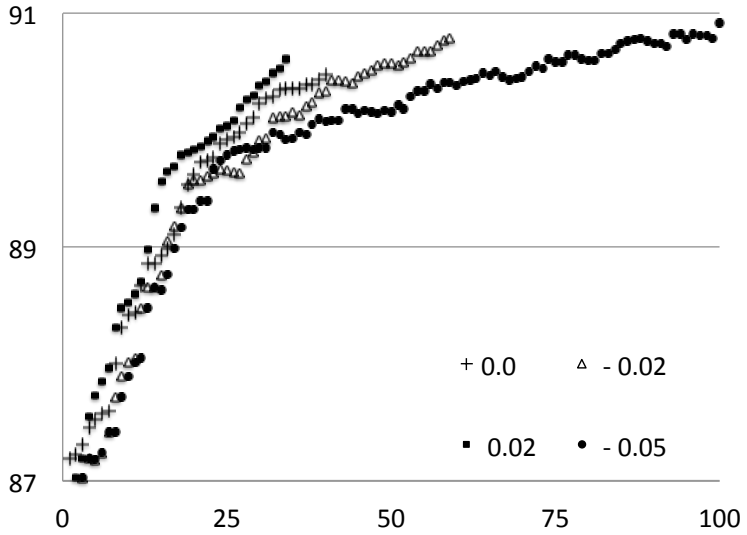


Figure 4: Selected features (x-axis) vs Labeled Accuracy Score (y-axis). Features: **transition-based**

	English				German				
	LAS	UAS	POS	#	LAS	UAS	POS	MOR	#
LAS	90.34	**91.71**	97.04	54	**90.89**	**92.98**	97.94	90.59	68
LUMP	**90.38**	91.57	**97.09**	55	90.82	92.88	**98.11**	90.65	53
PMLAS	90.12	91.38	97.02	40	89.27	91.66	98.01	**90.66**	31

Table 2: Experiments with evaluation metrics with a Δ of 0.0 on the development sets. Features: **transition-based**. The morphology results are only shown for German, because the English treebank does not contain separate morphological features.

4.3 Selecting the Best Scoring Method

We carried out a number of experiments to determine the best criterion for the inclusion of features into the model. We tested several evaluation measures that compute the results of each model, that are LAS [labeled attachment score], LUMP[8] [(labeled attachment score + unlabeled attachment score + morphology accuracy + part-of-speech accuracy)/4] and PMLAS[9] [labeled attachment score, morphology accuracy and part-of-speech accuracy]. Table 2 shows the results of the feature selection for English and German for all these scoring methods. We finally selected LAS as our scoring method given that it provides the best results for German and competitive results (at the same level) for English. LUMP is very similar, however, it seems a bit more restrictive than LAS. Moreover, PMLAS was the most restrictive measure, allowing only 31 features for German and 40 for English, which is the reason why there is a significant lower accuracy for the models selected with PMLAS.

Finally, it is worth mentioning that we explored an alternative criterion for the inclusion of features into the set. We explored the possibility to include only features that show a statistical significant improvement. However, this criterion is too strict as only very few features showed a statistical significant improvement on its own.

4.4 Selection of Feature Templates of the Graph-based Completion Model

The graph-based *completion model* re-scores the beam incrementally and leads to a higher accuracy. We tried to select the graph-based feature templates of the completion model after the selection of the

[8]LMP [(labeled attachment score + morphology accuracy + part-of-speech accuracy)/3] would have been another alternative. However, we wanted to give the syntax still a higher weight in the feature selection process.

[9]See (Bohnet et al., 2013)

transition-based feature templates. This approach could not reach the accuracy gain shown by Bohnet and Kuhn (2012). We attempted to compensate this by starting the selection procedure from the default set with the intention of maximizing potential accuracy gains. However, this procedure did not lead to a better accuracy when later combined with the selected transition-based feature templates. We tried also to relax the threshold to -0.02 in order to include more features and to achieve a higher accuracy. Since this leads to better results, we performed the feature selection for the graph-based completion model with this setting.

4.5 Morphology for English

The Penn Treebank is annotated with part-of-speech tags that include morphological features such as `NNS` (plural noun) or `VBD` (verb past tense). The corpus does not include separate morphological features. Splitting up these features could be useful because: (1) the parser might be able to generalize better when we use the word categories separated from morphological features, and (2) we might take advantage of the ability of the parser to predict morphology and part-of-speech based on the interaction with the syntax. Table 3 summarizes the results. Our transition-based parsing model shows only small differences between the scores for the original POS tag set and the tag set that separates the category and morphology.

	transition-based model				
	LAS	UAS	POS	MOR	POS&MOR
baseline dev	90.13	91.44	–	–	96.97
separate dev	90.11	91.26	97.66	98.81	97.08
baseline test	92.11	93.16	–	–	97.41
separate test	92.07	93.09	**97.88**	**97.93**	**97.35**
	transition-based model with completion model				
baseline test	92.41	93.35	–	–	97.41
separate test	**92.53**	**93.49**	97.85	98.89	97.28

Table 3: Experiments on Penn Treebank with separate representation of word category and morphology.

The results of the transition-based model, including the graph-based model shows some larger differences

The labeled and unlabeled accuracy scores are not statistically significant and we concluded that (1) and (2) do not probably hold. Splitting up the morphology is a neutral operation in terms of labeled and unlabeled accuracy scores; however, it is worth noting that our results with the separate test for the completion model is more competitive, providing an improvement of 0.14 UAS.

5 Experiments: Feature Selection

We applied the feature selection algorithm with the parameters determined in the previous sections on the corpora of Chinese, English, German, Hungarian and Russian, and we applied the outcome to parse the held-out test sets with a beam size of 40 and 25 training iterations. Table 4 shows the accuracy scores and the number of features selected for each language. The threshold for inclusion of the feature was set to 0, cf. section 4.

The first row (Full) shows the accuracy scores for the full set of features, that includes all 188 feature templates of the transition-based feature set. The second row gives the accuracy scores that have been obtained with the reduced feature set gained by the feature selection algorithm described in Section 3.

For the sole transition-based parsers trained with the selected features, we obtain for Chinese, Hungarian and Russian higher labeled and unlabeled accuracy scores. The scores for German are very similar to the ones obtained with the full set and the scores for English are slightly worse. In the case of the transition-based parser with graph-based completion model, the results are the same for Chinese, and slightly worse for the rest of the languages, with the parser at least twice as fast. It is worth noting that the number of feature templates is reduced by 2/3 across all languages which leads to a much faster parsing and training time, thus freeing up a huge amount of main memory.

	German					Hungarian					Russian				
	LAS	UAS	POS	MOR	#	LAS	UAS	POS	MOR	#	LAS	UAS	POS	MOR	#
Transition-based features															
Full	91.39	93.39	97.96	90.36	188	87.67	90.38	97.83	96.39	188	86.73	92.24	98.88	94.66	188
Select	91.34	93.36	97.88	90.48	68	87.94	90.51	97.87	96.38	71	87.21	92.40	98.88	94.74	64
Transition-based and graph-based features															
Full+Cmp	91.77	93.63	98.14	90.77	326	88.88	91.33	97.84	96.41	326	87.66	92.84	98.82	94.56	326
Sel+Cmp	91.81	93.72	97.85	90.44	206	88.67	91.16	97.83	96.39	209	87.93	93.01	98.89	94.73	202
Sel+Sel	91.60	93.61	97.85	90.39	91	88.40	90.50	97.86	96.39	97	87.57	92.76	98.88	94.59	75

	Chinese				English			
	LAS	UAS	POS	#	LAS	UAS	POS	#
Transition-based features								
Full	77.81	81.13	94.11	188	92.13	93.18	97.40	188
Select	78.04	81.20	94.17	56	91.89	92.93	97.38	57
Transition-based and graph-based features								
Full+Cmp	78.34	81.46	94.19	326	92.41	93.35	97.41	326
Sel+Cmp	78.74	81.86	94.13	197	92.22	93.19	97.37	195
Sel+Sel	78.74	81.77	94.28	67	92.08	93.05	97.44	74

Table 4: Labeled attachment score (LAS), unlabeled attachment score (UAS), part-of-speech accuracy (POS) and morphology accuracy (MOR) per language and model. The first two rows refer only to transition-based features while the last two rows include transition-based and graph-based features. **Full** refers to a model with all transition-based features. **Select** refers to a model with selected transition-based features. **Full+Cmp** refers to a model with all transition-based features and all graph-based features. **Sel+Cmp** refers to a model with selected transition-based features and all graph-based features. **Sel+Sel** refers to a model with selected transition-based features and selected graph-based features. The English and Chinese accuracy scores exclude punctuation marks.

More about parsing time, training time and memory requirements is depicted in Section 6. A comparison with state of the art results as shown in the Tables 5a to 5d reveal that the parser with the selected features of the transition-based, and graph-based model are on an equal level for Chinese, Russian and Hungarian with state-of-the-art results. With the selected transition-based and the full graph-based feature templates, the results for these languages surpass current state-of-the-art results.

6 Time and Memory Requirements

The number of feature templates has a serious impact on training time, parsing time and the amount of main memory required. The feature selection may have huge impact on the speed of a parser. Therefore, we measure the actual time and memory usage by applying the parser on the English test set of the Penn Treebank. This was done with different parsing models, and for each model, test runs were performed with an increasing number of CPU cores. Figure 6 shows an overview of the results.

The parsing model with all transition- and graph-based features takes on one CPU core 0.085 seconds per sentence (cf. Figure 6, line with rhombus). In contrast, the parser with selected transition-based features parses a sentence in less than half of the time (0.042 seconds, line with crosses). The parsing accuracy is only 0.42 percentage points worse (93.35 vs. 92.93 UAS). When we compare the first parsing model with the model with selected transition-based and graph-based features, we observe a parsing time of 0.066 seconds per sentence and a small accuracy difference of only 0.27.

If we use six CPU cores then parsing time decreases drastically to 0.016 seconds per sentence for the selected transition-based feature model, 0.023 for the selected transition- and graph-based feature model and to 0.05 seconds per sentence for the model with all features (which is much slower). Our experiments

Parser	UAS	LAS	POS
McDonald et al. (2005a)	90.9		
McDonald and Pereira (2006)	91.5		
Huang and Sagae (2010)	92.1		
Koo and Collins (2010)	93.04		
Zhang and Nivre (2011)	92.9		
Martins et al. (2010)	93.26		
Bohnet and Nivre (2012)	93.38	92.44	97.33
this work (sel. trans.& sel. cmpl.)	93.05	92.08	**97.44**
this work (P&M cf. Table 3)	**93.49**	**92.53**	–
Koo et al. (2008) †	93.16		
Carreras et al. (2008) †	93.5		
Suzuki et al. (2009) †	93.79		

(a) Accuracy scores for WSJ-PTB. Results marked with † use additional information sources and are not directly comparable to the others.

Parser	UAS	POS
MSTParser1	75.56	93.51
MSTParser2	77.73	93.51
Li et al. (2011) 3rd-order	80.60	92.80
Hatori et al. (2011) HS	79.60	94.01
Hatori et al. (2011) ZN	81.20	93.94
this work (sel. trans.)	81.20	94.17
this work (sel. trans.+ sel. cmp.)	**81.77**	**94.28**

(b) Accuracy scores for the Chinese treebank converted with the head rules of Zhang and Clark (2008). MSTParser results from Li et al. (2011). UAS scores from Li et al. (2011) and Hatori et al. (2011) recalculated from the separate accuracy scores for root words and non-root words.

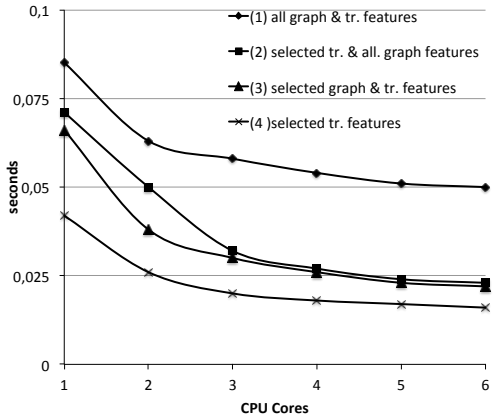
Parser	UAS	LAS	POS
Farkas et al. (2012)	90.1	87.2	
Bohnet et al. (2013)	**91.3**	**88.9**	**98.1**
this work (sel. trans. & sel. cmpl.)	90.50	88.40	97.83
this work (sel. trans. & full cmpl.)	91.16	88.67	97.86

(c) State of the art comparison for Hungarian. The table shows that we can reach state of the art performance with less features.

Parser	UAS	LAS	POS
Boguslavsky et al. (2011)	90.0	86.0	
Bohnet et al. (2013)	92.8	87.6	98.5
this work (sel. trans. & sel. cmp.)	92.76	87.57	**98.89**
this work (sel. trans. & full cmp.)	**93.01**	**87.93**	98.88

(d) State of the art comparison for Russian.

Figure 5: Comparison with state of the art results.



id	trans. features	graph features	# features	UAS
(1)	all	all	75.8 M	93.35
(2)	selected	all	43.5 M	93.22
(3)	selected	selected	22.1 M	93.08
(4)	selected	none	17.8 M	92.93

Figure 6: Parsing Time in relation to CPU cores and number of features in the hash kernel in millions.

demonstrate that we can double the parsing speed and maintain a very high parsing accuracy.

7 Conclusions

In this paper, we have presented the first feature selection algorithm for agenda-based dependency parsing. Our algorithm could be directly used out of the box,[10] and applied to a new data set or language to get an optimized feature model for a agenda-based parser such as the Mate tools.[11]

Our feature selection algorithm provides models with even higher accuracy for Chinese and Russian, cf.Table 4. For the remaining languages the models provide accuracy scores that are comparable to the ones obtained by models including a larger set of feature templates. For all languages, the feature models gained via feature selection are faster and require less memory, which make them very useful for practical applications. We conclude that feature models obtained with the feature selection algorithm

[10]The source code and the feature models found for each language are available at `https://code.google.com/p/mate-tools/`

[11]`https://code.google.com/p/mate-tools/wiki/ParserAndModels`

often provide a comparable accuracy level while they are considerable faster. Finally, our model for English with the separated morphology tag-set provides one of the best results reported with **93.49** UAS. Additionally, the feature selection algorithms for this setting shows competitive results with a largely reduced number of feature templates, and thus less parsing time and lower memory requirements. The parser is faster (almost double) and provides **93.05** UAS which is also among the best results.

Acknowledgments

This research project was supported by funding of the European Union (PCIG13-GA-2013-618143).

References

Eneko Agirre, Kepa Bengoetxea, Koldo Gojenola, and Joakim Nivre. 2011. Improving Dependency Parsing with Semantic Classes. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*, pages 699–703, Portland, USA.

Miguel Ballesteros and Joakim Nivre. 2014. MaltOptimizer: Fast and Effective Parser Optimization. *Natural Language Engineering*.

Miguel Ballesteros. 2013. Effective morphological feature selection with MaltOptimizer at the SPMRL 2013 shared task. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 53–60.

Igor Boguslavsky, Svetlana Grigorieva, Nikolai Grigoriev, Leonid Kreidlin, and Nadezhda Frid. 2000. Dependency treebank for Russian: Concept, tools, types of information. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING)*, pages 987–991.

Igor Boguslavsky, Ivan Chardin, Svetlana Grigorieva, Nikolai Grigoriev, Leonid Iomdin, Leonid Kreidlin, and Nadezhda Frid. 2002. Development of a dependency treebank for Russian and its possible applications in NLP. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC)*, pages 852–856.

Igor Boguslavsky, Leonid Iomdin, Victor Sizov, Leonid Tsinman, and Vadim Petrochenkov. 2011. Rule-based dependency parser refined by empirical and corpus statistics. In *Proceedings of the International Conference on Dependency Linguistics*, pages 318–327.

Bernd Bohnet and Jonas Kuhn. 2012. The best of bothworlds - a graph-based completion model for transition-based parsers. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 77–87.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465, Jeju Island, Korea, July. Association for Computational Linguistics.

Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajic. 2013. Joint morphological and syntactic analysis for richly inflected languages. *Transactions of the Association for Computational Linguistics(TACL)*, 1:415–428.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. TIGER treebank. In *Proceedings of the 1st Workshop on Treebanks and Linguistic Theories (TLT)*, pages 24–42.

Xavier Carreras, Michael Collins, and Terry Koo. 2008. Tag, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL)*, pages 9–16.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the CoNLL Shared Task at the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CONLL)*, pages 957–961.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 112–119.

Richárd Farkas, Veronika Vincze, and Helmut Schmid. 2012. Dependency parsing of hungarian: Baseline results and challenges. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 55–65.

Johan Hall, Jens Nilsson, Joakim Nivre, Gülsen Eryiğit, Beáta Megyesi, Mattias Nilsson, and Markus Saers. 2007. Single Malt or Blended? A Study in Multilingual Parser Optimization. In *Proceedings of the CoNLL Shared Task at the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CONLL)*, pages 933–939.

Johan Hall. 2008. *Transition-Based Natural Language Parsing with Dependency and Constituency Representations.* Ph.D. thesis, Växjö University.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental joint pos tagging and dependency parsing in chinese. pages 1216–1224.

He He, Hal Daumé III, and Jason Eisner. 2013. Dynamic feature selection for dependency parsing. In *EMNLP*, pages 1455–1464.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1077–1086.

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1–11.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 595–603.

Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011. Joint models for chinese POS tagging and dependency parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1180–1191.

Andre Martins, Noah Smith, Eric Xing, Pedro Aguiar, and Mario Figueiredo. 2010. Turbo parsers: Dependency parsing by approximate variational inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 34–44.

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

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 91–98.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 523–530.

Peter Nilsson and Pierre Nugues. 2010. Automatic Discovery of Feature Sets for Dependency Parsing. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pages 824–832.

Joakim Nivre and Johan Hall. 2010. A quick guide to MaltParser optimization. Technical report, maltparser.org.

Joakim Nivre, Johan Hall, Jens Nilsson, Gülsen Eryiğit, and Svetoslav Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 221–225.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13:95–135.

Joakim Nivre. 2006. *Inductive Dependency Parsing*. Springer.

Wolfgang Seeker and Jonas Kuhn. 2012. Making ellipses explicit in dependency conversion for a german treebank. pages 3132–3139.

Noah A. Smith. 2011. *Linguistic Structure Prediction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool, May.

Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 551–560.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 195–206.

Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 562–571.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 188–193, Portland, Oregon, USA.

Predicate-Argument Structure Analysis with Zero-Anaphora Resolution for Dialogue Systems

Kenji Imamura, Ryuichiro Higashinaka, and **Tomoko Izumi**
NTT Media Intelligence Laboratories, NTT Corporation
1-1 Hikari-no-oka, Yokosuka, 239-0847, Japan
{imamura.kenji,higashinaka.ryuchiro,izumi.tomoko}@lab.ntt.co.jp

Abstract

This paper presents predicate-argument structure analysis (PASA) for dialogue systems in Japanese. Conventional PASA and semantic role labeling have been applied to newspaper articles. Because pronominalization and ellipses frequently appear in dialogues, we base our PASA on a strategy that simultaneously resolves zero-anaphora and adapt it to dialogues. By incorporating parameter adaptation and automatically acquiring knowledge from large text corpora, we achieve a PASA specialized to dialogues that has higher accuracy than that for newspaper articles.

1 Introduction

Semantic role labeling (SRL) and predicate-argument structure analysis (PASA) are important analysis techniques for acquiring "who did what to whom" from sentences[1]. These analyses have been applied to written texts because most annotated corpora comprise newspaper articles (Carreras and Màrquez, 2004; Carreras and Màrquez, 2005; Matsubayashi et al., 2014).

Recently, systems for speech dialogue between humans and computers (e.g., Siri of Apple Inc. and Shabette Concier of NTT DoCoMo) have become familiar with the popularization of smart phones. A man-machine dialogue system has to interpret human utterances to associate them with system utterances. The predicate-argument structure could be an effective data structure for dialogue management. However, it is unclear whether we can apply the SRL/PASA for newspaper articles to dialogues because there are many differences between them, such as the number of speakers, written or spoken language, and context processing. For example, the following dialogue naturally includes pronouns, and thus anaphora resolution is necessary for semantic role labeling.

A:	$[I]_{ARG0}$ ***want*** $[an\ iPad\ Air]_{ARG1}$.
B:	$[When]_{ARGM}$ *will* $[you]_{ARG0}$ ***buy*** $[it$(=an iPad Air)$]_{ARG1}$?

Similar phenomena exist in Japanese dialogues. However, most pronouns are omitted (called zero-pronouns), and zero-anaphora resolution is necessary for Japanese PASA.

A:	$[iPad\ Air]_{NOM}$	*-ga*	***hoshii****-na.*
	iPad Air	NOM.	want
	"ϕ want an iPad Air."		
B:	*itsu*	$\phi_{NOM}\phi_{ACC}$	***kau****-no?*
	when		buy?
	"When will ϕ buy ϕ?"		

[1]Recent SRL systems assign labels of predicates and their arguments as semantic roles. Consequently, SRL and PASA are very similar tasks. We use the term predicate-argument structure analysis in this paper because most Japanese analyzers use this term.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 806–815, Dublin, Ireland, August 23-29 2014.

This paper presents predicate-argument structure analysis with zero-anaphora resolution for Japanese chat dialogues. Here, we regard the task of constructing PASA for dialogues as a kind of domain adaptation from newspaper articles to dialogues. Màrquez et al. (2008) and Pradhan et al. (2008) indicated that the tuning of parameter distribution and reducing the out-of-vocabulary are important for the domain adaptation of SRL. We also focus on parameter distribution and out-of-vocabulary to construct a PASA adapted to dialogues. To the best of our knowledge, this is the first paper to describe a PASA for dialogues that include many zero-pronouns.

The paper is organized as follows. Section 2 briefly reviews SRL/PASA in English and Japanese. Section 3 discusses characteristics of chat dialogues by comparing two annotated corpora, newspaper articles and dialogues. Section 4 describes the basic strategy of our PASA, and Section 5 shows how it was adapted for dialogues. Experiments are presented in Section 6, and Section 7 concludes the paper.

2 Related Work

2.1 Semantic Role Labeling in English

The advent of the supervised method proposed by Gildea and Jurafsky (2002) has led to the creation of annotated corpora for semantic role labeling. In the CoNLL-2004 and 2005 shared task (Carreras and Màrquez, 2004; Carreras and Màrquez, 2005), evaluations were carried out using the Proposition Bank (Palmer et al., 2005). Because the Proposition Bank was annotated to the Penn Treebank (i.e., the source texts were from the Wall Street Journal), the shared tasks were evaluated on newspaper articles. Màrquez et al. (2008) provides a review of SRL.

OntoNotes Corpus (Hovy et al., 2006) contains multiple genres such as newswire, broadcast news, broadcast conversation. The annotation to OntoNotes includes semantic role labels compliant with the Proposition Bank. It is currently used for coreference resolution (Pradhan et al., 2012), and is expected to be applied to dialogue analysis.

A few SRL studies have focused on not only verbal predicates (e.g., '*decide*') but also nominal predicates (e.g., '*decision*') (Jiang and Ng, 2006; Gerber and Chai, 2012; Laparra and Rigau, 2013). Because the subject and object of nominal predicates are frequently omitted (e.g., the object in the phrase "*the decision*" is omitted), problems similar to the Japanese zero-pronouns have to be resolved in the SRL of nominal predicates.

2.2 Predicate-Argument Structure Analyses in Japanese

Japanese material includes the NAIST Text Corpus (Iida et al., 2007)[2], which is an annotated corpus of predicate-argument structures and coreference information for newspaper articles. Argument noun phrases of the nominative, accusative, and dative cases are assigned to each predicate. The predicate and the noun phrases are not limited to the same sentence. If arguments of the predicate are represented as zero-pronouns, the antecedent noun phrases in other sentences are assigned as the arguments.

Many PASA methods have been studied on the NAIST Text Corpus (Komachi et al., 2007; Taira et al., 2008; Imamura et al., 2009; Yoshikawa et al., 2011). In Japanese, some of them simultaneously resolve the zero-anaphora caused by zero-pronouns.

Most English SRL and Japanese PASA currently target newspaper articles, and it is unclear whether the methods for newspapers can be applied to dialogue conversations.

3 Characteristics of Chat Dialogues

We first collected chat dialogues of two speakers and annotated them with the predicate-argument structure. The participants chatted via keyboard input. Therefore, fillers and repetitions, which are frequent in speech dialogues, were rare. The theme was one of 20 topics, such as meals, travel, hobbies, and TV/radio programs. Annotation of the predicate-argument structure complied with the NAIST Text Corpus. Figure 1 shows a chat dialogue example and its predicate-argument structure annotation.

[2] `http://cl.naist.jp/nldata/corpus/`. We use version 1.5 with our own preprocessing in this paper. NAIST is an acronym of "Nara Institute of Science and Technology."

A: *natsu-wa* (exo2)$_{NOM}$ (exog)$_{DAT}$ ***dekake**-tari-shimashi-ta-ka?*
"Did (you)$_{NOM}$ **go** (anywhere)$_{DAT}$ in this summer?"

B: *8-gatsu-wa Ito-no* [*hanabi-taikai*]$_{DAT}$*-ni* (exo1)$_{NOM}$ ***yuki**-mashi-ta.*
"(I)$_{NOM}$ **went** to [the fireworks^{*1}]$_{DAT}$ at Ito in August."

A: [*hanabi*^{*2}]$_{ACC}$, [*watashi*^{*3}]$_{NOM}$*-mo* ***mi**-takatta-desu.*
"[Fireworks^{*2}]$_{ACC}$, [I^{*3}]$_{NOM}$ also wanted to **see** (it)."

A: *demo, kotoshi-wa* (exo1)$_{NOM}$ ***isogashiku**-te* (exo1)$_{NOM}$ ($\star$2)$_{ACC}$ ***mi**-ni* ($\star$2)$_{DAT}$ ***ike**-masen-deshita.*
"But (I)$_{NOM}$ couldn't **go** ($*2$)$_{DAT}$ to **see** (it=*2)$_{ACC}$ this year because (I)$_{NOM}$ **was busy**."

Figure 1: Chat Dialogue Example and Its Predicate-Argument Structure Annotation
Lower lines denote glosses of the upper lines. The bold words denote predicates, the square brackets [] denote intra-sentential arguments, and the round brackets () denote inter-sentential or exophoric arguments.

Corpus	Set	# of Articles /Dialogues	# of Sentences /Utterances	# of Words (per Sentence)		# of Predicates (per Sentence)	
NAIST Text Corpus	Training	1,751	24,283	664,898	(27.4)	68,602	(2.83)
	Development	480	4,833	136,585	(28.3)	13,852	(2.87)
	Test	696	9,284	255,624	(27.5)	26,309	(2.83)
Chat Dialog Corpus	Training	184	6,960	61,872	(8.9)	7,470	(1.07)
	Test	101	4,056	38,099	(9.4)	5,333	(1.31)

Table 1: Sizes of Corpora

Case	Corpus	# of Arguments	Dep	Zero-Intra	Zero-Inter	Exophora exo1	exo2	exog
Nominative	NAIST	68,598	54.5%	17.3%	11.4%	2.0%	0.0%	14.7%
	Dialogue	7,467	31.8%	7.4%	12.6%	23.9%	5.6%	18.8%
Accusative	NAIST	27,986	89.2%	6.9%	3.4%	0.0%	0.0%	0.4%
	Dialogue	1,901	46.6%	12.8%	27.5%	0.8%	0.1%	12.2%
Datative	NAIST	6,893	84.7%	10.2%	4.3%	0.0%	0.0%	0.8%
	Dialogue	2,089	37.6%	7.8%	15.0%	2.5%	1.1%	36.1%

Table 2: Distribution of Arguments in Training Corpora

Table 1 shows the statistics of the NAIST Text Corpus and the Chat Dialogue Corpus we created[3]. The size of the Dialogue Corpus is about 10% of the NAIST Corpus. The NAIST Corpus is divided into three parts: training, development, and test. The Dialogue Corpus is divided into training and test.

Table 2 shows distributions of arguments in the training sets of the NAIST/Dialogue corpora. We classified the arguments into the following six categories because each argument presents different difficulties for analysis by its position and syntactic relation. The first two categories (Dep and Zero-Intra) are the ones that in which the predicate and the argument occupy the same sentence.

- Dep: The argument directly depends on the predicate and vice versa on the parse tree.
- Zero-Intra: Intra-sentential zero-pronoun. The predicate and the argument are in the same sentence, but there is no direct dependency.
- Zero-Inter: Inter-sentential zero-pronoun. The predicate and the argument are in different sentences.
- exo1/exo2/exog: These are exophoric and denote zero-pronouns of the first person, second person, and the others (general), respectively.

By Table 2, we can see that the ratios of Dep in all cases decreased in the Dialogue Corpus. In the other categories, the tendencies between the nominative case and the accusative/dative cases were different. In the nominative case, the Zero-Intra also decreased in the Dialogue Corpus, and the declines were

[3]We regard a dialogue and an utterance as an article and a sentence, respectively.

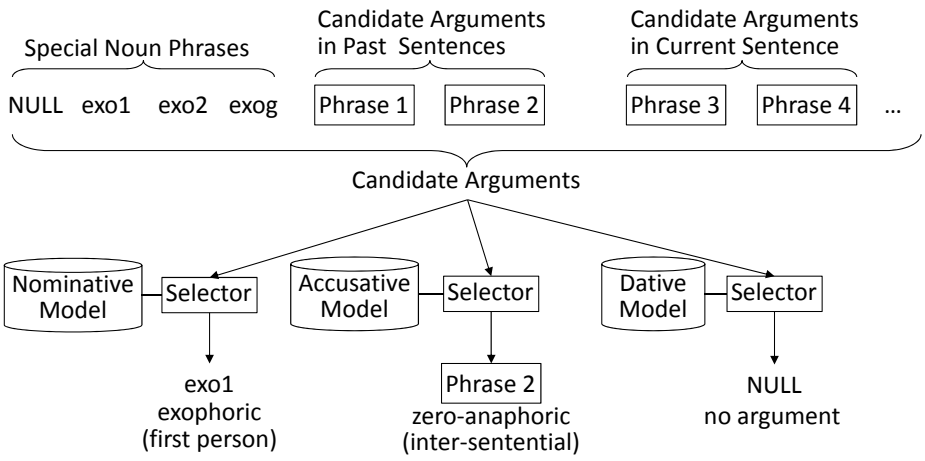


Figure 2: Structure of Argument Identification and Classification

assigned to `exo1` and `exo2`. Namely, the arguments in a sentence were reduced, and zero-pronouns increased compared with the newspaper articles. Note that many antecedents were the first or second person. On the other hand, in the accusative and dative cases, the declines of the `Dep` were assigned to the `Zero-Inter` or the `exog` in the Dialogue Corpus. Namely, anaphora resolution across multiple sentences is important to dialogue analysis. In contrast, most arguments and the predicate appear in the same sentence in the accusative/dative cases of newspapers.

4 Basic Strategy for Predicate-Argument Structure Analysis and Zero-Anaphora Resolution

4.1 Architecture

We use Imamura et al. (2009)'s method developed for newspaper articles as the base PASA in this paper. It can simultaneously identify arguments of a predicate in the sentence, those in other sentences, and exophoric arguments. The analyzer receives the entire article (dialogue) and performs the following steps for each sentence (utterance).

1. The input sentences are tagged and parsed. During parsing, the base phrases and their headwords are also identified. At this time, the part-of-speech tags and the parse trees of the Dialogue Corpus are supplied by applying the morphological analyzer MeCab (Kudo et al., 2004) and the dependency parser CaboCha (Kudo and Matsumoto, 2002). The NAIST Corpus version 1.5 already includes the part-of-speech tags and the parse trees.

2. Predicate phrases are identified from the sentences. We use the correct predicates in the corpora for the evaluation. When we build dialogue systems on PASA, predicate phrases will be identified using part-of-speech patterns that include verbs, adjectives, and copular verbs.

3. For each predicate, candidate arguments are acquired from the sentence that includes the predicate (called the current sentence) and the past sentences. Concretely, the following base phrases are regarded as candidates.

   - All noun phrases in the current sentence are extracted as intra-sentential candidates regardless of syntactic relations.
   - From the past sentences, noun phrases are contextually extracted as inter-sentential candidates. Details are described in Section 4.4.
   - Exophoric labels (`exo1`, `exo2`, and `exog`) and the `NULL` (the argument is not required) are added as special noun phrases.

4. The features are generated from the predicate phrase, candidate arguments, and their relations. The best candidate for each case is independently selected (Figure 2).

4.2 Models

The models for the selector are based on maximum entropy classification. The selector identifies the best noun phrase $\hat{n}$ that satisfies the following equations from the candidate argument set **N**.

$$\hat{n} = \operatorname*{argmax}_{n_j \in \mathbf{N}} P(d(n_j) = 1 | X_j; M_c) \quad (1)$$

$$P(d(n_j) = 1 | X_j; M_c) = \frac{1}{Z_c(X)} \exp \sum_k \{\lambda_{ck} f_k(d(n_j) = 1, X_j)\} \quad (2)$$

$$Z_c(X) = \sum_{n_j \in \mathbf{N}} \exp \sum_k \{\lambda_{ck} f_k(d(n_j) = 1, X_j)\} \quad (3)$$

$$X_j = \langle n_j, v, A \rangle \quad (4)$$

where n denotes a candidate argument, **N** denotes a set of candidate arguments of predicate v, $d(n)$ is a function that returns 1 iff candidate n becomes the argument, and M_c denotes the model of case c. In addition, $f_k(d(n_j) = 1, X_j)$ is a feature function, λ_{ck} denotes a weight parameter of the feature function, and A denotes the article from which all sentences are parsed.

Training phase optimizes the weight parameters in order to maximize the difference in posterior probabilities among the correct noun phrase and the other candidates. Specifically, the model of case M_c is learnt by minimizing the following loss function ℓ_c.

$$\ell_c = -\sum_i \log P(d(n_i) = 1 | X_i; M_c) + \frac{1}{2C} \sum_k ||\lambda_{ck}||^2 \quad (5)$$

where n_i denotes the correct noun phrase of the i-th predicate in the training set, X_i denotes the i-th tuple of the correct noun phrase, the predicate, and the article $\langle n_i, v_i, A_i \rangle$. Since the posterior probability is normalized for each set of candidate arguments of a predicate by Equation (3), the probability of the correct noun phrase approaches closer to 1.0, and the probabilities of the other candidates approach closer to 0.0 in Equation (5).

4.3 Features

Similar to other studies (e.g., (Gildea and Jurafsky, 2002)), we use three types of features: 1) predicate features, 2) noun phrase (NP) features, and 3) the relationship between predicates and noun phrases (Table 3). We also introduce combined features of the 'Noun' with all other binary features because the features aim to select the best noun phrase.

The special features in this paper are the dependency language models (three types) and the obligatory case information ('Frame' feature), which are automatically acquired from large text corpora. We discuss them in Section 5.2.

4.4 Context Processing

Contexts of dialogues and newspaper articles are different. We should employ context processing specialized for the dialogues. However, contexts, including system and user utterances, should be managed collectively by the dialogue manager from the viewpoint of dialogue systems. Thus, this study uses the same context processing for the newspaper articles and dialogues. Note that the method in this paper controls the context by selecting the inter-sentential candidates. We can easily alter context management by providing candidate arguments from an external manager.

Context processing in this paper is as follows.

- From the current sentence, trace back to the past, and find a sentence that contains the other predicate (we call this the prior sentence). This process aims to ignore utterances that do not contain predicates.

Type	Name	Value	Remark
Predicate	Pred	Binary	Lemma of the predicate.
	PType	Binary	Type of predicate. One of 'verb', 'adjective', and 'copular verb'.
	Voice	Binary	Declarative or not. If not, the passive/causative auxiliary verb is assigned.
	Suffix	Binary	Sequence of the functional words of the main clause. This feature aims to reflect the speech act of the utterance.
	Frame	Binary	Obligatory case information. The case requires argument (1) or not (0).
Noun Phrase	Noun	Binary	Headword of the NP
	Particle	Binary	Case particle of the base phrase. If the NP is a special noun phrase, this is `NULL`.
	NType	Binary	If the substance of the NP is in the article, this is 'NP'; otherwise the same value of the 'Noun' feature.
	Surround	Binary	POS tags of the surrounding words of the NP. The window size is ± 2.
Relation between Predicate and NP	PhPosit	Binary	Distance between the predicate and the NP. If they are in different sentences, or the NP is an exophora, this is `NULL`.
	Syn	Binary	Dependency path between the predicate and the NP. If they are in different sentences, or the NP is an exophora, this is `NULL`.
	Speaker	Binary	Whether the speakers of the predicate and the NP are the same (SAME) or not (OTHER).
Dependency Language Models	$\log P(n\|c, v)$	Real	Generation probability of NP n given predicate v and case c.
	$\log P(v\|c, n)$	Real	Generation probability of predicate v given NP n and case c.
	$\log P(c\|n)$	Real	Generation probability of case c given NP n.

Table 3: List of Features

- All noun phrases that lie between the prior to the current sentence are added to the candidate arguments. In addition, noun phrases that are used as arguments of any predicates are also added (called argument recycling (Imamura et al., 2009)). Argument recycling covers wide contexts because it can employ distant noun phrases if the past predicates have inter-sentential arguments.

5 Adaptation to Chat Dialogues

The method described in the previous section is common to dialogues and newspaper articles. This section describes the adaptation made to target dialogues.

5.1 Adaptation of Model Parameters

In order to tune the difference in the argument distribution, model parameters of the selectors are adapted to the dialogue domain. We use the feature augmentation method (Daumé, 2007) as the domain adaptation technique; it has the same effect as regarding the source domain to be prior knowledge, and the parameters are optimized to the target domain. Concretely, the models of the selectors are learnt and applied as follows.

1. First, the feature space is segmented into three parts: common, source, and target.
2. The NAIST Corpus and the Dialogue Corpus are regarded as the source and the target domains, respectively. The features from the NAIST Corpus are deployed to the common and the source spaces, and those from the Dialogue Corpus are deployed to the common and the target spaces.
3. The parameters are estimated in the usual way on the above feature space. The weights of the common features are emphasized if the features are consistent between the source and target. With regard to domain-dependent features, the weights in the respective space, source or target, are emphasized.
4. When the argument is identified, the selectors use only the features in the common and target spaces. The parameters in the spaces are optimized to the target domain, plus we can utilize the features that appear only in the source domain data.

5.2 Weak Knowledge Acquisition from Very Large Resources

In this paper, we use two types of knowledge to reduce the harmful effect of out-of-vocabulary in the training corpus. Both types are constructed by automatically analyzing, summing up, and filtering large

text corpora (Kawahara and Kurohashi, 2002; Sasano et al., 2008; Sasano et al., 2013). They provide information about unknown words with some confidence but they do contain some errors. We use them as the features of the models, and parameters are optimized by the discriminative learning of the selectors.

5.2.1 Obligatory Case Information (Frame Feature)

Case frames are important clues for SRL and PASA. The obligatory case information (OCI) comprises subsets of the case frames that only clarify whether the cases of each predicate are necessary or not.

The OCI dictionary is automatically constructed from large text corpora as follows. The process assumes that 1) most of the cases match the case markers if the noun phrase directly depends on the predicate, and 2) if the case is obligatory, the occurrence rate on a specific predicate is higher than the average rate of all predicates.

1. Similar to PASA in this paper (c.f., Section 4.1), predicates and base phrases are identified by tagging and parsing raw texts.
2. Noun phrases that directly depend on the predicate and accompany a case marker are extracted. We sum up the frequency of the predicate and cases.
3. Highly frequent predicates are selected according to the final dictionary size. Obligation of the cases is determined so as to satisfy the following two conditions.
   - Co-occurrence of the predicate and the case $\langle v, c \rangle$ are higher than the significance level ($p \leq 0.001$; $LLR \geq 10.83$) by the log-likelihood-ratio test.
   - The case of the predicate appears at least 10% more frequently than the average of all predicates.

We constructed two OCI dictionaries. The Blog dictionary contains about 480k predicates from one year of blogs (about 2.3G sentences,). The News dictionary contains about 200k predicates from 12 years of newspaper articles (about 7.7M sentences). The coverage of predicates in the training set of the Dialogue Corpus was 98.5% by the Blog dictionary and 96.4% by the News dictionary.

5.2.2 Dependency Language Models

Dependency language models (LMs) represent semantic/pragmatic collocations among predicate v, case c, and noun phrase n. The generation probabilities of v, c, and n are computed by n-gram models. More concretely, the following real values are computed. The purpose of the biases (probabilities involved <unk>) is to correct the values to be positive.

- $\log P(n|c, v) - \log P(\texttt{<unk>}|c, v)$
- $\log P(v|c, n) - \log P(v|c, \texttt{<unk>})$
- $\log P(c|n) - \log P(c|\texttt{<unk>})$

Each dependency LM is constructed from the tuples of $\langle v, c, n \rangle$ extracted in Section 5.2.1 using the SRILM (Stolcke et al., 2011). Note that since the obligatory case information corresponds to the generation probability of the case ($P(c|v)$), we exclude it from the dependency LMs.

Similar to the OCI dictionaries, we constructed two sets of dependency language models from the Blog and the News sentences. The coverage of triples $\langle v, c, n \rangle$ appeared in the training set of the Dialogue Corpus was 76.4% by the Blog LMs and 38.3% by the News LMs. The Blog LMs cover the Dialogue Corpus more comprehensively than the News LMs.

6 Experiments

We evaluate the accuracies of the proposed PASA on the Dialogue Corpus (Table 1) from the perspectives of parameter adaptation and the effect of the automatically acquired knowledge. The evaluation metric is F-measure of each case (includes exophora identification).

Case	Type	# of Args.	a) Adaptation OCI:Blog LMs:Blog	b) NAIST♡ OCI:Blog LMs:Blog	c) Dialogue♢ OCI:Blog LMs:Blog	d) Adaptation OCI:News♠ LMs:Blog	e) Adaptation OCI:Blog LMs:News♣
Nominative	Dep	1,811	**83.3**% ♡♢	77.6%	82.7%	83.0%	82.7%
	Zero-Intra	511	37.4%	**43.7**% ♡	36.6%	36.5%	38.1%
	Zero-Inter	767	8.6% ♣	**9.1**%	9.0%	8.3%	4.5%
	exo1	1,193	70.2% ♡	13.5%	69.9%	70.1%	**70.3**%
	exo2	281	46.8% ♡♢	0.0%	43.1%	**47.2**%	46.8%
	exog	767	46.8% ♡	32.5%	27.9%	47.2%	**47.7**% ♣
	Total	5,330	**61.5**% ♡	44.4%	61.1%	61.4%	61.4%
Accusative	Dep	614	**84.2**% ♡♢ ♣	78.6%	81.5%	**84.2**%	82.4%
	Zero-Intra	149	42.9% ♡ ♠♣	27.1%	**45.0**%	38.9%	34.3%
	Zero-Inter	399	30.4% ♡ ♣	0.5%	**30.9**%	29.4%	24.3%
	exo1	19	0.0%	0.0%	0.0%	9.5%	**10.0**%
	exo2	7	0.0%	0.0%	0.0%	0.0%	0.0%
	exog	98	25.6% ♡	0.0%	**27.9**%	25.2%	25.6%
	Total	1,286	**59.0**% ♡ ♣	51.6%	58.9%	58.4%	56.0%
Dative	Dep	566	80.5% ♡♢	54.0%	79.0%	80.1%	**80.7**%
	Zero-Intra	70	**20.7**% ♡ ♣	0.0%	20.0%	**20.7**%	11.8%
	Zero-Inter	169	14.6% ♡	0.0%	**14.8**%	14.4%	13.4%
	exo1	32	0.0%	0.0%	0.0%	0.0%	0.0%
	exo2	4	0.0%	0.0%	0.0%	0.0%	0.0%
	exog	265	**45.4**% ♡♢	0.0%	43.1%	44.0%	44.9%
	Total	1,106	**58.6**% ♡♢	32.2%	57.2%	58.2%	58.4%

Table 4: F-measures among Methods/OCI dictionary/Dependency LMs on Dialogue Test Set
The bold values denote the highest F-measures among all methods. The marks ♡, ♢, ♠, ♣ denote significantly better methods by comparing a) with b), c), d), and e), respectively. We used the bootstrap resampling method (1,000 iterations) as the significance test, in which the significance level was 0.05.

6.1 Experiment 1: Effect of Parameter Adaptation

We compared three methods in order to evaluate parameter adaptation: a) The feature augmentation is applied to the training (Adaptation). b) Only the NAIST Corpus is used for training (NAIST Training). c) Only the Dialogue Corpus is used (Dialogue Training). The NAIST Training corresponds to a conventional PASA for newspaper articles. The results on the Dialogue test set are shown in the 4th, 5th, and 6th columns in Table 4.

First, comparing methods a) Adaptation and b) NAIST training, Adaptation was better than the NAIST training for most types (The ♡ mark denotes 'significantly better'). In particular, the total F-measures of all cases were significantly better than NAIST training. Focusing on the types of arguments, the most characteristic results were exophoras of the first/second persons (exo1 and exo2) of the nominative case. These two types dominate of the nominative case (about 28%), and exo1 (70.2%) and exo2 (46.8%) became analyzable. Other types such as the Zero-Inter and the exog of the accusative and dative cases, which could not be analyzed by NAIST training, became analyzable.

Comparing methods a) Adaptation and c) Dialogue training (c.f., ♢), the F-measures of Dialogue training approached those of Adaptation even though the size of the Dialogue Corpus was small. Only the F-measure of the dative case of Adaptation was significantly better than Dialogue training in total. This does not imply that the corpus size is sufficient. Rather, we suppose that the Adaptation strategy could not adequately utilize the advantages of the NAIST Corpus. Adding more dialogue data would further improve the accuracies on the Dialogue test set.

6.2 Experiment 2: Differences among Automatically Acquired Knowledge

The columns a), d), and e) in Table 4 show the results for the proposed method (Adaptation). Note that the combination of the OCI dictionary and the dependency language models were changed to a) ⟨Blog, Blog⟩, d) ⟨News, Blog⟩, and e) ⟨Blog, News⟩.

When the OCI dictionary was changed from a) Blog to d) News (c.f., ♠), there were no significant differences in almost all types except for the Zero-Intra of the accusative case. We suppose that this

is because the coverage of the Blog and News dictionaries were almost the same, and obligatory cases of predicates are general information regardless of the domain.

On the contrary, when the dependency LMs were changed from a) Blog to e) News (c.f., ♣), the F-measures of some types significantly dropped, especially the `Zero-Intra` and `Zero-Inter` types, which are strongly influenced by semantic relation. For example, the `Zero-Inter` type of the accusative case was changed from 30.4% to 24.3%, and the F-measure consequently decreased by 3.0 points in total in the accusative case. Zero-anaphora resolution cannot rely on syntax, and the dependency LMs that measure semantic collocation become relatively important. The Blog LMs yielded greater coverage than the News LMs in this experiment. We can conclude that high-coverage LMs are better for improving the zero-anaphora resolution.

7 Conclusion

This paper presented predicate-argument structure analysis with zero-anaphora resolution for dialogues. We regarded this task as a kind of domain adaptation from newspaper articles, which are conventionally studied, to dialogues. The model parameters were adapted to the dialogues by using a domain adaptation technique. In order to address the out-of-vocabulary issue, the obligatory case information and the dependency language models were constructed from large text corpora and applied to the selectors.

As a result, arguments that could not be analyzed by PASA for newspaper articles (e.g., zero-pronouns of the first and second persons in the nominative case) became analyzable by adding only a small number of dialogues. The parameter adaptation achieved some improvement. Moreover, we confirmed that high-coverage dependency LMs contribute to improving zero-anaphora resolution and the overall accuracy.

Although we focused on parameter distribution and out-of-vocabulary in this paper, there are the other differences between dialogues and newspaper articles. For example, we did not discuss the exchange of turns, which is a special phenomenon of dialogues. To consider further phenomena is our future work. We are also evaluating the effectiveness of our PASA by incorporating it into a dialogue system (Higashinaka et al., 2014).

References

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In Hwee Tou Ng and Ellen Riloff, editors, *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 89–97, Boston, Massachusetts, USA, May.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan, June.

Hal Daumé, III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic, June.

Matthew Gerber and Joyce Y. Chai. 2012. Semantic role labeling of implicit arguments for nominal predicates. *Computational Linguistics*, 38(4):755–798.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Ryuichiro Higashinaka, Kenji Imamura, Toyomi Meguro, Chiaki Miyazaki, Nozomi Kobayashi, Hiroaki Sugiyama, Toru Hirano, Toshiro Makino, and Yoshihiro Matsuo. 2014. Towards an open domain conversational system fully based on natural language processing. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014)*, Dublin, Ireland, August.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA, June.

Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. 2007. Annotating a Japanese text corpus with predicate-argument and coreference relations. In *Proceedings of the Linguistic Annotation Workshop*, pages 132–139, Prague, Czech Republic, June.

Kenji Imamura, Kuniko Saito, and Tomoko Izumi. 2009. Discriminative approach to predicate-argument structure analysis with zero-anaphora resolution. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 85–88, Singapore, August.

Zheng Ping Jiang and Hwee Tou Ng. 2006. Semantic role labeling of NomBank: A maximum entropy approach. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 138–145, Sydney, Australia, July.

Daisuke Kawahara and Sadao Kurohashi. 2002. Fertilization of case frame dictionary for robust Japanese case analysis. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002)*, pages 425–431, Taipei, Taiwan, August.

Mamoru Komachi, Ryu Iida, Kentaro Inui, and Yuji Matsumoto. 2007. Learning-based argument structure analysis of event-nouns in Japanese. In *Proceedings of the Conference of the Pacific Association for Computational Linguistics (PACLING)*, pages 208–215, Melbourne, Australia, September.

Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *CoNLL 2002: Proceedings of the 6th Conference on Natural Language Learning 2002 (COLING 2002 Post-Conference Workshops)*, pages 63–69, Taipei, Taiwan.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 230–237, Barcelona, Spain, July.

Egoitz Laparra and German Rigau. 2013. ImpAr: A deterministic algorithm for implicit semantic role labelling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1180–1189, Sofia, Bulgaria, August.

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. 2008. Semantic role labeling: An introduction to the special issue. *Computational Linguistics*, 34(2):145–159.

Yuichiro Matsubayashi, Ryu Iida, Ryohei Sasano, Hikaru Yokono, Suguru Matsuyoshi, Atsushi Fujita, Yusuke Miyao, and Kentaro Inui. 2014. Issues on annotation guidelines for Japanese predicate-argument structures. *Journal of Natural Language Processing*, 21(2):333–377, April. in Japanese.

Martha Palmer, Daniel Gildia, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105.

Sameer S. Pradhan, Wayne Ward, and James H. Martin. 2008. Towards robust semantic role labeling. *Computational Linguistics*, 34(2):289–310.

Sameer Pradhan, Alessandro Moschitti, and Nianwen Xue, editors. 2012. *Joint Conference on EMNLP and CoNLL: Proceeding of the Shared Task: Modeling Multilingual Unrestricted Coreference in Onto Notes*, Jeju, Korea, July.

Ryohei Sasano, Daisuke Kawahara, and Sadao Kurohashi. 2008. A fully-lexicalized probabilistic model for Japanese zero anaphora resolution. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 769–776, Manchester, UK, August.

Ryohei Sasano, Daisuke Kawahara, Sadao Kurohashi, and Manabu Okumura. 2013. Automatic knowledge acquisition for case alternation between the passive and active voices in Japanese. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1213–1223, Seattle, Washington, USA, October.

Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. SRILM at sixteen: Update and outlook. In *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU 2011)*, Waikoloa, Hawaii, December.

Hirotoshi Taira, Sanae Fujita, and Masaaki Nagata. 2008. A Japanese predicate argument structure analysis using decision lists. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 523–532, Honolulu, Hawaii, October.

Katsumasa Yoshikawa, Masayuki Asahara, and Yuji Matsumoto. 2011. Jointly extracting Japanese predicate-argument relation with markov logic. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1125–1133, Chiang Mai, Thailand, November.

Feature Embedding for Dependency Parsing

Wenliang Chen[†], **Yue Zhang**[‡], **and Min Zhang**[†*]
[†]School of Computer Science and Technology, Soochow University, China
[‡]Singapore University of Technology and Design, Singapore
{wlchen, mzhang}@suda.edu.cn
yue_zhang@sutd.edu.sg

Abstract

In this paper, we propose an approach to automatically learning feature embeddings to address the feature sparseness problem for dependency parsing. Inspired by word embeddings, feature embeddings are distributed representations of features that are learned from large amounts of auto-parsed data. Our target is to learn feature embeddings that can not only make full use of well-established hand-designed features but also benefit from the hidden-class representations of features. Based on feature embeddings, we present a set of new features for graph-based dependency parsing models. Experiments on the standard Chinese and English data sets show that the new parser achieves significant performance improvements over a strong baseline.

1 Introduction

Discriminative models have become the dominant approach for dependency parsing (Nivre et al., 2007; Zhang and Clark, 2008; Hatori et al., 2011). State-of-the-art accuracies have been achieved by the use of rich features in discriminative models (Carreras, 2007; Koo and Collins, 2010; Bohnet, 2010; Zhang and Nivre, 2011). While lexicalized features extracted from non-local contexts enhance the discriminative power of parsers, they are relatively sparse. Given a limited set of training data (typically less than 50k sentences for dependency parsing), the chance of a feature occurring in the training data but not in the test data can be high.

Another limitation on features is that many are typically derived by (manual) combination of atomic features. For example, given the head word (w_h) and part-of-speech tag (p_h), dependent word (w_d) and part-of-speech tag (p_d), and the label (l) of a dependency arc, state-of-the-art dependency parsers can have the combined features: $[w_h; p_h]$, $[w_h; p_h; w_d; p_d]$, $[w_h; p_h; w_d]$, and so on, in addition to the atomic features: $[w_h]$, $[p_h]$, etc. Such combination is necessary for high accuracies because the dominant approach uses linear models, which can not capture complex correlations between atomic features.

We tackle the above issues by borrowing solutions from word representations, which have been intensely studied in the NLP community (Turian et al., 2010). In particular, distributed representations of words have been used for many NLP problems, which represent a word by information from the words it frequently co-occurs with (Lin, 1997; Curran, 2005; Collobert et al., 2011; Bengio, 2009; Mikolov et al., 2013b). The representation can be learned from large amounts of raw sentences, and hence used to reduce OOV rates in test data. In addition, since the representation of each word carries information about its context words, it can also be used to calculate word similarity (Mikolov et al., 2013a), or used as additional semantic features (Koo et al., 2008).

In this paper, we show that a distributed representation can be learned for features also. Learned from large amount of automatically parsed data, the representation of each feature can be defined on the

*Corresponding author

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 816–826, Dublin, Ireland, August 23-29 2014.

features it frequently co-occurs with. Similar to words, the feature representation can be used to reduce the rate of unseen features in test data, and to capture inherent correlations between features. Borrowing terminologies from word embeddings, we call the feature representation **feature embeddings**.

Compared with the task of learning word embeddings, the task of learning feature embeddings is more difficult because the size of features is much larger than the vocabulary size and tree structures are more complex than word sequences. This requires us to find an effective embedding format and an efficient inference algorithm. Traditional LSA and RNN (Collobert et al., 2011; Bengio, 2009) models turn out to be very slow for feature embeddings. Recently, Mikolov et al. (2013a) and Mikolov et al. (2013b) introduce efficient models to learn high-quality word embeddings from extremely large amounts of raw text, which offer a possible solution to the efficiency issue of learning feature embeddings. We adapt their approach for learning feature embeddings, showing how an unordered feature context can be used to learn the representation of a set of complex features. Using this method, a large number of embeddings are trained from automatically parsed texts, based on which a set of new features are designed and incorporated into a graph-based parsing model (McDonald and Nivre, 2007).

We conduct experiments on the standard data sets of the Penn English Treebank and the Chinese Treebank V5.1. The results indicate that our proposed approach significantly improves parsing accuracies.

2 Background

In this section, we introduce the background of dependency parsing and build a baseline parser based on the graph-based parsing model proposed by McDonald et al. (2005).

2.1 Dependency parsing

Given an input sentence $x = (w_0, w_1, ..., w_i, ..., w_m)$, where w_0 is ROOT and w_i ($i \neq 0$) refers to a word, the task of dependency parsing is to find y^* which has the highest score for x,

$$y^* = \arg\max_{y \in Y(x)} score(x, y)$$

where $Y(x)$ is the set of all the valid dependency trees for x. There are two major models (Nivre and McDonald, 2008): the transition-based model and graph-based model, which showed comparable accuracies for a wide range of languages (Nivre et al., 2007; Bohnet, 2010; Zhang and Nivre, 2011; Bohnet and Nivre, 2012). We apply feature embeddings to a graph-based model in this paper.

2.2 Graph-based parsing model

We use an ordered pair $(w_i, w_j) \in y$ to define a dependency relation in tree y from word w_i to word w_j (w_i is the head and w_j is the dependent), and G_x to define a graph that consists of a set of nodes $V_x = \{w_0, w_1, ..., w_i, ..., w_m\}$ and a set of arcs (edges) $E_x = \{(w_i, w_j) | i \neq j, w_i \in V_x, w_j \in (V_x - \{w_0\})\}$. The parsing model of McDonald et al. (2005) searches for the maximum spanning tree (MST) in G_x.

We denote $Y(G_x)$ as the set of all the subgraphs of G_x that are valid spanning trees (McDonald and Nivre, 2007). The score of a dependency tree $y \in Y(G_x)$ is the sum of the scores of its subgraphs,

$$score(x, y) = \sum_{g \in y} score(x, g) = \sum_{g \in y} \mathbf{f}(x, g) \cdot \mathbf{w} \quad (1)$$

where g is a spanning subgraph of y, which can be a single arc or adjacent arcs, $\mathbf{f}(x, g)$ is a high-dimensional feature vector based on features defined over g and x, and $\mathbf{w}$ refers to the weights for the features. In this paper we assume that a dependency tree is a spanning projective tree.

2.3 Baseline parser

We use the decoding algorithm proposed by Carreras (2007) and use the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2003; McDonald et al., 2005) to train feature weights $\mathbf{w}$. We use the feature templates of Bohnet (2010) as our base feature templates, which produces state-of-the-art accuracies. We further extend the features by introducing more lexical features to the base features. The

First-order
$[wp]_h$, $[wp]_d$, d(h, d)
$[wp]_h$, d(h, d)
w_d, p_d, d(h, d)
$[wp]_d$, d(h, d)
w_h, p_h, w_d, p_d, d(h, d)
p_h, w_h, p_d, d(h, d)
w_h, w_d, p_d, d(h, d)
w_h, p_h, $[wp]_d$, d(h, d)
p_h, p_b, p_d, d(h, d)
p_h, p_{h+1}, p_{d-1}, p_d, d(h, d)
p_{h-1}, p_h, p_{d-1}, p_d, d(h, d)
p_h, p_{h+1}, p_d, p_{d+1}, d(h, d)
p_{h-1}, p_h, p_d, p_{d+1}, d(h, d)
Second-order
p_h, p_d, p_c, d(h, d, c)
w_h, w_d, c$_w$, d(h, d, c)
p_h, $[wp]_c$, d(h, d, c)
p_d, $[wp]_c$, d(h, d, c)

Second-order (continue)
w_h, $[wp]_c$, d(h, d, c)
w_d, $[wp]_c$, d(h, d, c)
$[wp]_h$, $[wp]_{h+1}$, $[wp]_c$, d(h, d, c)
$[wp]_{h-1}$, $[wp]_h$, $[wp]_c$, d(h, d, c)
$[wp]_h$, $[wp]_{c-1}$, $[wp]_c$, d(h, d, c)
$[wp]_h$, $[wp]_c$, $[wp]_{c+1}$, d(h, d, c)
$[wp]_{h-1}$, $[wp]_h$, $[wp]_{c-1}$, $[wp]_c$, d(h, d, c)
$[wp]_h$, $[wp]_{h+1}$, $[wp]_{c-1}$, $[wp]_c$, d(h, d, c)
$[wp]_{h-1}$, $[wp]_h$, $[wp]_c$, $[wp]_{c+1}$, d(h, d, c)
$[wp]_h$, $[wp]_{h+1}$, $[wp]_c$, $[wp]_{c+1}$, d(h, d, c)
$[wp]_d$, $[wp]_{d+1}$, $[wp]_c$, d(h, d, c)
$[wp]_{d-1}$, $[wp]_d$, $[wp]_c$, d(h, d, c)
$[wp]_d$, $[wp]_{c-1}$, $[wp]_c$, d(h, d, c)
$[wp]_d$, $[wp]_c$, $[wp]_{c+1}$, d(h, d, c)
$[wp]_d$, $[wp]_{d+1}$, $[wp]_{c-1}$, $[wp]_c$, d(h, d, c)
$[wp]_d$, $[wp]_{d+1}$, $[wp]_c$, $[wp]_{c+1}$, d(h, d, c)
$[wp]_{d-1}$, $[wp]_d$, $[wp]_{c-1}$, $[wp]_c$, d(h, d, c)
$[wp]_{d-1}$, $[wp]_d$, $[wp]_c$, $[wp]_{c+1}$, d(h, d, c)

Table 1: Base feature templates.

base feature templates are listed in Table 1, where h and d refer to the head, the dependent, respectively, c refers to d's sibling or child, b refers to the word between h and d, $+1$ (-1) refers to the next (previous) word, w and p refer to the surface word and part-of-speech tag, respectively, $[wp]$ refers to the surface word or part-of-speech tag, $d(h, d)$ is the direction of the dependency relation between h and d, and $d(h, d, c)$ is the directions of the relation among h, d, and c.

We train a parser with the base features and use it as the Baseline parser. Defining $\mathbf{f}_b(x, g)$ as the base features and $\mathbf{w}_b$ as the corresponding weights, the scoring function becomes,

$$score(x, g) = \mathbf{f}_b(x, g) \cdot \mathbf{w}_b \tag{2}$$

3 Feature Embeddings

Our goal is to reduce the sparseness of rich features by learning a distributed representation of features, which is dense and low dimensional. We call the distributed feature representation **feature embeddings**. In the representation, each dimension represents a hidden-class of the features and is expected to capture a type of similarities or share properties among the features.

The key to learn embeddings is making use of information from a local context, and to this end various methods have been proposed for learning word embeddings. Lin (1997) and Curran (2005) use the count of words in a surrounding word window to represent distributed meaning of words. Brown et al. (1992) uses bigrams to cluster words hierarchically. These methods have been shown effective on words. However, the number of features is much larger than the vocabulary size, which makes it infeasible to apply them on features. Another line of research induce word embeddings using neural language models (Bengio, 2008). However, the training speed of neural language models is too slow for the high dimensionality of features. Mikolov et al. (2013b) and Mikolov et al. (2013a) introduce efficient methods to directly learn high-quality word embeddings from large amounts of unstructured raw text. Since the methods do not involve dense matrix multiplications, the training speed is extremely fast.

We adapt the models of Mikolov et al. (2013b) and Mikolov et al. (2013a) for learning feature embeddings from large amounts of automatically parsed dependency trees. Since feature embeddings have a high computational cost, we also use Negative sampling technique in the learning stage (Mikolov et al., 2013b). Different from word embeddings, the input of our approach is features rather than words, and the feature representations are generated from tree structures instead of word sequences. Consequently,

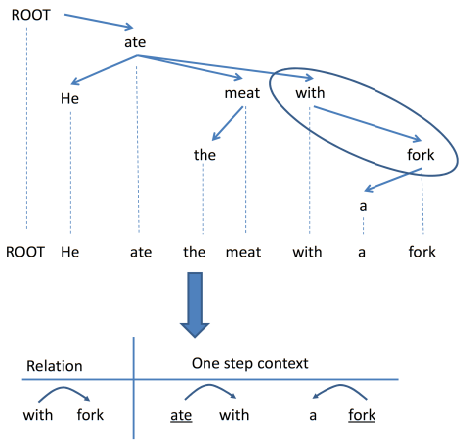


Figure 1: An example of one-step context.

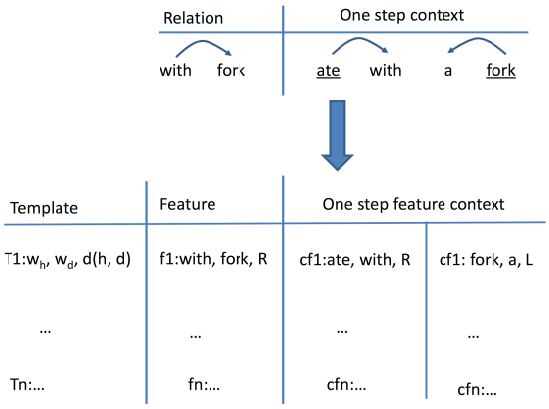


Figure 2: One-step surrounding features.

we give a definition of unordered feature contexts and adapt the algorithms of Mikolov et al. (2013b) for feature embeddings.

3.1 Surrounding feature context

The most important difference between features and words is the contextual structure. Given a sentence $x = w_1, w_2, ..., w_n$ and its dependency tree y, we define the M-step context as a set of relations reachable within M steps from the current relation. Here one step refers to one dependency arc. For instance, the one-step context includes the surrounding relations that can be reached in one arc, as shown in Figure 1. In the figure, for the relation between "with" and "fork", the relation between "ate" and "with" is in the one-step context, while the relation between "He" and "ate" is in the two-step context because it can be reached via the arc between "ate" and "with". A larger M results in more contextual features and thus might lead to a more accurate embedding, but at the expense of training speed.

Based on the M-step context, we use surrounding features to represent the features on the current dependency relations. The surrounding features are defined on the relations in the M-step context. Take 1-step context as an example. Figure 2 shows the representations for the current relation between "with" and "fork" in Figure 1. Given the current relation and the relations in its one-step context, we generate the features based on the base feature templates. In Figure 2 the current feature "f1:with, fork, R" can be represented by the surrounding features "cf1:ate, with, R" and "cf1: fork, a, L" based on the template "T1:w_h, w_d, d(h, d)". Similarly, all the features on the current relation are represented by the features on the relations in the one-step context. To reduce computational cost, we generate for every feature its contextual features based on the same feature template. As a result, the embeddings for each feature template is trained separately. In the experiments, we use one-step context for learning feature embeddings.

3.2 Feature Embedding model

We adapt the models of Mikolov et al. (2013b) and Mikolov et al. (2013a) to infer feature embeddings (FE). Based on the representation of surrounding context, the input to the learning models is a set of features and the output is feature embeddings as shown in Figure 3. For each dependency tree in large amounts of auto-parsed data, we generate the base features and associate them with their surrounding contextual features. Then all the base features are put into a set, which is used as the training instances for learning models.

In the embedding model, we use the features on the current dependency arc to predict the surrounding features, as shown in Figure 4. Given sentences and their corresponding dependency trees Y, the objective of the model is to maximize the conditional log-likelihood of context features,

$$\sum_{y \in Y} \sum_{f \in F_y} \sum_{cf \in CF_f} \log(p(cf|f)) \quad (3)$$

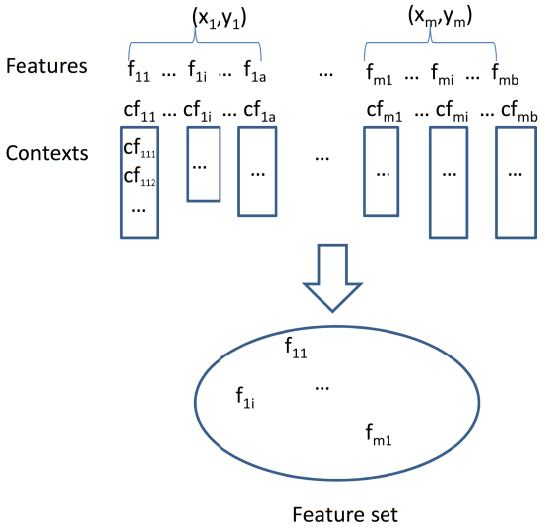


Figure 3: Input feature set.

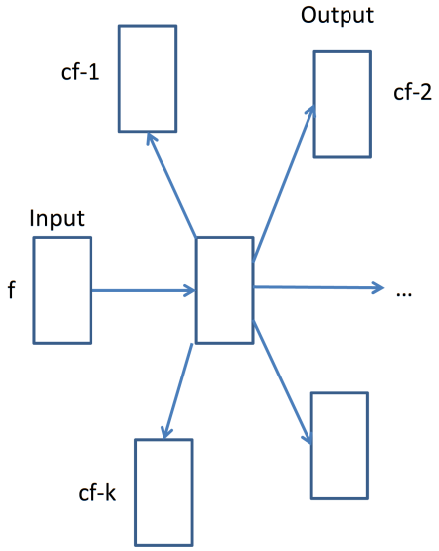


Figure 4: The feature embedding model.

where F_y is a set of features generated from tree y, CF_f is the set of surrounding features in the M-step context of feature f. $p(cf|f)$ can be computed by using the softmax function (Mikolov et al., 2013b), for which the input is f and the output is cf,

$$p(cf|f) = \frac{exp({v'_{cf}}^{\mathrm{T}} v_f)}{\sum_{i=1}^{F} exp({v'_{cf_i}}^{\mathrm{T}} v_f)} \tag{4}$$

where v_f and v'_f are the input and output vector representations of f, and F is the number of features in the feature table. The formulation is impractical for large data because the number of features is large (in the millions) and the computational cost is too high.

To compute the probabilities efficiently, we use the Negative sampling method proposed by Mikolov et al. (2013b), which approximates the probability by the correct example and K negative samples for each instance. The formulation to compute $log(p(cf|f))$ is,

$$\log \sigma({v'_{cf}}^{\mathrm{T}} v_f) + \sum_{k=1}^{K} \mathrm{E}_{cf_k \sim P(cf)} [\log \sigma(-{v'_{cf_k}}^{\mathrm{T}} v_f)] \tag{5}$$

where $\sigma(z) = 1/(1 + exp(-z))$ and $P(f)$ is the noise distribution on the data. Following the setting of Mikolov et al. (2013b), we set K to 5 in our experiments.

We predict the set of features one by one. Stochastic gradient ascent is used to perform the following iterative update after predicting the i^{th} feature,

$$\theta \leftarrow \theta + \alpha(\frac{\partial \sum_{cf} log(p(cf_i|f)}{\partial \theta}) \tag{6}$$

where α is the learning rate and θ includes the parameters of the model and the vector representations of features. The initial value of α is 0.025. If the log-likelihood does not improve significantly after one update, the rate is halved (Mikolov et al., 2009).

3.3 Distributed representation

Based on the proposed surrounding context, we use the feature embedding model with the help of the Negative sampling method to learn feature embeddings. For each base template T_i, the distributed representations are stored in a matrix $\mathcal{M}_i \in \mathbb{R}^{d \times |\mathcal{F}_i|}$, where d is the number of dimensions (to be chosen in the experiments) and $|\mathcal{F}_i|$ is the size of the features $\mathcal{F}_i$ for T_i. For each feature $f \in \mathcal{F}_i$, its vector is $v_f = [v_1, ..., v_d]$.

$< j : T(f) \cdot \Phi(v_j) >$ for $j \in [1, d]$
$< j : T(f) \cdot \Phi(v_j), w_h >$ for $j \in [1, d]$

Table 2: FE-based templates.

4 Parsing with feature embeddings

In this section, we discuss how to apply the feature embeddings to dependency parsing.

4.1 FE-based feature templates

The base parsing model contains only binary features, while the values in the feature embedding representation are real numbers that are not in a bounded range. If the range of the values is too large, they will exert much more influence than the binary features. To solve this problem, we define a function $\Phi(v_i)$ (details are given in Section 4.3) to convert real values to discrete values. The vector $v_f = [v_1, ..., v_d]$ is converted into $v_f^N = [\Phi(v_1), ..., \Phi(v_d)]$.

We define a set of new feature templates for the parsing models, capturing feature embedding information. Table 2 shows the new templates, where $T(f)$ refers to the base template type of feature f. We remove any new feature related to the surface form of the head if the word is not one of the Top-N most frequent words in the training data. We used N=1000 for the experiments, which reduces the size of the feature sets.

4.2 FE parser

We combine the base features with the new features by a new scoring function,

$$score(x, g) = \mathbf{f}_b(x, g) \cdot \mathbf{w}_b + \mathbf{f}_e(x, g) \cdot \mathbf{w}_e \tag{7}$$

where $\mathbf{f}_b(x, g)$ refers to the base features, $\mathbf{f}_e(x, g)$ refers to the FE-based features, and $\mathbf{w}_b$ and $\mathbf{w}_e$ are their corresponding weights, respectively. The feature weights are learned during training using MIRA (Crammer and Singer, 2003; McDonald et al., 2005).

We use the same decoding algorithm in the new parser as in the Baseline parser. The new parser is referred to as the FE Parser.

4.3 Discretization functions

There are various functions to convert the real values in the vectors into discrete values. Here, we use a simple method. First, for the i^{th} base template, the values in the j^{th} dimension are sorted in decreasing order L_{ij}. We divide the list into two parts for positive (L_{ij+}) and negative (L_{ij-}), respectively, and define two functions. The first function is,

$$\Phi_1(v_j) = \begin{cases} +B1 & \text{if } v_j \text{ is in top 50\% in } L_{ij+} \\ +B2 & \text{if } v_j \text{ is in bottom 50\% in } L_{ij+} \\ -B1 & \text{if } v_j \text{ is in top 50\% in } L_{ij-} \\ -B2 & \text{if } v_j \text{ is in bottom 50\% in } L_{ij-} \end{cases}$$

The second function is,

$$\Phi_2(v_j) = \begin{cases} +B1 & \text{if } v_j \text{ is in top 50\% in } L_{ij+} \\ -B2 & \text{if } v_j \text{ is in bottom 50\% in } L_{ij-} \end{cases}$$

In Φ_2, we only consider the values ("+B1" and "-B2"), which have strong values (positive or negative) on each dimension, and omit the values which are close to zero. We refer the systems with Φ_1 as M1 and the ones with Φ_2 as M2. We also tried the original continuous values and the scaled values as used by Turian et al. (2010), but the results were negative.

5 Experiments

We conducted experiments on English and Chinese data, respectively.

	train	dev	test
PTB	2-21	22	23
CTB5	001-815	886-931	816-885
	1001-1136	1148-1151	1137-1147

Table 3: Data sets of PTB and CTB5.

	# of words	# of sentences
BLLIP WSJ	43.4M	1.8M
Gigaword Xinhua	272.3M	11.7M

Table 4: Information of raw data.

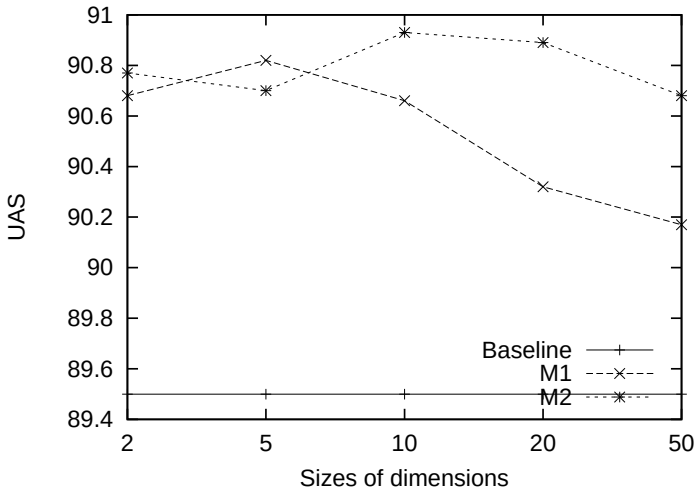


Figure 5: Effect of different sizes of embeddings on the development data.

5.1 Data sets

We used the Penn Treebank (PTB) to generate the English data sets, and the Chinese Treebank version 5.1 (CTB5) to generate the Chinese data sets. "Penn2Malt"[1] was used to convert the data into dependency structures with the English head rules of Yamada and Matsumoto (2003) and the Chinese head rules of Zhang and Clark (2008). The details of data splits are listed in Table 3, where the data partition of Chinese were chosen to match previous work (Duan et al., 2007; Li et al., 2011b; Hatori et al., 2011).

Following the work of Koo et al. (2008), we used a tagger trained on the training data to provide part-of-speech (POS) tags for the development and test sets, and used 10-way jackknifing to generate part-of-speech tags for the training set. For English we used the MXPOST (Ratnaparkhi, 1996) tagger and for Chinese we used a CRF-based tagger with the feature templates defined in Zhang and Clark (2008). We used gold-standard segmentation in the CTB5 experiments. The accuracies of part-of-speech tagging are 97.32% for English and 93.61% for Chinese on the test sets, respectively.

To obtain feature contexts, we processed raw data to obtain dependency trees. For English, we used the BLLIP WSJ Corpus Release 1.[2] For Chinese, we used the Xinhua portion of Chinese Gigaword[3] Version 2.0 (LDC2009T14). The statistical information of raw data sets is listed in Table 4. The MXPOST part-of-speech tagger and the Baseline dependency parser trained on the training data were used to process the sentences of the BLLIP WSJ corpus. For Chinese, we need to perform word segmentation and part-of-speech tagging before parsing. The MMA system (Kruengkrai et al., 2009) trained on the training data was used to perform word segmentation and tagging, and the Baseline parser was used to parse the sentences in the Gigaword corpus.

We report the parser quality by the unlabeled attachment score (UAS), i.e., the percentage of tokens (excluding all punctuation tokens) with the correct HEAD. We also report the scores on complete dependency tree matches (COMP).

[1] http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html
[2] We excluded the texts of PTB from the BLLIP WSJ Corpus.
[3] We excluded the texts of CTB5 from the Gigaword data.

	UAS	COMP
Baseline	92.78	48.08
Baseline+BrownClu	93.37	49.26
M2	93.74	50.82
Koo and Collins (2010)	93.04	N/A
Zhang and Nivre (2011)	92.9	48.0
Koo et al. (2008)	93.16	N/A
Suzuki et al. (2009)	93.79	N/A
Chen et al. (2009)	93.16	47.15
Zhou et al. (2011)	92.64	46.61
Suzuki et al. (2011)	94.22	N/A
Chen et al. (2013)	93.77	51.36

Table 5: Results on English data.
N/A=Not Available.

	POS	UAS	COMP
Baseline	93.61	81.04	29.73
M2	93.61	82.94	31.72
Li et al. (2011a)	93.08	80.74	29.11
Hatori et al. (2011)	93.94	81.33	29.90
Li et al. (2012)	94.51	81.21	N/A
Chen et al. (2013)	N/A	83.08	32.21

Table 6: Results on Chinese data.
N/A=Not Available.

5.2 Development experiments

In this section, we use the English development data to investigate the effects of different vector sizes of feature embeddings, and compare the systems with the discretization functions Φ_1 (M1) and Φ_2 (M2) (defined in Section 4.3), respectively. To reduce the training time, we used 10% of the labeled training data to train the parsing models.

Turian et al. (2010) reported that the optimal size of word embedding dimensions was task-specific for NLP tasks. Here, we investigated the effect of different sizes of embedding dimensions on dependency parsing. Figure 5 shows the effect on UAS scores as we varied the vector sizes. The systems with FE-based features always outperformed the Baseline. The curve of M2 was almost flat and we found that M1 performed worse as the sizes increased. Overall, M2 performed better than M1. For M2, 10-dimensional embeddings achieved the highest score among all the systems. Based on the above observations, we chose the following settings for further evaluations: 10-dimensional embeddings for M2.

5.3 Final results on English data

We trained the M2 model on the full training data and evaluated it on the English testing data. The results are shown in Table 5. The parser using the FE-based features outperformed the Baseline. We obtained absolute improvements of 0.96 UAS points. As for the COMP scores, M2 achieved absolute improvement of 2.74 over the Baseline. The improvements were significant in McNemar's Test ($p < 10^{-7}$) (Nivre et al., 2004).

We listed the performance of the related systems in Table 5. We also added the cluster-based features of Koo et al. (2008) to our baseline system listed as "Baseline+BrownClu" in Table 5. From the table, we found that our FE parsers obtained comparable accuracies with the previous state-of-the-art systems. Suzuki et al. (2011) reported the best result by combining their method with the method of Koo et al. (2008). We believe that the performance of our parser can be further enhanced by integrating their methods.

5.4 Final results on Chinese data

We also evaluated the systems on the testing data for Chinese. The results are shown in Table 6. Similar to the results on English, the parser using the FE-based features outperformed the Baseline. The improvements were significant in McNemar's Test ($p < 10^{-8}$) (Nivre et al., 2004).

We listed the performance of the related systems[4] on Chinese in Table 6. From the table, we found that the scores of our FE parser was higher than most of the related systems and comparable with the results of Chen2013, which was the best reported scores so far.

[4]We did not include the result (83.96) of Wu et al. (2013) because their part-of-speech tagging accuracy is 97.7%, much higher than ours and other work. Their tagger includes rich external resources.

6 Related work

Learning feature embeddings are related to two lines of research: deep learning models for NLP, and semi-supervised dependency parsing.

Recent studies used deep learning models in a variety of NLP tasks. Turian et al. (2010) applied word embeddings to chunking and Named Entity Recognition (NER). Collobert et al. (2011) designed a unified neural network to learn distributed representations that were useful for part-of-speech tagging, chunking, NER, and semantic role labeling. They tried to avoid task-specific feature engineering. Socher et al. (2013) proposed a Compositional Vector Grammar, which combined PCFGs with distributed word representations. Zheng et al. (2013) investigated Chinese character embeddings for Chinese word segmentation and part-of-speech tagging. Wu et al. (2013) directly applied word embeddings to Chinese dependency parsing. In most cases, words or characters were the inputs to the learning systems and word/character embeddings were used for the tasks. Our work is different in that we explore distributed representations at the feature level and we can make full use of well-established hand-designed features.

We use large amounts of raw data to infer feature embeddings. There are several previous studies relevant to using raw data for dependency parsing. Koo et al. (2008) used the Brown algorithm to learn word clusters from a large amount of unannotated data and defined a set of word cluster-based features for dependency parsing models. Suzuki et al. (2009) adapted a Semi-supervised Structured Conditional Model (SS-SCM) to dependency parsing. Suzuki et al. (2011) reported the best results so far on the standard test sets of PTB using a condensed feature representation combined with the word cluster-based features of Koo et al. (2008). Chen et al. (2013) mapped the base features into predefined types using the information of frequencies counted in large amounts of auto-parsed data. The work of Suzuki et al. (2011) and Chen et al. (2013) were to perform feature clustering. Ando and Zhang (2005) presented a semi-supervised learning algorithm named alternating structure optimization for text chunking. They used a large projection matrix to map sparse base features into a small number of high level features over a large number of auxiliary problems. One of the advantages of our approach is that it is simpler and more general than that of Ando and Zhang (2005). Our approach can easily be applied to other tasks by defining new feature contexts.

7 Conclusion

In this paper, we have presented an approach to learning feature embeddings for dependency parsing from large amounts of raw data. Based on the feature embeddings, we represented a set of new features, which was used with the base features in a graph-based model. When tested on both English and Chinese, our method significantly improved the performance over the baselines and provided comparable accuracy with the best systems in the literature.

Acknowledgments

Wenliang Chen was supported by the National Natural Science Foundation of China (Grant No. 61203314) and Yue Zhang was supported by MOE grant 2012-T2-2-163. We would also thank the anonymous reviewers for their detailed comments, which have helped us to improve the quality of this work.

References

R.K. Ando and T. Zhang. 2005. A high-performance semi-supervised learning method for text chunking. *ACL*.

Yoshua Bengio. 2008. Neural net language models. In *Scholarpedia*, page 3881.

Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of EMNLP-CoNLL 2012*, pages 1455–1465. Association for Computational Linguistics.

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

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, T. J. Watson, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-Based n-gram Models of Natural Language. *Computational Linguistics*.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 957–961, Prague, Czech Republic, June. Association for Computational Linguistics.

Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Improving dependency parsing with subtrees from auto-parsed data. In *Proceedings of EMNLP 2009*, pages 570–579, Singapore, August.

Wenliang Chen, Min Zhang, and Yue Zhang. 2013. Semi-supervised feature transformation for dependency parsing. In *Proceedings of EMNLP 2013*, pages 1303–1313, Seattle, Washington, USA, October. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.*, 3:951–991.

James Curran. 2005. Supersense tagging of unknown nouns using semantic similarity. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 26–33, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Xiangyu Duan, Jun Zhao, and Bo Xu. 2007. Probabilistic models for action-based chinese dependency parsing. In *Proceedings of ECML/ECPPKDD*, Warsaw, Poland.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental joint pos tagging and dependency parsing in chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1216–1224, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of ACL 2010*, pages 1–11, Uppsala, Sweden, July. Association for Computational Linguistics.

T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, June.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging. In *Proceedings of ACL-IJCNLP2009*, pages 513–521, Suntec, Singapore, August. Association for Computational Linguistics.

Zhenghua Li, Wanxiang Che, and Ting Liu. 2011a. Improving chinese pos tagging with dependency parsing. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1447–1451, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011b. Joint models for chinese pos tagging and dependency parsing. In *Proceedings of EMNLP 2011*, UK, July.

Zhenghua Li, Min Zhang, Wanxiang Che, and Ting Liu. 2012. A separately passive-aggressive training algorithm for joint pos tagging and dependency parsing. In *Proceedings of the 24rd International Conference on Computational Linguistics (Coling 2012)*, Mumbai, India. Coling 2012 Organizing Committee.

Dekang Lin. 1997. Using syntactic dependency as local context to resolve word sense ambiguity. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 64–71, Madrid, Spain, July. Association for Computational Linguistics.

R. McDonald and J. Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of EMNLP-CoNLL*, pages 122–131.

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

Tomas Mikolov, Jiri Kopecky, Lukas Burget, Ondrej Glembek, and Jan Cernocky. 2009. Neural network based language models for highly inflective languages. In *Proceedings of ICASSP 2009*, pages 4725–4728. IEEE.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

J. Nivre and R. McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, June.

J. Nivre, J. Hall, and J. Nilsson. 2004. Memory-based dependency parsing. In *Proc. of CoNLL 2004*, pages 49–56.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP 1996*, pages 133–142.

Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of ACL 2013*. Citeseer.

Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of EMNLP2009*, pages 551–560, Singapore, August. Association for Computational Linguistics.

Jun Suzuki, Hideki Isozaki, and Masaaki Nagata. 2011. Learning condensed feature representations from large unsupervised data sets for supervised learning. In *Proceedings of ACL2011*, pages 636–641, Portland, Oregon, USA, June. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL 2010*, pages 384–394. Association for Computational Linguistics.

Xianchao Wu, Jie Zhou, Yu Sun, Zhanyi Liu, Dianhai Yu, Hua Wu, and Haifeng Wang. 2013. Generalization of words for chinese dependency parsing. In *Proceedings of IWPT 2013*, pages 73–81.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT 2003*, pages 195–206.

Y. Zhang and S. Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of EMNLP 2008*, pages 562–571, Honolulu, Hawaii, October.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of ACL-HLT2011*, pages 188–193, Portland, Oregon, USA, June. Association for Computational Linguistics.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *Proceedings of EMNLP 2013*, pages 647–657. Association for Computational Linguistics.

Guangyou Zhou, Jun Zhao, Kang Liu, and Li Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of ACL-HLT2011*, pages 1556–1565, Portland, Oregon, USA, June. Association for Computational Linguistics.

Identifying Emotional and Informational Support in Online Health Communities

Prakhar Biyani[1] **Cornelia Caragea**[2] **Prasenjit Mitra**[1] **John Yen**[1]
(1) College of Information Sciences and Technology, The Pennsylvania State University, USA
(2) Department of Computer Science and Engineering, University of North Texas, USA
pxb5080@ist.psu.edu, ccaragea@unt.edu, {pmitra,jyen}@ist.psu.edu

Abstract

A large number of online health communities exist today, helping millions of people with social support during difficult phases of their lives when they suffer from serious diseases. Interactions between members in these communities contain discussions on practical problems faced by people during their illness such as depression, side-effects of medications, etc and answers to those problems provided by other members. Analyzing these interactions can be helpful in getting crucial information about the community such as dominant health issues, identifying sentimental effects of interactions on individual members and identifying influential members. In this paper, we analyze user messages of an online cancer support community, Cancer Survivors Network (CSN), to identity the two types of social support present in them: *emotional* support and *informational* support. We model the task as a binary classification problem. We use several generic and novel domain-specific features. Experimental results show that we achieve high classification performance. We, then, use the classifier to predict the type of support in CSN messages and analyze the posting behaviors of regular members and influential members in CSN in terms of the type of support they provide in their messages. We find that influential members generally provide more emotional support as compared to regular members in CSN.

1 Introduction

Increasingly more people turn to online health communities (OHCs) to seek social support during their illnesses (LaCoursiere, 2001; Beaudoin and Tao, 2007). When people suffering from a serious disease such as cancer or AIDS *interact* with other people who have experienced similar medical conditions, they feel emotionally supported. In addition, through these interactions, people can obtain important information about the disease, e.g., about various medications, symptoms, and side-effects. Although authoritative health-related web sites contain the information they search for, obtaining this information directly from people in OHCs adds substantial value to it. Previous studies showed that obtaining social support in OHCs can help people feel better (Dunkel-Schetter, 1984; Maloney-Krichmar and Preece, 2005; Beaudoin and Tao, 2007; Vilhauer, 2009; Qiu et al., 2011).

As a result of online interactions in OHCs, a huge volume of user-generated content exists today on various issues/problems related to specific diseases. This content comprises of important information such as people's experiences with diseases, recommendations and feedbacks about certain medications or medical procedures, and emotional support in the form of encouragement, sympathy, and success stories. Mining this content can prove to be very useful in obtaining crucial insights into community dynamics such as identifying dominant health issues or the effects of social support on community members, identifying influential members, as well as designing smart information retrieval systems for users.

In this study, we focus on an online cancer support community, the Cancer Survivors Network[1] (CSN) of the American Cancer Society. We analyze user messages of CSN to identify the two most important types of social support present in them: informational and emotional support (Davison et al., 2000). Emotional support comprises of seeking or providing caring/concern, understanding, empathy, sympathy,

[1] http://www.csn.cancer.org

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 827–836, Dublin, Ireland, August 23-29 2014.

encouragement, affirmation and validation. In contrast, informational support comprises of seeking or providing knowledge such as advice, referrals, and suggestions (Bambina, 2007). We further explore the relation between the type of support present in messages and users' influence in the community.

Identifying the type of support in user messages in an OHC can potentially be used in many important applications including the following:

1. **Identify influential members in OHCs**: Every community has a set of members who influence (a much larger set of) other members in the community. These members are called leaders or influential members. The attributes of a leader in a community depends upon the community's nature (QA, Twitter, OHC, forum, blogsite, etc.). For example, high activity may not be an indicator of high influence in the blogosphere (Agarwal et al., 2008) and high popularity does not necessarily imply influence in Twitter (Romero et al., 2011). In OHCs, bringing positivity in the community and answering members' concerns effectively by posting messages that contain certain type of support (informational or emotional) may be an indicator of influence.
2. **Improve information search in OHCs**: Interactions in OHCs contain valuable information in the form of people's experiences, advice, referrals, pertaining to diseases, medications, side-effects, etc. Users embed this information often in messages containing other types of support, of which emotional support constitutes a major part. To efficiently search OHCs for this information, it must be separated from emotional support. Hence, identifying the type of support in user messages can help improve search and retrieval in OHCs.
3. **Understand social relationships in OHCs**: Emotional support is one of the dimensions of social tie strength between members in a social network (Gilbert and Karahalios, 2009). Previous studies have shown that members receiving emotional support in OHCs are more likely to remain in the community for a longer period of time as compared to members receiving informational support (Wang et al., 2012). Identifying emotional and informational support can help understand the social dynamics of an OHC. For example, it would be interesting to see if there is a correlation between the social tie strength of members and the type of support present in their interactions.

Hi X, I had a bilateral with radical on the right and prophylactic on the left. I think all you can do is gentle exercises to strengthen your back (yoga). There are also herbal painkillers that work well too. I just tolerate the pain and consider it a signal of my new limit and go down to rest. You want to talk, anytime! We are all there with you.

Table 1: A user message. Sentences in grey and black fonts are informational and emotional, respectively.

We model the task of identifying the two types of supports as a binary classification problem. Specifically, we classify each sentence in a user message as containing either emotional or informational support [2]. Table 1 shows a user message containing emotional and informational supports. We use several features computed from sentences of messages such as unigrams, part-of-speech tags, lexicon-based features and word patterns for the classification. After building the classification model, we predict the amounts of the two supports in all CSN messages and explore the following research question:

RQ: Do influential members of CSN post one of the two types of supports significantly more compared to regular members?

We analyze messages posted by regular members and messages posted by certain members, identified as *influential* by the CSN community managers and two staff members who monitor the contents of the CSN on a full time basis, for the type of support (informational and emotional) present in them. Using the classification model, we calculate the amounts of the two supports posted by influential members and regular members and compare them across the two populations (For details, see Section 3.1).

Previous works on analyzing social support in OHCs have mainly been in the field of social science (Eriksson and Lauri, 2000; Rodgers and Chen, 2005; Høybye et al., 2005; Pfeil and Zaphiris, 2007; Beaudoin and Tao, 2007; Buis, 2008; Han et al., 2011). These works used manual techniques for identifying the type of support in user messages and hence, are limited to a small number of messages as

[2] Although a sentence may belong to both the classes, we did not find such cases in our data.

compared to the real world data. In contrast, the current work builds machine learning classifiers that can automatically predict the type of support in messages. Also, to the best of our knowledge, there have been no reported works on analyzing the relationship between users' influence and the type of support present in their messages in OHCs. Next, we review related works.

2 Related Work

Many studies in social science have focused on analyzing social support in user messages of OHCs (Coursaris and Liu, 2009; Han et al., 2011; Pfeil and Zaphiris, 2007), finding impacts of social support on users (Eriksson and Lauri, 2000; Rodgers and Chen, 2005; Buis, 2008; Høybye et al., 2005), identifying information needs of users in OHCs (Rozmovits and Ziebland, 2004), etc. Among various types of social supports, emotional support and informational support have received major attention. In this section, we first review social science works on analyzing online social support, discuss works on identifying the type of social support, and, finally, compare the current problem with **subjectivity analysis**.

LaCoursiere (2001) presented an integrated theory conceptualizing online social support. She defined three channels through which online social support occurs: 1) *perceptual*: individual feeling the need of social support arising due to emotional states such as stress, etc, 2) *cognitive*: individual seeking information about certain medical entities such as procedures, medication, etc, 3) *transactional*: individual evaluating the received social support. In our case, these channels correspond to emotional support and informational support. Høybye et al. (2005) conducted a qualitative study to analyze the effects of online social support by interviewing women with breast cancer who used an online support group and found that the women were empowered by the exchanges of knowledge and experience within the online support group. Rodgers et al. (2005) conducted a longitudinal content analysis of messages of participants in a breast cancer discussion board to analyze changes in affect/sentiment of the participants towards breast cancer and found that a positive shift in sentiment occurred over the period of time. Pfeil and Zaphiris (2007) analyzed messages of SeniorNet forum to extract language patterns used to provide empathic support. Budak and Agrawal (2013) interviewed participants of group chats in Twitter and found that informational support is more important than emotional support in educational Twitter chats.

All the above works used manual methods of data preparation such as interviews with users of support groups, manual coding of messages to identify emotional and informational support and performed further qualitative and/or quantitative analyses based on that data. Since, manual methods have serious limitations in terms of scalability, the number of messages used for analysis in these studies is too small compared to the real world data which contains millions of messages. To address these limitations, we develop automatic methods for identifying the type of support in user messages in an online cancer support group using machine learning. We develop a classifier that learns on a smaller set of manually labeled messages and makes predictions on a much larger set of messages with a very high accuracy.

A recent work by Wang et al. (2012) is close to our work. They used a linear regression model to predict the amount of informational and emotional supports present in messages of a cancer forum. For a test message, the trained model predicts the amount of the two supports on a scale of $1 - 7$. Since a message may contain both types of support, it is generally difficult for human annotators to assess the amount of each support in an entire message on a particular scale for model training. In contrast, we label each sentence as belonging to either informational or emotional support class and identify the two types of support at sentence level in messages (using binary classification). Note that it is much easier and less ambiguous for a human annotator to identify the type of support present in a sentence (of a message) compared to giving a score to an entire message based on the amount of the two supports present in it.

Relationship with Subjectivity Analysis: Subjectivity analysis is an active area of research in computational linguistics. It essentially deals with separating subjective parts (e.g., expressing opinion, emotion, speculation and other private states of mind) from objective parts (presenting facts, verifiable information) of a text (Wiebe et al., 1999; Biyani et al., 2012a). It has been widely used in applications like opinion mining from product reviews (Liu, 2010), community question-answering (Li et al., 2008a; Stoyanov et al., 2005a; Somasundaran et al., 2007), summarization (Carenini et al., 2006; Seki et al., 2005), and finding opinionative threads in online forums (Biyani et al., 2014; Biyani et al., 2012b; Biyani et al., 2013a). Though the current work has some relation with subjectivity analysis in the sense that both are

text classification, there are important differences between the two problems. The two classes in subjectivity analysis (subjective and objective) are different from the two types of support that we identify. While emotional support is subjective in nature, informational support is not necessarily objective as it also contains opinions of users. Also, social support in OHCs encompasses several types of supports such as understanding, caring, concern, sympathy, empathy, knowledge about medications, etc. which are generally not provided by users in other sites such as product reviews, question-answering sites, etc. These differences make the two problems different in both the nature and the approaches that can be used to address them. For example, we use certain word patterns to identify sympathy and affirmation and use the presence of terms related to cancer medications, procedures and side-effects for computing features for classification. These features have not been used in subjectivity classification.

3 Problem Formulation

Online health communities provide social support to its members of which emotional and informational supports constitute a major part and have received major attention as compared to other supports such as companionship, community building, network support, etc. (Bambina, 2007; Meier et al., 2007; Himle et al., 1991; Wang et al., 2012; Pfeil and Zaphiris, 2007). We focus on the two supports and follow their definitions as given by Bambina (2007) in their study of social supports expressed in a cancer support group. They define *emotional* messages as the messages that have the following supports: caring/concern, understanding, empathy, sympathy, encouragement, affirmation and validation. *Informational support* is defined as providing advice, knowledge and referrals. Since a user message often contains a mixture of these supports, we identify the two supports at sentence level. Table 1 contains a user message with sentences marked with the type of support in them. Specifically, given a sentence s, in a user message, we want to classify it into one of the two classes: emotional support or informational support. We use machine learning methods for classification. After training the classifier, we use it to predict the type of support in the sentences of user messages in CSN and address our research question outlined in Section 1. We present the details of the features used for classification in Section 3.2.

3.1 Research Question

To address the research question **(RQ)**, we need to compute the amounts of the two supports in the messages of regular and influential members and then compare the two amounts. Let u denote a user and M be the set of messages posted by her such that $M = \{m_1, m_2, ...m_p\}$ where p is the total number of messages in the set M. For a message $m_k \in M$, we compute its *emotional index*, $e_{uk} = n_{ek}/(n_k)$ where n_{ek} and n_k are the number of sentences containing emotional support and the total number of sentences in m_k. Since a sentence can belong to either emotional support or informational support class, informational index of m_k, $i_{uk} = 1 - e_{uk}$. The overall emotional index of u (e_u) is the average of the emotional indices of her messages: $e_u = \frac{1}{p}\sum_{k=1}^{p} e_{uk}$. The informational index of u, $i_u = 1 - e_u$. Since, the informational index can be derived from emotional index, we compute only emotional indices for all regular and influential members and compare them between the two user populations (regular and influential). We compute the emotional indices of regular members, E_R, and emotional indices of influential members, E_I. We compare the means of the two populations of emotional indices (μ_{Re} and μ_{Ie}) and test the null hypothesis (H_0) and the alternate hypothesis (H_1) as follows:
H_0: The two populations have equal means, i.e., $\mu_{Re} - \mu_{Ie} = 0$.
H_1: The two populations have significantly different means , i.e., $\mu_{Re} - \mu_{Ie} \neq 0$.

For one of the population indices to be significantly more than the other, we should have the null hypothesis rejected. We use one-sided t-test to conduct hypothesis testing and report the results in Section 4.5. Next, we discuss the features used in the classification.

3.2 Features for Classification

3.2.1 Words and POS tags

Words and their part-of-speech tags capture basic lexical properties of text and have been extensively used in text classification problems such as subjectivity classification and sentiment classification (Li et al., 2008b; Yu and Hatzivassiloglou, 2003; Biyani et al., 2013b). We use frequency of words and their POS tags in a sentence as features in our classification model.

3.2.2 Lexicon-based Features

Emotional support expresses caring, concern, sympathy, and other kinds of sentimental support whereas informational support provides knowledge about cancer medications, cancer reports, referrals, and other kinds of information (Bambina, 2007). Due to this difference in the nature of these supports, a sentence expressing emotional support is likely to contain emotional words which are subjective in nature and a sentence containing informational support is likely to have cancer-related keywords such as drug names, names of cancer procedures, etc. To capture this difference, we use frequencies of subjective words and cancer-related keywords as features. Specifically, we design five features to code frequencies of weak subjective words (**numWeak**), strong subjective words (**numStrong**), cancer drugs (**numDrug**), side-effects of cancer medications (**numSide**), and cancer procedures (**numProc**) respectively in a sentence. We use the subjectivity lexicon compiled from the MPQA corpus (Stoyanov et al., 2005b) to get weak and strong subjective words. We compile lexicon of cancer drugs [3], and CSN staff members helped get a list of side-effects and cancer procedures. Some of the side-effects of cancer medications are hair loss, neuropathy, fatigue, fibrosis, etc.

3.2.3 Linguistic Features

We analyzed user messages to find patterns that are expressive of emotional and informational support. We found that members, generally, use certain word patterns to express similar feelings. For example, to provide affirmation and sympathy, people use positive verbs such as *know, feel, understand, sense, support*, etc. in patterns *<I $posVerb>* and *<I $aux $posVerb>*, where *$posVerb* is a positive verb and *$aux* is an auxiliary verb from the set {can, could, do, would, will, may}. Some people use *"We"* instead of *"I"* in their messages to provide support such as "**we understand** *what you are going through*". To take into account such cases, we use the same patterns by replacing *"I "* with *"We"*. Hence, we get four patterns for emotional support. For providing informational support, people often use patterns such as *<You $advice>*, *<I $opinion>*, *<I $aux $opinion>* to provide advice and opinions. *$advice* is an auxilliary verb from the set {should, must, need, might}, *$opinion* is an opinion verb from the set {recommend, advise, suggest, advocate, request}, and *$aux* is an auxilliary verb. People also give information about their experiences using patterns such as *<I too>*, *<I also>* and *<I $pastVerb>* to tell their own experiences related to similar problems as that of the support seeker where *$pastVerb* is a past tense verb such as *underwent, undergone, experienced, had, found*, etc. So, we get six patterns for informational support. We design two features (**IsEmPattern** and **IsInPattern**) to encode presence (1) or absence (0) of the two types of patterns.

For a sentence, we also use its number of words (**numWords**) and its type, question sentence (**IsQues**) and/or exclamatory sentence (**isExclaim**), as features. To identify question sentences, we see if a sentence starts with any of the 5W1H words *(what, why, who, when, where, how)* or words in the set {do, does, did} or ends with a question mark.

4 Experiments

We now describe our data and the experimental setting, and present our results.

4.1 Data Preparation

We use data from a popular online cancer support community, the Cancer Survivors' Network (CSN), developed and maintained by the American Cancer Society. CSN is an online community for cancer patients, cancer survivors, their families and friends. Its features are similar to many online forums with dynamic interactive medium such as chat rooms, discussion boards, etc. Members of CSN post in discussion boards for seeking and sharing information about cancer related issues and for seeking and providing emotional support. To conduct our experiments, we used user messages in the discussion threads of the Breast Cancer sub forum of CSN that were posted between June 2000 to June 2012. Breast cancer is the largest among all the sub-forums of CSN. A dataset of $250,868$ messages posted by 5516 users in $22,297$ discussion threads is used in this study.

To prepare the evaluation dataset for classification experiments, we randomly sampled 240 messages from 27 discussion threads. Since, our focus is on the messages that provide support, we do not consider

[3] http://www.cancer.gov/cancertopics/druginfo/alphalist

messages posted by thread starters in discussion threads as they seek support. We took help of three human annotators to tag all the sentences of all the messages in one of the two support classes. First, two annotators tagged all the sentences. The percentage agreement between them was 89%. For the remaining 11% sentences, majority vote was taken with the help of the third annotator. Following this tagging scheme, we obtained a total of 1066 sentences with 390 sentences in the informational support class and 676 sentences in the emotional support class. In many cases, members only write a few words, e.g., see you, bye, or their names at the end of a message. To deal with these situations, we filter out sentences that have less than four words.

4.2 Experimental Protocol

We experimented with various machine learning algorithms (Naive Bayes, Support Vector Machines, Logistic Regression, Bagging, Boosting, etc.) to conduct our classification experiments. Naive Bayes Multinomial gave the best performance with words & POS tags features, logistic regression with lexicon-based features and AdaBoost (with Decision Stump as the weak learner) with linguistic features. For combining the models built on the three types of features, we used the following three methods:

1. **Feature combination:** Classification model built on the feature set generated by combining the three types of features. It is denoted by **All**. We use Multinomial Naive Bayes for this model.
2. **Average confidence:** Ensemble of the three classifiers built on the three types of features respectively. The final confidence of the ensemble is calculated by taking average of the confidences outputted by the three classifiers. It is denoted by **AllAvgConf**.
3. **Highest confidence:** Similar to the **AllAvgConf** model but the final prediction of the ensemble is taken as the prediction of the most confident classifier of the three classifiers. More precisely, the prediction for an instance is given by the classifier that returns the maximum prediction confidence for one class or the other. It is denoted by **AllMostConf**.

We used Weka data mining toolkit (Hall et al., 2009) to conduct classification experiments. To evaluate the performance of our classifiers, we used macro-averaged precision, recall and F-1 score. We use F-1 score to compare performances of two classifiers and used 10-fold cross validation. A naive baseline that classifies all the instances in the majority class will have a macro-averaged precision, recall and F-1 score of 0.402, 0.634 and 0.492, respectively.

4.3 Classification Results

Table 2 presents the results of the support classification experiments. The table reports precision, recall and F-1 score of different classification models for the individual classes and the overall result. Words & POS tags are the best performing features followed by lexicon-based features and linguistic features. Further, combining all the features (model denoted as "**All**") improves the performance over individual feature types for both classes. We see that **AllMostConf** model is the best performing of all the models, particularly outperforming **All** and **AllAvgConf** models. This observation suggests that the three classifiers built on the three features types have different knowledge. For some instances, a particular classifier is more confident than the rest while for other instances, other classifiers are more confident. Hence, we see that taking prediction of the most

Model	Precision	Recall	F-1
Emotional support class			
Words & POS tags	0.855	0.858	0.857
Lexicon-based features	0.722	0.836	0.775
Linguistic features	0.698	0.837	0.761
All	0.862	0.861	0.862
AllAvgConf	0.848	0.893	0.87
AllMostConf	0.851	0.911	**0.88**
Informational support class			
Words & POS tags	0.753	0.749	0.751
Lexicon-based features	0.608	0.441	0.511
Linguistic features	0.569	0.372	0.45
All	0.76	0.762	0.761
AllAvgConf	0.797	0.723	0.758
AllMostConf	0.825	0.723	**0.77**
Overall			
Words & POS tags	0.818	0.818	0.818
Lexicon-based features	0.68	0.691	0.678
Linguistic features	0.651	0.667	0.647
All	0.825	0.825	0.825
AllAvgConf	0.829	0.830	0.83
AllMostConf	0.841	0.842	**0.84**

Table 2: Classification results.

confident classifier gives the best performance. It is interesting to note that combining the three classifiers' knowledge in this fashion is more effective than simply combining all the three types of features and train a single classifier on the combined feature set. We also note that all the models have better performance for the emotional support class than for the informational support class. This can be caused by the fact that there are significantly more number of instances in the former class and, hence, more patterns to learn for the class.

4.4 Informative Features

Next, we study the importance of individual features by measuring their chi-squared statistic with respect to the class variable. We, first, study the word features and then present rankings of the other types of features. Figure 1 shows a cloud of top 26 most informative words. The size of a word is proportional to its chi-squared statistic, i.e., bigger a word, more informative it is. We see that cancer specific keywords such as herceptin, tamoxifen, chemo, dose, stage, etc and words conveying emotions such as good, hope, glad, pain, hugs, etc are highly informative for the support classification. Since, chi-square method gives feature ranking for the class variable and not for individual classes, we compute word rankings for individual classes using $tf - idf$ scores of words. Specifically, for a term t and a class c, we compute the term frequency of t by counting its number of occurrences in the instances (sentences) belonging to c and multiply the term frequency with the inverse document frequency of t (calculated from the entire corpus) to get the $tf - idf$ score of t for c. Using this method, we calculated $tf - idf$ scores for all the words and ranked them according to their scores for the two classes. Figure 2 shows top ten $tf - idf$ ranked keywords for the two classes. We see that cancer-related keywords and words expressing emotions are among the top ten most informative words for the informational and the emotional support classes respectively. We also note that most of the top ten words for the two classes in Figure 2 are in the word cloud of the top 26 words computed using chi-squared method except "keep" for the emotional support class and "after", "first", "because" and "cancer" for the informational class. These words have semantic relationships with the classes. For example, "keep" is often used by support providers in phrases such as "**keep** *you in prayers*", "*may god* **keep** *you in good health*", etc to provide emotional support and "after" and "first" are used in the context of providing one's own experience related to cancer procedures, medications, etc such as "**After** *my* **first** *chemo, I did not feel light*".

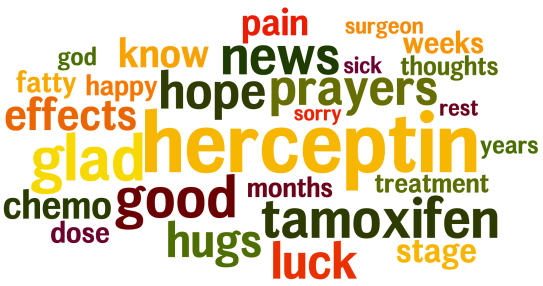


Figure 1: Top 26 words ranked by Chi-squared test.

Emotional support	Informational support
good	chemo
know	after
glad	radiation
news	first
hope	herceptin
keep	treatment
prayers	tamoxifen
luck	cancer
hugs	because
better	pain

Figure 2: Top ten words for the two classes ranked using tf-idf scheme.

We, now, discuss the ranking of non-word features: POS tags, lexicon-based and linguistic features. The chi-squared ranking for the lexicon-based and linguistic features is as follows: numStrong > numWords > isExclaim > numDrug > numSide > numProc > isInPattern > isEmPattern > numWeak > isQues. The features on the right side of > have higher rank than those on the left side. We see that the number of strong subjective words in a sentence is the most informative feature followed by number of words in a sentence. Among cancer-related terms, drug names are more informative than side-effects and cancer procedures. Also, informational support word patterns are more informative than word patterns capturing emotional support. It is interesting to note that isQues is the least informative feature, maybe due to the fact that, while providing support, people generally do not ask questions. The top 5

most informative POS tags are: cardinal number (CD) followed by singular noun (NN), participle verb (VBN), past tense verb (VBD) and preposition (IN).

4.5 Influence versus Support type

CSN managers provided a list of 62 influential members (IMs) for the breast cancer forum. IMs posted a total of $340,147$ sentences in $85,244$ messages and regular members posted $825,651$ sentences in $165,624$ messages in the breast cancer forum. As described in Section 3, we conduct statistical hypothesis testing on the two populations of emotional indices (regular members and IMs) to understand if there is a significant difference in their posting behaviors in terms of providing one of the two supports more often than the other. To test our hypothesis, we conducted one sided t-test on the two populations. We found that the mean of emotional indices of IMs (0.713) is significantly larger than that of the regular members (0.542). We also note that the posting behavior of regular members in CSN follows a power law distribution with most of the members posting very few messages ($mode = 1, median = 2, mean = 30$) and only a few members posting very many messages. To verify that this behavior does not have impacts on our hypothesis testing, we conducted three more t-tests between the two populations using a threshold on the number of messages that a member has posted. We used three threshold values on the number of messages: 1, 2, and 30 (as mode, median and mean values). For all the three t-tests, the null hypothesis was rejected at p-value< 0.001, suggesting that IMs posted significantly more emotional support than regular members. The values of Mean Emotional Indices corresponding to the three thresholds are 0.715, 0.719 and 0.746 for influential members and 0.564, 0.581 and 0.646 for regular members respectively.

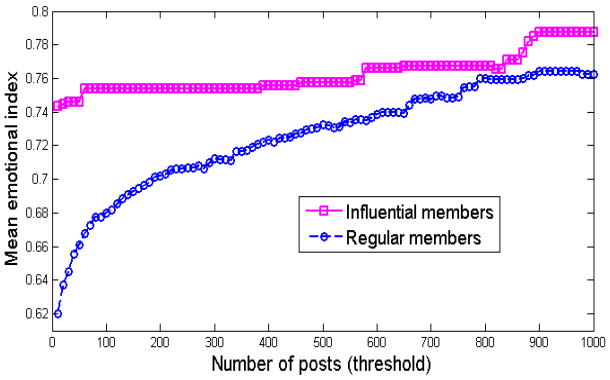


Figure 3: Plot showing the change in mean emotional indices of influential members (pink) and regular members (blue) with the threshold on the number of messages posted by them.

In our analysis, we observed an interesting behavior. As we increased the threshold, the mean of emotional indices also increased. To further investigate this finding, we plotted the means of emotional indices of regular members and IMs as the function of the threshold on the number of messages posted by them. We increased the threshold from 10 to 1000 in steps of 10. Figure 3 reports the finding. We see that the mean of emotional indices of regular members increase with the threshold suggesting that more active members post more emotional support as compared to the less active members. We also see that the mean of emotional indices of IMs is higher than that of regular members for all the thresholds. These interesting observations can be helpful in analyzing behavior of influential members in OHCs.

5 Acknowledgments

We would like to thank Iulia Bivolaru for her help with data preparation. This material is based upon work supported by the National Science Foundation under Grant No. 0845487.

6 Conclusion and Future Work

We identified two types of social support, emotional and informational, provided in user messages of an online cancer support community using machine learning classification models. We used three types of features and got the best results by using ensemble of the three classifiers built on the three individual feature types. Our models achieved strong results with over 80% F-1 score. We also found that influential members provide significantly more emotional support to the community as compared to regular members. The finding can be helpful in identifying properties of influential members in online health communities. In future, we plan to analyze effects of the two types of supports on OHCs' dynamics and use it to improve information search in OHCs.

References

Nitin Agarwal, Huan Liu, Lei Tang, and Philip S Yu. 2008. Identifying the influential bloggers in a community. In *Proceedings of the 2008 international conference on web search and data mining*, pages 207–218. ACM.

Antonina Bambina. 2007. *Online social support: The interplay of social networks and computer-mediated communication*. Cambria press.

Christopher E Beaudoin and Chen-Chao Tao. 2007. Benefiting from social capital in online support groups: An empirical study of cancer patients. *CyberPsychology & Behavior*, 10(4):587–590.

Prakhar Biyani, Sumit Bhatia, Cornelia Caragea, and Prasenjit Mitra. 2012a. Thread specific features are helpful for identifying subjectivity orientation of online forum threads. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 295–310.

Prakhar Biyani, Cornelia Caragea, Amit Singh, and Prasenjit Mitra. 2012b. I want what i need!: analyzing subjectivity of online forum threads. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 2495–2498.

Prakhar Biyani, Cornelia Caragea, and Prasenjit Mitra. 2013a. Predicting subjectivity orientation of online forum threads. In *Proceedings of the 14th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 109–120.

Prakhar Biyani, Cornelia Caragea, Prasenjit Mitra, Chong Zhou, John Yen, Greta E Greer, and Kenneth Portier. 2013b. Co-training over domain-independent and domain-dependent features for sentiment analysis of an online cancer support community. In *ASONAM*, pages 413–417. ACM.

Prakhar Biyani, Sumit Bhatia, Cornelia Caragea, and Prasenjit Mitra. 2014. Using non-lexical features for identifying factual and opinionative threads in online forums. *Knowledge-Based Systems*.

Ceren Budak and Rakesh Agrawal. 2013. On participation in group chats on twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 165–176. International World Wide Web Conferences Steering Committee.

Lorraine R Buis. 2008. Emotional and informational support messages in an online hospice support community. *Computers Informatics Nursing*, 26(6):358–367.

G. Carenini, R. Ng, and A. Pauls. 2006. Multi-document summarization of evaluative text. In *EACL*, pages 305–312.

Constantinos K Coursaris and Ming Liu. 2009. An analysis of social support exchanges in online hiv/aids self-help groups. *Computers in Human Behavior*, 25(4):911–918.

Kathryn P Davison, James W Pennebaker, and Sally S Dickerson. 2000. Who talks? the social psychology of illness support groups. *American Psychologist*, 55(2):205.

Christine Dunkel-Schetter. 1984. Social support and cancer: Findings based on patient interviews and their implications. *Journal of Social Issues*, 40(4):77–98.

Elina Eriksson and Sirkka Lauri. 2000. Informational and emotional support for cancer patients relatives. *European Journal of Cancer Care*, 9(1):8–15.

Eric Gilbert and Karrie Karahalios. 2009. Predicting tie strength with social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 211–220. ACM.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. 2009. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Jeong Yeob Han, Dhavan V Shah, Eunkyung Kim, Kang Namkoong, Sun-Young Lee, Tae Joon Moon, Rich Cleland, Q Lisa Bu, Fiona M McTavish, and David H Gustafson. 2011. Empathic exchanges in online cancer support groups: distinguishing message expression and reception effects. *Health communication*, 26(2):185–197.

David P Himle, Srinika Jayaratne, and Paul Thyness. 1991. Buffering effects of four social support types on burnout among social workers. In *Social Work Research and Abstracts*, volume 27, pages 22–27. Oxford University Press.

Mette Terp Høybye, Christoffer Johansen, and Tine Tjørnhøj-Thomsen. 2005. Online interaction. effects of storytelling in an internet breast cancer support group. *Psycho-Oncology*, 14(3):211–220.

Sheryl Perreault LaCoursiere. 2001. A theory of online social support. *Advances in Nursing Science*, 24(1):60–77.

B. Li, Y. Liu, A. Ram, E.V. Garcia, and E. Agichtein. 2008a. Exploring question subjectivity prediction in community qa. In *SIGIR*, pages 735–736. ACM.

Baoli Li, Yandong Liu, and Eugene Agichtein. 2008b. Cocqa: co-training over questions and answers with an application to predicting question subjectivity orientation. In *EMNLP '08*, pages 937–946.

B. Liu. 2010. Sentiment analysis and subjectivity. *Handbook of Natural Language Processing,*, pages 978–1420085921.

Diane Maloney-Krichmar and Jenny Preece. 2005. A multilevel analysis of sociability, usability, and community dynamics in an online health community. *TOCHI*, 12(2):201–232.

Andrea Meier, Elizabeth J Lyons, Gilles Frydman, Michael Forlenza, and Barbara K Rimer. 2007. How cancer survivors provide support on cancer-related internet mailing lists. *Journal of Medical Internet Research*, 9(2).

Ulrike Pfeil and Panayiotis Zaphiris. 2007. Patterns of empathy in online communication. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 919–928. ACM.

Baojun Qiu, Kang Zhao, P. Mitra, Dinghao Wu, C. Caragea, J. Yen, G.E. Greer, and K. Portier. 2011. Get online support, feel better – sentiment analysis and dynamics in an online cancer survivor community. In *SocialComm' 11*, pages 274–281.

Shelly Rodgers and Qimei Chen. 2005. Internet community group participation: Psychosocial benefits for women with breast cancer. *Journal of Computer-Mediated Communication*, 10(4):00–00.

Daniel M Romero, Wojciech Galuba, Sitaram Asur, and Bernardo A Huberman. 2011. Influence and passivity in social media. In *Machine learning and knowledge discovery in databases*, pages 18–33. Springer.

Linda Rozmovits and Sue Ziebland. 2004. What do patients with prostate or breast cancer want from an internet site? a qualitative study of information needs. *Patient education and counseling*, 53(1):57–64.

Y. Seki, K. Eguchi, N. Kando, and M. Aono. 2005. Multi-document summarization with subjectivity analysis at duc 2005. In *DUC*. Citeseer.

S. Somasundaran, T. Wilson, J. Wiebe, and V. Stoyanov. 2007. Qa with attitude: Exploiting opinion type analysis for improving question answering in on-line discussions and the news. In *ICWSM*.

V. Stoyanov, C. Cardie, and J. Wiebe. 2005a. Multi-perspective question answering using the opqa corpus. In *EMNLP 2005*, pages 923–930. ACL.

Veselin Stoyanov, Claire Cardie, and Janyce Wiebe. 2005b. Multi-perspective question answering using the opqa corpus. In *HLT-EMNLP '05*, HLT '05, pages 923–930, Stroudsburg, PA, USA. ACL.

Ruvanee P Vilhauer. 2009. Perceived benefits of online support groups for women with metastatic breast cancer. *Women & health*, 49(5):381–404.

Yi-Chia Wang, Robert Kraut, and John M Levine. 2012. To stay or leave?: the relationship of emotional and informational support to commitment in online health support groups. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pages 833–842. ACM.

J.M. Wiebe, R.F. Bruce, and T.P. O'Hara. 1999. Development and use of a gold-standard data set for subjectivity classifications. In *ACL*, pages 246–253. ACL.

Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 129–136. Association for Computational Linguistics.

Identifying Emotion Labels from Psychiatric Social Texts Using Independent Component Analysis

Liang-Chih Yu[1,2] and **Chun-Yuan Ho**[1]
[1]Department of Information Management
[2]Innovation Center for Big Data and Digital Convergence
Yuan Ze University, Chung-Li, Taiwan
lcyu@saturn.yzu.edu.tw, s986304@mail.yzu.edu.tw

Abstract

Accessing the web has been an efficient and effective means to acquire self-help knowledge when suffering from depressive problems. Many mental health websites have developed community-based services such as web forums and blogs for Internet users to share their depressive problems with other users and health professionals. Other users or health professionals can then make recommendations in response to these problems. Such communications produce a large number of documents called psychiatric social texts containing rich emotion labels representing different depressive problems. Automatically identify such emotion labels can make online psychiatric services more effective. This study proposes a framework combining *latent semantic analysis (LSA)* and *independent component analysis (ICA)* to extract concept-level features for emotion label identification. LSA is used to discover latent concepts that do not frequently occur in psychiatric social texts, and ICA is used to extract independent components by minimizing the term dependence among the concepts. By combining LSA and ICA, more useful latent concepts can be discovered for different emotion labels, and the dependence between them can also be minimized. The discriminant power of classifiers can thus be improved by training them on the independent components with minimized term overlap. Experimental results show that the use of concept-level features yielded better performance than the use of word-level features. Additionally, combining LSA and ICA improved the performance of using each LSA and ICA alone.

1 Introduction

Sentiment analysis has been successfully applied for many applications (Picard, 1997; Pang and Lee, 2008; Calvo and D'Mello, 2010; Liu, 2012; Johansson and Moschitti, 2013; Balahur et al., 2014). Analysis of online psychiatric or mental health texts (Wu et al., 2005; Yu et al., 2009) is also an emerging field that could benefit from sentiment analysis techniques because more and more people search for help from the web when they suffered from depressive problems, which boost the development of online community-based services for Internet users to share their depressive problems with other users and health professionals. Through these services, individuals can describe their depressive symptoms via web forums and blogs. Other users or health professionals can then make recommendations in response to these problems. Figure 1 shows an example psychiatric social text collected from PsychPark (http://www.psychpark.org), a virtual psychiatric clinic, maintained by a group of volunteer professionals belonging to the Taiwan Association of Mental Health Informatics (Bai et al., 2001; Lin et al., 2003).

This example shows a subject's depressive problems and the responses recommended by the experts. Some meaningful tags called emotion labels herein are also annotated by the experts to indicate which categories the text belongs to. These emotion labels are useful information and can make online psychiatric services more effective. For instance, psychiatric retrieval systems are able to retrieve relevant documents according to the depressive problems (emotion labels) described in user queries so that the

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 837–847, Dublin, Ireland, August 23-29 2014.

User Problem:	
I broke up with my dear but cruel boyfriend recently. Since then, I have often felt like crying out of nowhere, and I feel pain every day. Also, it takes me a long time to fall asleep at night. So, I think that continuing to live like this is meaningless	 <Depression> <Insomnia> <Suicide>
Recommendation:	
Feeling this way is normal when going through these kinds of struggles, but over time your emotions should level out. Suicide doesn't solve anything; think about how it would affect your family. It's only when we learn to face our despair that we can learn the value of life, and also how to help other people. There are a few things you can try to help you get to sleep at night, like doing some light exercise in the evening, drinking warm milk, and listening to relaxing music; all of these can be conducive to sleep. If you still have trouble dealing with the pain, and you feel as if your mood is getting worse, it wouldn't hurt to get seek help from a healthcare professional, who can help you work through your emotions.	
Emotion Label: <Depression>, <Insomnia>, <Suicide>	

Figure 1. Example of a psychiatric social text.

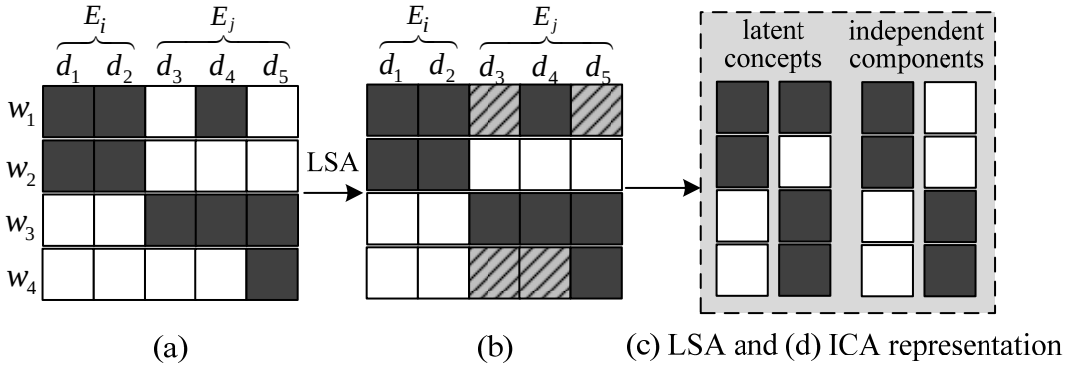


Figure 2. Comparison of LSA and ICA for feature representation.

users can learn self-help knowledge from the responses. Therefore, this study aims to identify emotion labels from psychiatric social texts. We cast this problem into a multi-label text classification task because a psychiatric social text may contain multiple emotion labels. Additionally, we propose the use of concept-level features to build classifiers instead of using surface-level features such as words, n-grams and dependency structure commonly used in the previous studies (Naughton et al., 2008; Chitturi and Hansen, 2008; Li and Zong, 2008; Kessler and Schütze, 2012; Post and Bergsma, 2013; Yu et al., 2011).

In extraction of concept-level features, latent semantic analysis (LSA) (Landauer et al., 1998) has been demonstrated its effectiveness in exploring the latent structure from a collection of documents. It uses singular value decomposition (SVD) (Golub and Van Loan, 1996) to discover latent features that do not frequently occur in the documents through the indirect associations between words and documents. Figure 2 shows an example. The original matrix, as shown in Figure 2(a), is built using five documents with two different emotion labels E_i and E_j. Suppose that the words w_1, w_2 are the useful features for E_i, and w_3, w_4 are useful for E_j, but w_4 is a latent feature because it does not frequently occur in the documents of E_j. After applying SVD, the latent features can be identified by replacing the zero entries in the original matrix with non-zero real values through the indirect associations between words and documents. For instance, w_4 originally does not occur in d_3 and d_4, but it does co-occur with w_3 in the matrix (e.g., in d_5), which means that w_4 might also occur in the documents where w_3 occurs

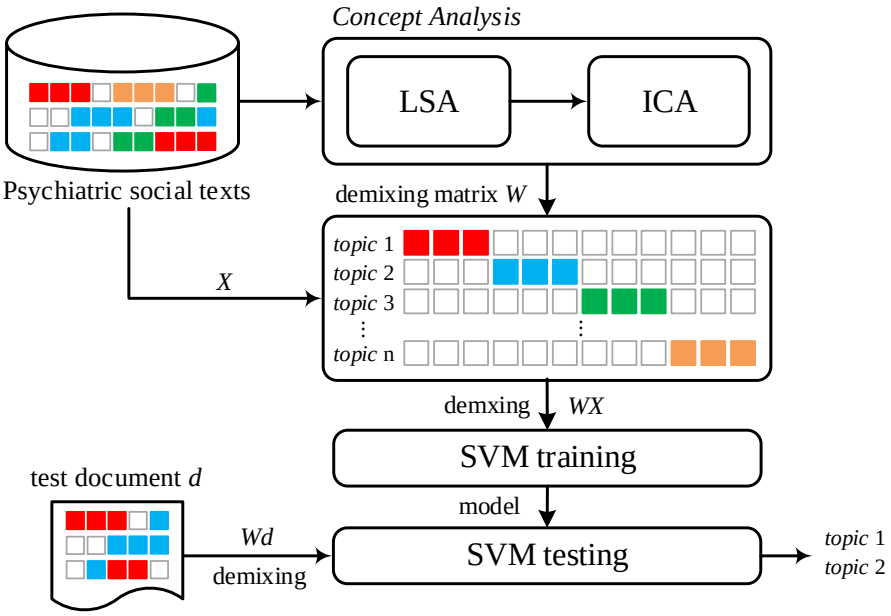


Figure 3. Framework of emotion label identification.

(e.g., d_3 and d_4). Therefore, the zero entries (w_4, d_3) and (w_4, d_4) are replaced with a non-zero value through the indirect associations between of w_3 and w_4 in d_5, as shown in Figure 2(b). This helps identify a useful latent feature w_4 for E_j. However, identifying latent features through the indirect associations cannot avoid feature overlap when different emotion labels share common words. For instance, in Figure 2(a), w_1, which is useful for E_i, still occurs in the document of E_j (e.g., d_4). Through the indirect associations between of w_1 and w_3 in d_4, the frequency of w_1 increases in the document of E_j because it may also occur in the documents where w_3 occurs (e.g., d_3 and d_5), as shown in Figure 2(b). Therefore, when all word features are to be accommodated in a low-dimensional space reduced by SVD, term overlap may occur between the latent concepts. As indicated in Figure 2(c), the two sample latent concepts which contribute to two different emotion labels share a common feature w_1. Classifiers trained on such latent vectors with term overlap may decrease classification performance.

To reduce the term overlap among concepts, we used the *independent component analysis* (ICA) (Lee, 1998; Hyvärinen et al., 2001; Naik and Kumar, 2011) because it can extract independent components from a mixture of signals and has been used in various text applications (Kolenda and Hansen, 2000; Rapp, 2004; Honkela et al., 2010; Yu and Chien, 2013). For our task, the psychiatric social texts are a mixture of emotion labels, which can be separated by ICA to obtain a set of independent components (concepts) with minimized term dependency for different emotion labels. Instead of using ICA alone, we propose a framework combining LSA and ICA for emotion label identification. The LSA is used to discover latent features that do not frequently occur in psychiatric texts, and ICA is used to further minimize the dependence of the latent features such that overlapped features can be removed, as presented in Figure 2(d). Based on this combination, the proposed framework can discover more useful latent features for different emotion labels, and the dependence between them can also be minimized. The discriminant power of classifiers can thus be improved by training them on the independent components with minimized term overlap. In experiments, we evaluate the proposed method to determine whether the use of concept-level features could improve the classification performance, and determine whether the combination method could improve the performance of using each LSA and ICA alone.

The rest of this paper is organized as follows. Section 2 describes the overall framework including LSA and ICA for emotion label identification. Section 3 summarizes comparative results. Conclusions are finally drawn in Section 4.

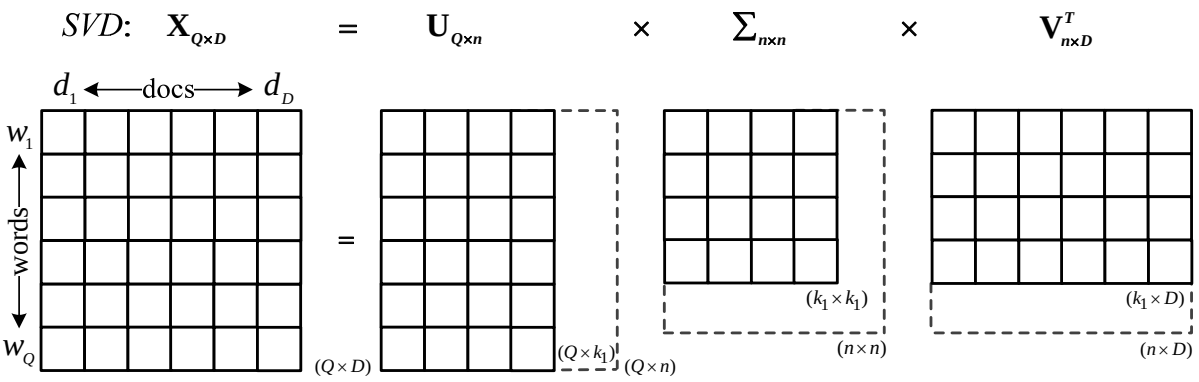


Figure 4. Illustrative example of singular value decomposition for latent semantic analysis.

2 Framework of Emotion Label Identification

Figure 3 shows the overall framework for emotion label identification. A corpus of psychiatric social texts with annotation of emotion labels are first collected from the web. This corpus which is a mixture of different emotion labels is then sequentially analyzed by LSA and ICA to generate a demixing matrix composed of a set of concepts with minimized term dependency for different emotion labels. The demixing matrix is used to separate the psychiatric social texts with mixed emotion labels into independent components for building a support vector machine (SVM) classifier. The classifier can then be benefit from the independent components to identity multiple emotion labels contained in each test example.

2.1 Latent Semantic Analysis (LSA)

LSA is a technique for analyzing the relationships between words and documents. For our task, LSA is used to identify useful latent concepts for emotion labels through indirect associations between words and documents. The first step in LSA is to build a word-by-document matrix from a corpus of psychiatric texts with different emotion labels, as shown in the sample matrix **X** in Figure 4.

The columns in $\mathbf{X}_{Q\times D}$ represent D psychiatric texts in the corpus, and the rows represent Q distinct words occurring in the corpus. Singular value decomposition (SVD) is then used to decompose the matrix $\mathbf{X}_{Q\times D}$ into three matrices as follows:

$$\mathbf{X}_{Q\times D} = \mathbf{U}_{Q\times n} \sum_{n\times n} \mathbf{V}^T_{n\times D}, \tag{1}$$

where **U** and **V** respectively consist of a set of latent vectors of words and documents, $\sum$ is a diagonal matrix of singular values, and $n = \min(Q, D)$ denotes the dimensionality of the latent semantic space. Each element in **U** represents the weight of a word, and the higher-weighted words are the useful features for the emotion labels. By selecting the largest k_1 ($\leq n$) singular values together with the first k_1 columns of **U** and **V**, the word and documents can be represented in a low-dimensional latent semantic space. The matrix $\mathbf{V}^T_{n\times D}$ can then represented with the reduced dimensions, as shown in Eq. (2).

$$\mathbf{V}^T_{k_1\times D} = \sum^{-1}_{k_1\times k_1} \mathbf{U}^T_{k_1\times Q} \mathbf{X}_{Q\times D}, \tag{2}$$

In SVM training and testing, each input psychiatric text first transformed into the latent semantic representation as follows:

$$\hat{\mathbf{t}}_{k_1\times 1} = \sum^{-1}_{k_1\times k_1} \mathbf{U}^T_{k_1\times Q} \mathbf{t}_{Q\times 1}, \tag{3}$$

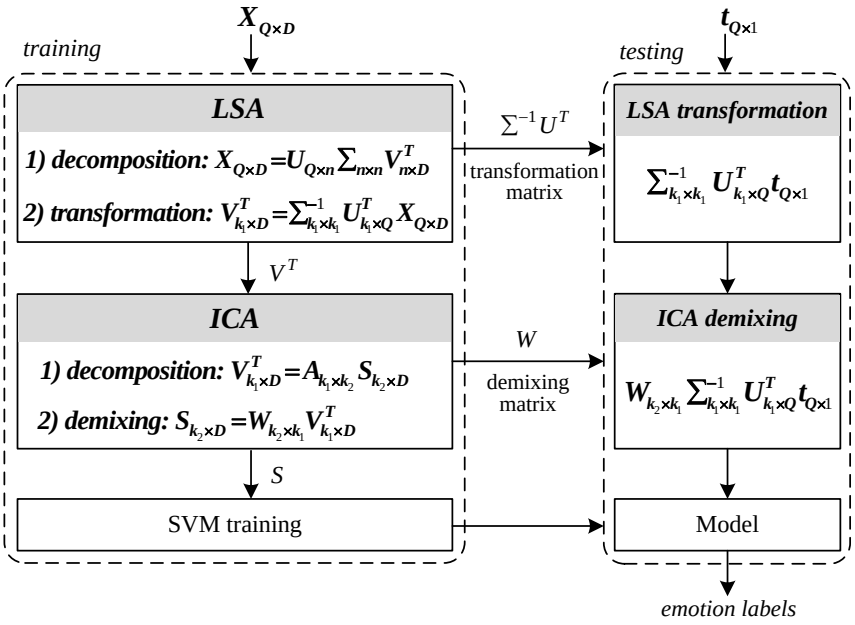


Figure 5. ICA-based method for emotion label identification.

where $\mathbf{t}_{Q\times 1}$ denotes the vector representation of an input instance, and $\hat{\mathbf{t}}_{k_1\times 1}$ denotes the transformed vector in the latent semantic space. An SVM classifier is then trained with the transformed training vectors.

2.2 Independent Component Analysis (ICA)

ICA is a technique for extracting independent components from a mixture of signals and has been successfully applied to solve the blind source separation problem (Saruwatari et al., 2006; Chien and Hsieh, 2012). The ICA model can be formally described as

$$\mathbf{X} = \mathbf{AS} \tag{4}$$

where $\mathbf{X}$ denotes the observed mixture signals, $\mathbf{A}$ denotes a mixing matrix, and $\mathbf{S}$ denotes the independent components. The goal of ICA is to estimate both $\mathbf{A}$ and $\mathbf{S}$. Once the mixing matrix $\mathbf{A}$ is estimated, the demixing matrix can be obtained by $\mathbf{W} = \mathbf{A}^{-1}$, and Eq. (4) can be re-written as

$$\mathbf{S} = \mathbf{WX} \tag{5}$$

That is, the observed mixture signals can be separated into independent components using the demixing matrix. For our problem, psychiatric texts can be considered as a mixture of signals because each of them may contain multiple emotion labels. Therefore, ICA used herein is to estimate the demixing matrix so that it can separate the psychiatric texts with mixed emotion labels to derive the independent components for each emotion label. Figure 5 shows the diagram of the proposed method.

2.1.1 LSA decomposition and transformation

In the training phase, the original matrix $\mathbf{X}_{Q\times D}$ is first processed by SVD using Eq. (1) and (2) Useful latent features that do not frequently occur in the original matrix can thus be discovered in this step.

2.1.2 ICA decomposition and demixing

The matrix $\mathbf{V}^T_{k_1\times D}$ decomposed by SVD is then passed to ICA to estimate the demixing matrix. ICA accomplishes this by decomposing $\mathbf{V}^T_{k_1\times D}$ using Eq. (6). Figure 6 shows an example of the decomposition.

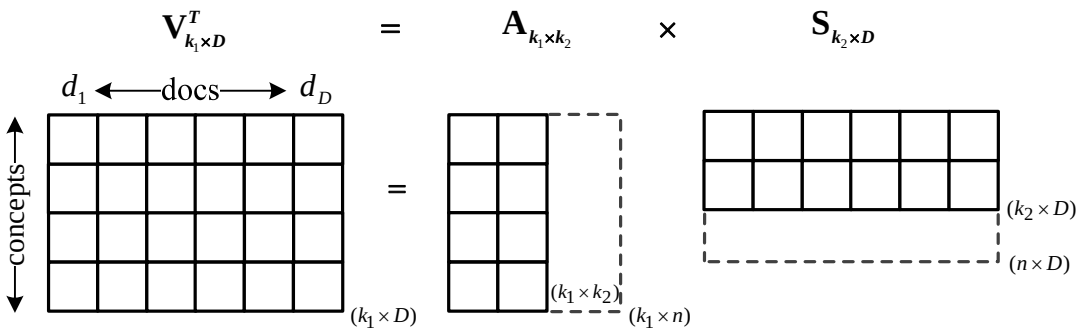


Figure 6. Illustrative example of ICA decomposition.

$$\mathbf{V}^{T}_{k_1 \times D} = \mathbf{A}_{k_1 \times k_2} \mathbf{S}_{k_2 \times D}. \tag{6}$$

Based on this decomposition, the demixing matrix can be obtained by $\mathbf{W}_{k_2 \times k_1} = \mathbf{A}^{-1}_{k_1 \times k_2}$, where k_2 ($\leq k_1$) is the number of independent components. The demixing matrix is then used to separate $\mathbf{V}^{T}_{k_1 \times D}$ to derive the independent components as follows:

$$\mathbf{S}_{k_2 \times D} = \mathbf{W}_{k_2 \times k_1} \mathbf{V}^{T}_{k_1 \times D}, \tag{7}$$

An SVM classifier is then trained with the independent components $\mathbf{S}_{k_2 \times D}$, as shown in Figure 5. In testing, each test instance $\mathbf{t}_{Q \times 1}$ is transformed using both LSA and ICA, and then predicted with the trained SVM model.

3 Experimental Results

3.1 Experiment Setup

3.1.1 Data

The data set used for experiments included 1,711 Chinese psychiatric social texts collected from the PsychPark. Each psychiatric social text was manually annotated with an emotion label by a group of volunteer mental health professionals. Table 1 shows the proportions of the emotion labels in the corpus. In calculating the proportion of each emotion label, a psychiatric social text was counted for multiple emotion labels depending on the number of emotion labels contained in it. In evaluation, 20% of psychiatric social texts in the corpus were randomly selected as a test set, and the remaining 80% were used for training.

No.	Emotion Label	Proportion
1	Depression	35.26%
2	Drug	13.38%
3	Insomnia	5.79%
4	Mood	30.04%
5	OCD (Obsessive compulsive disorder)	4.51%
6	Schizophrenia	5.36%
7	Social Anxiety	5.65%

Table 1. Distribution of emotion labels in experimental data

3.1.2 Classifiers

The classifiers involved in this experiment included PureSVM, LSA, ICA, and LSA+ICA. The PureSVM was trained on word-level features, and the others were trained on concept-level features derived using LSA, ICA, and combination of them, respectively. The implementation details for each classifier are as follows:

- **PureSVM**: An SVM classifier trained with bag-of-words features.
- **LSA:** An SVM classifier trained with the latent vectors obtained from the word-by-document matrix built from the training corpus.
- **ICA:** An SVM classifier trained with the independent components obtained by demixing the word-by-document matrix built from the training corpus.
- **LSA+ICA:** An SVM classifier trained with the independent components obtained by demixing the word-by-document matrix produced by LSA.

To identify multiple emotion labels contained in test examples, each emotion label presented in Table 1 was trained a binary classifier in the training phase. That is, for each method presented above, we built seven binary classifiers so that they can output multiple positive results to indicate that a test example contained multiple emotion labels.

3.1.3 Evaluation Metrics

The metrics used for performance evaluation included *recall*, *precision*, and *F-measure*, respectively. Recall was defined as the number of emotion labels correctly identified by the method divided by the total number of emotion labels in the test set. Precision was defined as the number of emotion labels correctly identified by the method divided by the number of emotion labels identified by the method. The F-measure (F1) was defined as $2 \times recall \times precision$ / (*recall* + *precision*).

3.2 Evaluation of LSA and ICA

This experiment compared the performance of LSA and ICA using different settings for the parameters k_1 and k_2, which respectively represent the dimensionality of the latent semantic space and the

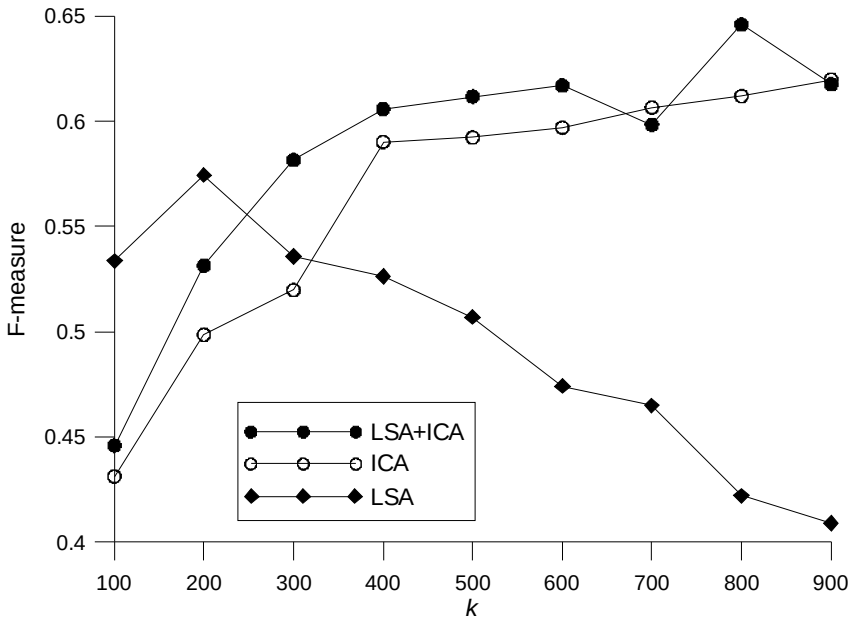


Figure 7. Performance of the LSA, ICA and LSA+ICA, as a function of *k*.

number of independent components. Figure 7 shows the F-measure of LSA, ICA, and combination of them with the setting $k = k_1 = k_2$. The F-measure is the average F-measure over the seven emotion labels. The results show that the optimal settings of LSA was *k*=200. The performance of LSA dropped dramatically as *k*>200, indicating that most useful latent features were discovered within the first 200 concepts and the remaining concepts may contain noisy features, thus reducing performance. The respective optimal settings for ICA and LSA+ICA were *k*=900 and *k*=800. In addition, both ICA and LSA+ICA outperformed LSA for most settings of *k*. The best settings of the parameters were used in the following experiments.

3.3 Comparative Results

This section reports the classification performance of PureSVM, LSA, ICA, and LSA+ICA. Table 2 shows the comparative results. Compared to the use of word-level features (i.e., PureSVM), LSA, ICA, and LSA+ICA achieved a higher F-measure. Additionally, LSA yielded a much greater recall than did PureSVM, whereas ICA yielded much greater precision. These findings indicate that the concept-level features are useful for emotion label identification. Among the three concept-based methods, LSA can discover latent concepts for emotion labels, whereas ICA can extract independent components that can minimize the term dependence within them. The results show that ICA yielded higher recall and F-measure but lower precision than did LSA. By combining LSA and ICA, the performance was improved on all measures because LSA+ICA can not only discover latent concepts but also minimize term overlap among the concepts.

Another observation is that the emotion label Depression yielded the highest F-measure while both OCD and Schizophrenia yielded the lowest. One possible reason for these results is the distribution of emotion labels in the test set (e.g., Depression and Mood are the major classes). However, the skewed distribution was just a minor factor. For example, the test set included four small classes (Insomnia, OCD, Schizophrenia and Social Anxiety) with similar proportions (5.79%, 4.51%, 5.36% and 5.65%), but their F-measures were quite different (70%, 57%, 57% and 64%). Terms overlap emotion labels could have a significant impact on classification performance. For example, Insomnia had a much higher classification performance than the other three minor classes because the words used in this class were quite distinct from those used for other classes. Conversely, the words used for OCD and Social Anxiety overlapped significantly, thus yielding lower performance. Table 3 shows some representative words (with higher weights) in the independent components for the emotion labels.

Class	PureSVM			LSA			ICA			LSA+ICA		
	R	P	F	R	P	F	R	P	F	R	P	F
Depression	58	59	59	68	74	71	72	75	73	73	78	75
Drug	60	38	47	57	71	63	51	69	59	55	72	62
Insomnia	53	66	59	49	76	60	65	76	70	66	75	70
Mood	63	48	54	61	56	58	65	59	62	67	61	64
OCD	58	39	47	53	53	53	53	53	53	56	59	57
Schizophrenia	63	23	34	56	64	60	56	47	51	58	57	57
Social Anxiety	34	40	37	24	78	37	52	71	60	56	74	64
Avg.	**56**	**45**	**48**	**53**	**67**	**57**	**59**	**64**	**61**	**62**	**68**	**64**

Table 2. Performance for different classifiers. The columns R, P, and F represent recall, precision, and f-measure, respectively. (in %)

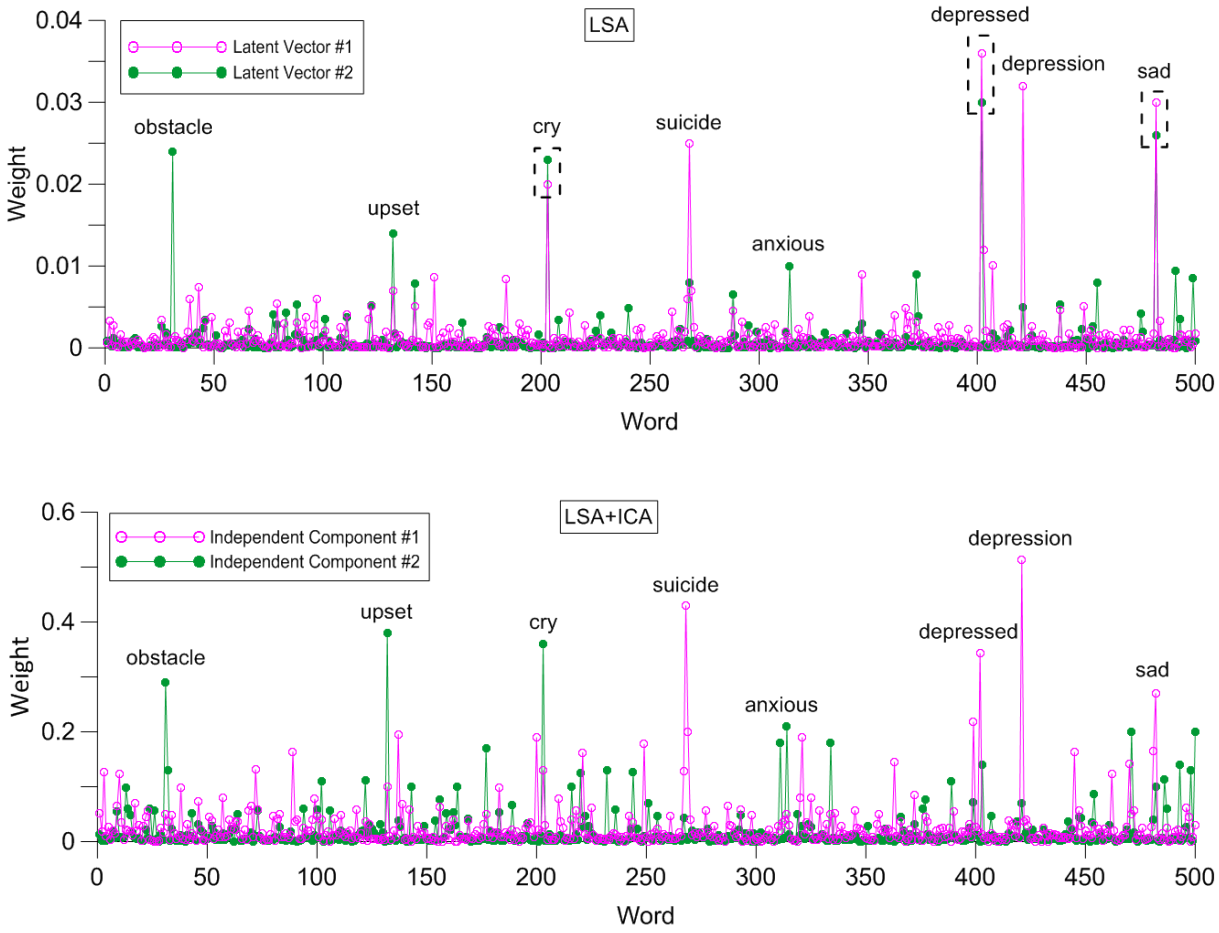


Figure 8. Examples of latent vectors, selected from $\mathbf{U}_{Q\times k}$, and independent components, selected from $\mathbf{W}_{Q\times k}^{T}$, for the emotion labels <Depression> and <Mood>.

3.4 Term Overlap Analysis

In order to investigate the term overlap in LSA and LSA+ICA, we analyze their respective corresponding matrices $\mathbf{U}_{Q\times k}$ and $\mathbf{W}_{Q\times k}^{T}$ where $\mathbf{W}_{Q\times k}^{T}$ is the transpose of the demixing matrix obtained with the input of $\mathbf{X}_{Q\times D}$ reconstructed using LSA. Each column of $\mathbf{U}_{Q\times k}$ and $\mathbf{W}_{Q\times k}^{T}$ represents a latent vector/independent component of *Q* words, and each element in the vector is a word weight representing its relevance to the corresponding latent vector/independent component. Figure 8 shows two sample latent vectors for LSA and two independent components for LSA+ICA, where the weights shown in this figure are the absolute values.

The upper part of Fig. 8 shows parts of the words and their weights in the two latent vectors, where latent vector #1 can be characterized by *depressed*, *depression*, and *sad* which are the useful features for identifying the emotion label <Depression>, and latent vector #2 can be characterized by *depressed*, *sad,* and *cry* which are useful for identifying <Mood>. Although the two latent vectors contained useful features for the respective emotion labels, these features still had some overlap between the latent vectors, as marked by the dashed rectangles. The overlapped features, especially those with higher weights, may reduce the classifier's ability to distinguish between the emotion labels. The lower part of Fig. 8 also shows two independent components for the emotion labels <Depression> and<Mood>. As indicated, the term overlap between the two independent components was relatively low. Table 3 shows some representative words (with higher weights) in the independent components for the emotion labels.

Emotion Label	Representative Words
Depression	depression, depressed, sad, down, suicide
Drug	Medication, drug, dose, sedative, withdrawal,
Insomnia	sleep, insomnia, dream, nightmare, awake
Mood	cry, upset, anxious, energy, obstacle
OCD	OCD, compulsion, weight, overeating, behavior
Schizophrenia	paranoia, fantasy, memory, split, genetic
Social Anxiety	crowd, tense, friend, stiffness, ridicule, shivering

Table 3. Representative words for the emotion labels.

4 Conclusions

This work has presented a framework combining LSA and ICA for emotion label identification. Both LSA and ICA are used to analyze concept-level features, where LSA is used to discover latent concepts that do not frequently occur in psychiatric texts, and ICA is used to further minimize the term dependence among the concepts. The experimental results show that the use of concept-level features yielded better performance than the use of word-level features. Additionally, ICA can reduce the degree of term overlap of LSA so that combining LSA and ICA can discover more useful latent concepts with minimized term dependence for different emotion labels, thus improving classification performance. Future work will focus on investigating the use of the machine-labeled emotion labels as meta-information to improve online psychiatric services such as information retrieval for self-help knowledge recommendation.

Acknowledgments

This work was supported by the Ministry of Science and Technology, Taiwan, ROC, under Grant No. NSC102-2221-E-155-029-MY3.

Reference

Y. M. Bai, C. C. Lin, J. Y. Chen, and W.C. Liu. 2001. Virtual psychiatric clinics. *American Journal of Psychiatry*, 158(7): 1160-1161.

A. Balahur, R. Mihalcea, and A. Montoyo. 2014. Computational approaches to subjectivity and sentiment analysis: Present and envisaged methods and applications. *Computer Speech & Language* 28(1): 1-6.

R. A. Calvo and S. D'Mello. 2010. Affect Detection: An interdisciplinary review of models, methods, and their applications. *IEEE Trans. Affective Computing*, 1(1): 18-37.

J. T. Chien and H. L.Hsieh. 2012. Convex divergence ICA for blind source separation. *IEEE Trans. Audio, Speech, and Language Processing*, 20(1): 302-313.

R. Chitturi and J. H. L. Hansen. 2008. Dialect classification for online podcasts fusing acoustic and language based structural and semantic information. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL-08)*, pages 21-24.

G. H. Golub and C. F. Van Loan. 1996. *Matrix Computations*, Third Edition, Johns Hopkins University Press, Baltimore, MD.

T. Honkela, A. HyvÄrinen, and J. J. VÄyrynen. 2010. WordICA — emergence of linguistic representations for words by independent component analysis. *Natural Language Engineering*, 16(3): 277-308.

A. Hyvärinen, J. Karhunen, and E. Oja. 2001. *Independent Component Analysis*. Wiley, New York.

R. Johansson and A. Moschitti. 2013. Relational features in fine-grained opinion analysis. *Computational Linguistics*, 39(3): 473-509.

W. Kessler and H. Schütze. 2012. Classification of inconsistent sentiment words using syntactic constructions. *Proceedings of the 24th International Conference on Computational Linguistics (COLING-12)*, pages 569-578.

T. Kolenda and L. K. Hansen. 2000. Independent components in text. *Advances in Neural Information Processing Systems* 13: 235-256.

T. K. Landauer, P. W. Foltz, and D. Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes*, 25(2&3): 259-284.

T. W. Lee. 1998. Independent Component Analysis—Theory and Applications. Kluwer, Norwell, MA.

S. Li and C. Zong. 2008. Multi-domain Sentiment Classification. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL-08)*, pages 257-260.

C. C. Lin, Y. M. Bai, and J. Y. Chen. 2003. Reliability of information provided by patients of a virtual psychiatric clinic, *Psychiatric Services*, 54(8): 1167-1168.

B. Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool, Chicago, IL.

G. R. Naik and D. K. Kumar. 2011. An overview of independent component analysis and its applications. *Informatica* 35(1): 63-81.

M. Naughton, N. Stokes, and J. Carthy. 2008. Investigating statistical techniques for sentence-level event classification. *Proceedings of the 22nd International Conference on Computational Linguistics (COLING-08)*, pages 617-624.

B. Pang and L. Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2: 1-135.

R. W. Picard. 1997. *Affective Computing*, MIT Press, Cambridge, MA.

M. Post and S. Bergsma. 2013. Explicit and implicit syntactic features for text classification. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL-13)*, pages 866–872.

R. Rapp. 2004. Mining text for word senses using independent component analysis. *Proceedings of the 4th SIAM International Conference on Data Mining (SDM-04)*, pages 422-426.

H. Saruwatari, T. Kawamura, T. Nishikawa, A. Lee, and K. Shikano. 2006. Blind source separation based on a fast-convergence algorithm combining ICA and beamforming. *IEEE Trans. Audio, Speech, and Language Processing*, 14(2): 666-678.

C. H. Wu, L. C. Yu, and F. L. Jang. 2005. Using semantic dependencies to mine depressive symptoms from consultation records. *IEEE Intelligent System*, 20(6): 50-58.

L. C. Yu, C. L. Chan, C. C. Lin, and I. C. Lin. 2011. Mining association language patterns using a distributional semantic model for negative life event classification. *Journal of Biomedical Informatics*, 44(4): 509-518.

L. C. Yu and W N. Chien. 2013. Independent component analysis for near-synonym choice. *Decision Support Systems*, 55(1): 146-155.

L. C. Yu, C. H. Wu, and F. L. Jang. 2009. Psychiatric document retrieval using a discourse-aware model. *Artificial Intelligence*, 173(7-8): 817-829.

Modeling Mutual Influence Between Social Actions and Social Ties

Xiaofeng YU **Junqing XIE**

HP Labs China
Universal Business Park
10 Jiu XianQiao Road, Chaoyang District, Beijing, China
{xiaofeng.yu,jun-qing.xie}@hp.com

Abstract

In online social media, social action prediction and social tie discovery are two fundamental tasks for social network analysis. Traditionally, they were considered as separate tasks and solved independently. In this paper, we investigate the high correlation and mutual influence between social actions (i.e. user-behavior interactions) and social ties (i.e. user-user connections). We propose a unified coherent framework, namely mutual latent random graphs (MLRGs), to flexibly encode evidences from both social actions and social ties. We introduce latent, or hidden factors and coupled models with users, users' behaviors and users' relations to exploit mutual influence and mutual benefits between social actions and social ties. We propose a gradient based optimization algorithm to efficiently learn the model parameters. Experimental results show the validity and competitiveness of our model, compared to several state-of-the-art alternative models.

1 Introduction

With the dramatically rapid growth and great success of many large-scale online social networking services, social media bridge our daily physical life and the virtual Web space. Popular social media sites (e.g., Facebook and Twitter) and mobile social networks (e.g., Foursquare) have gathered billions of acting users and are still attracting millions of newbies everyday. Modeling *social actions* and *social ties* are two fundamental tasks in online social media. Social actions are the users' activities or behaviors in socially connected networks. For example, a social action can be "posting a tweet" on Twitter or the "check-in" behavior on Foursquare. A social tie or social relation is referred to any relationship between two or more individual users in a social network, such as the friend and colleague relationships. By understanding a user's behaviors and accordingly exploiting potentially interesting services to her/him, one can improve the user's experience and boost the revenue of social media sites. Also, precise social tie prediction will help people tap into the wisdom of crowds, to aid in making more informed decisions.

Since individual users are socially connected, social influence occurs through information diffusion in social networks. Social influence happens when one's opinions or behaviors are affected by others. It is well known that different types of social ties have essentially different influence on social actions. Intuitively, a user's trusted friends on the web affect that user's online behavior. Ma *et al.* (2009) and Ma *et al.* (2011) claimed that one user's final behavior decision is the balance between his/her own taste and her/his trusted friends' favors. On the other hand, social actions also have important influence on social ties. Obviously, users with similar preferences or behaviors are more likely to be friends than others in social media. Users with momentous activities will attract many other users to be connected with. On the contrary, no body will be interested in users with trivial or insignificant behaviors.

Consequently, we face some very interesting questions: Is there any dynamics or mutual influence between social actions and social ties? To what extent do they influence each other? A fundamental mechanism that drives the dynamics of networks is the underlying social phenomenon of *homophily* (McPherson *et al.*, 2001): people tend to follow the behaviors of their friends, and people tend to create relationships with other people who are already similar to them. This suggests that both actions and ties

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 848–859, Dublin, Ireland, August 23-29 2014.

are bi-directionally correlated and mutually influenced in social media, they could be mutually reinforced if modeled jointly.

Inspired by this mechanism, we propose a single unified framework based on exponential-family random graph models ((Frank and Strauss, 1986), (Wasserman and Pattison, 1996)) to exploit homophily for simultaneous social action prediction and social tie discovery. This mutual latent random graph (MLRG) framework incorporates shared latent factors with users, users' behaviors and users' relations, and defines coupled models to encode both social action and social tie information, to capture dynamics and mutual influence between them. We propose a gradient based algorithm for learning and optimization. During the learning procedure, social actions (i.e. user-behavior interactions), social ties (i.e. user-user connections), and deep dependencies and interactions between them could be efficiently explored. Experimental results demonstrate that social actions and social ties are highly correlated and mutually helpful. By coupling actions with ties jointly in a single coherent framework, MLRG achieves significantly better performance on both social action prediction and social tie inference, compared to state-of-the-art systems modeling them independently.

2 Related Work

Social network analysis has attracted much interest in both academia and industry recently. Considerable research and engineering has been conducted for social media modeling, analytics and optimization, including social community detection (Fortunato, 2010), user behavior modeling and prediction ((Benevenuto *et al.*, 2009), (Kwak *et al.*, 2010), (Ma *et al.*, 2009), (Ma *et al.*, 2011)), social tie analysis ((Tang *et al.*, 2011), (Tang *et al.*, 2012)), social sentiment analysis ((Wasserman *et al.*, 1994), (Pang and Lee, 2008)), etc.

Social action prediction and social tie discovery are two fundamental tasks for social media and social network analysis. Traditionally, they were considered as separate tasks and solved independently without considering the bidirectional interactions and interdependencies between them. Social action investigation is essentially important in online social media. Users behaviors could be affected by various kinds of complex factors, such as users' attributes, users's historical behaviors, social influence and social network structures. Based on this motivation, Tan *et al.* (2010) proposed a noise tolerant time-varying model to track social actions. Aiming at modeling user actions more accurately and realistically, Ma *et al.* (2009) and Ma *et al.* (2011) considered connections among users and proposed social trust ensemble to fuse the users' tastes and their trusted friends' favors together. Gao *et al.* (2013) investigated users' social behaviors from a spatio-temporal-social aspect in location-based mobile social networks. In particular, Gao *et al.* (2013) focused on temporal effects in terms of temporal preferences and temporal correlations, and modeled temporal cyclic patterns to capture a user's mobile behavior to investigate correlations to the spatial context and social context in location-based social networks.

Social tie is the most basic unit to form the network structure. Tang *et al.* (2011) proposed a semi-supervised framework, the partially labeled factor graph model to infer the type of social relationships. The task was formulated as a relationship mining problem to detect the relationship semantics in real-world networks. Tang *et al.* (2012) further incorporated social theories and leveraged features based on those social theories to infer social ties across heterogeneous networks via a transfer learning framework. As can be seen, predicting social actions and inferring social ties were modeled as separate and independent tasks in the above-mentioned approaches, deep interactions and mutual influence between them were not taken into consideration. In social media users interact with one another to share the content they both create and consume. According to the homophily phenomenon, exploring bi-directional information and mutual influence between them is intuitively appealing.

We are also aware of several research work attempting to explore joint models to capture mutual benefits and deep dependencies between different tasks in NLP, data mining and information extraction research communities ((Ko *et al.*, 2007), (Yu and Lam, 2008), (Yu *et al.*, 2009), (Liu *et al.*, 2009), (Yu and Lam, 2010b), (Yu and Lam, 2010a), (Yu *et al.*, 2011), (Yu and Lam, 2012), (Zeng *et al.*, 2013)). In general, joint models aim to handle multiple hypotheses and uncertainty information and to predict many variables simultaneously such that subtasks can aid each other to boost the performance.

Ko *et al.* (2007) proposed a joint answer ranking framework based on probabilistic graphical models for question answering. Yu and Lam (2008) proposed an integrated probabilistic and logic approach based on Markov Logic Networks (MLNs) (Richardson and Domingos, 2006) to encyclopedia relation extraction. However, this modeling only captures single relation extraction task. Liu *et al.* (2009) developed a Bayesian hierarchical approach, the topic-link LDA, to perform topic modeling and author community discovery for large-scale linked documents in one unified framework. Yu *et al.* (2009) integrated two sub-models in a unified framework via Markov chain Monte Carlo (MCMC) sampling based inference algorithms. This is a loosely coupled model since parameter estimation is performed separately for the two sub-models. Yu and Lam (2012) further proposed a joint model incorporating probabilistic graphical models and first-order logic for information extraction. This joint model exploits structured variational approximation for tractable parameter learning. Zeng *et al.* (2013) presented a semi-supervised graph-based approach to joint Chinese word segmentation and POS tagging. However, none of these models has been investigated or applied to social media and social network analysis. We believe that one major reason could be the problem of high computational complexity, such as Yu *et al.* (2009) and Yu and Lam (2012). Since many social network sites contain millions of users, exploiting such models could be very challenging. Currently, research on building joint approaches is still in the infancy stage. To the best of our knowledge, there is few systematically study on building joint models to explore mutual influence for social actions and social ties.

3 Model

In this section we consider both social action prediction and social tie inference in the context of social media, where evidences for both actions and ties are available. We begin by necessary description of preliminaries and notations, we then present the mutual latent random graphs (MLRGs) model, upon which both sources of evidence could be exploited simultaneously to capture their mutual influence. We also discuss the major difference and superiority of this model against several alternative models.

3.1 Preliminaries and Notations

Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a social network graph, where $\mathbf{V} = \{v_1, v_2, \ldots, v_N\}$ is the set of $|\mathbf{V}| = N$ users and $\mathbf{E} = \{e_{11}, e_{12}, \ldots, e_M\} \subset \mathbf{V} \times \mathbf{V}$ is the set of $|\mathbf{E}| = M$ connections between users. Let $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}(y_i \in \mathcal{Y})$ be the set of actions associated with N users, and $\mathbf{s} = \{s_{11}, s_{12}, \ldots, s_M\}(s_{ij} \in \mathcal{S})$ be the set of corresponding social tie labels associated with M connections. The connection $e_{ij}(1 \leq i, j \leq N, i \neq j)$ between v_i and v_j might be directed or undirected. To be consistent, both $s_{ij} \neq s_{ji}$ and $s_{ij} = s_{ji}$ are valid settings. Given the observed social network data $\mathcal{D}$ constructing the graph $\mathcal{G}$, our goal is to simultaneously detect the most likely types of actions $\mathbf{y}^*$ and ties $\mathbf{s}^*$ such that both of them are optimized.

The exponential-family random graph models (ERGMs) ((Frank and Strauss, 1986), (Wasserman and Pattison, 1996)) take the form of an exponential family as $P_{y_i|\mathcal{G}} = \prod_{y_i \in \mathcal{Y}} \phi(y_i) = \frac{\exp\{\sum_{y_i \in \mathcal{Y}} \eta \boldsymbol{\xi}(y_i)\}}{\kappa_\eta}$ for the social action y_i in the social network graph $\mathcal{G}$, where $\phi(\cdot)$ is a factor, η is a vector of parameters, $\boldsymbol{\xi}(\cdot)$ is a p-vector of sufficient statistics, which captures network features of interest, its postulated dependence structure, or both. Lastly, κ_η is a normalization function to make all probabilities sum to one. The class of ERGMs is a popular framework for social network modeling to capture global network characteristics.

3.2 Modeling Social Actions

To characterize the user action y_i, we assume that for the user v_i there exist observable attributes or properties $\mathbf{m}_i$, such as the user's registered information and historical actions. Without loss of generality, we further assume that there exist some hidden, or latent properties $\mathbf{x}_{ij}$ for v_i. These properties are implicit and cannot be observed directly, such as the influence from social ties. Consequently, we denote the observable factor $\phi(y_i, v_i, \mathbf{m}_i)$ for observable properties and latent factor $\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$ for hidden properties, respectively. Given the graph $\mathcal{G}$, the probability distribution of y_i depends on both observable and latent factors as:

$$P_{y_i|\mathcal{G}} \sim \phi(y_i, v_i, \mathbf{m}_i), \quad P_{y_i|\mathcal{G}} \sim \phi_h(y_i, s_{ij}, \mathbf{x}_{ij}), \quad P_{y_i|\mathcal{G}} \sim \phi(y_i, v_i, \mathbf{m}_i)\phi_h(y_i, s_{ij}, \mathbf{x}_{ij}). \tag{1}$$

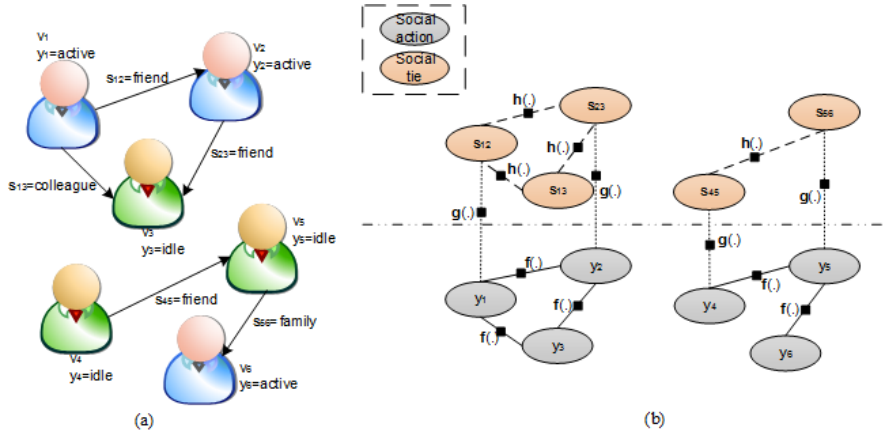


Figure 1: (a) A social network containing 6 users and 5 social ties. The social action can be active or idle, and the social tie can be friend, colleague, or family. (b) The three-dimensional graphical representation of the corresponding MLRG model. We use different lines to represent functions $\mathbf{f}(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$.

The Mutual Latent Random Graph (MLRG) model	
$\forall y_i \in \mathcal{Y}$	$P_{y_i\|\mathcal{G}} \sim \phi(y_i, v_i, \mathbf{m}_i)\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$
$\forall s_{ij} \in \mathcal{S}$	$P_{s_{ij}\|(y_i,\mathcal{G})} \sim \phi'(s_{ij}, v_i, v_j, \mathbf{w}_{ij})\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$
$\forall y_i \in \mathcal{Y}, \forall s_{ij} \in \mathcal{S}$	$P_{(y_i,s_{ij})\|\mathcal{G}} \sim \phi(y_i, v_i, \mathbf{m}_i)\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})\phi'(s_{ij}, v_i, v_j, \mathbf{w}_{ij})$

This modeling integrates two types of factors for both observable and latent properties. It captures not only the user-behavior dependencies, but also the influence from social ties, for exploring social actions.

3.3 Modeling Social Ties

To characterize the social tie s_{ij} between user pair (v_i, v_j), we also assume that there exist observable properties $\mathbf{w}_{ij}$, such as the posterior probability of the social tie s_{ij} assigned to (v_i, v_j). We denote the observable factor $\phi'(s_{ij}, v_i, v_j, \mathbf{w}_{ij})$ for $\mathbf{w}_{ij}$. Similarly, we further assume that there exist some latent properties to incorporate the social action influence on social ties. To be consistent, we still use the vector $\mathbf{x}_{ij}$ to represent the latent properties and the latent factor $\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$ to capture the social action influence on social ties. Note that both $\mathbf{x}_{ij}$ and $\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$ now play double duties in encoding social action dependency and social tie connection simultaneously. On the one hand, $\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$ exploits influence from social ties for modeling social actions. On the other hand, this factor exploits influence from social actions for modeling social ties. By doing so, the latent factor $\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$ is bi-directionally coupled, encoding both sources of evidence and exploring mutual influence and dynamics between social actions and social ties. Such mutual influence and dynamics are crucial and modeling them often leads to improved performance. Given the user action y_i and the graph $\mathcal{G}$, we devise the following model for the probability distribution of s_{ij} depending on both observable and latent factors as:

$$P_{s_{ij}|(y_i,\mathcal{G})} \sim \phi'(s_{ij}, v_i, v_j, \mathbf{w}_{ij}), \quad P_{s_{ij}|(y_i,\mathcal{G})} \sim \phi_h(y_i, s_{ij}, \mathbf{x}_{ij}), \quad P_{s_{ij}|(y_i,\mathcal{G})} \sim \phi'(s_{ij}, v_i, v_j, \mathbf{w}_{ij})\phi_h(y_i, s_{ij}, \mathbf{x}_{ij}). \quad (2)$$

3.4 Modeling Mutual Influence

The mutual correlation between social actions and social ties advocates joint modeling of both sources of evidence in a single unified framework. Based on the above descriptions, we define our mutual latent random graph (MLRG) based on exponential-family random graph models (ERGMs) ((Frank and Strauss, 1986), (Wasserman and Pattison, 1996)), which have gained tremendous successes in social network analysis and have even become the current state-of-the-art (Robins *et al.*, 2007). To design a concrete model, one needs to specify distributions for the dependencies for MLRGs. According to the celebrated Hammersley-Clifford theory, the joint conditional distribution $P_{(y_i,s_{ij})|\mathcal{G}}$ is factorized as a product of potential functions over all cliques in the graph $\mathcal{G}$ and we summarize the MLRG in the above table. In summary, our model consists of three factors: the factor $\phi(y_i, v_i, \mathbf{m}_i)$ measuring dependencies

of the social action y_i conditioned on $\mathcal{G}$, the factor $\phi'(s_{ij}, v_i, v_j, \mathbf{w}_{ij})$ measuring the social tie s_{ij} between two arbitrary users v_i and v_j in $\mathcal{G}$, and the latent factor $\phi_h(y_i, s_{ij}, \mathbf{x}_{ij})$ exploiting mutual influence between the social action y_i and social tie s_{ij}.

The three factors $\phi(\cdot)$, $\phi_h(\cdot)$, and $\phi'(\cdot)$ can be instantiated in different ways. In this paper, each factor is defined as the exponential family of an inner product over sufficient statistics (feature functions) and corresponding parameters. Each factor is a clique template whose parameters are tied. More specifically, we define these factors as

$$\phi(y_i, v_i, \mathbf{m}_i) = \frac{1}{Z_\alpha} \exp\{\sum_{y_i \in \mathcal{Y}} \alpha \mathbf{f}(y_i, v_i, \mathbf{m}_i)\}, \quad \phi_h(y_i, s_{ij}, \mathbf{x}_{ij}) = \frac{1}{Z_\beta} \exp\{\sum_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \beta \mathbf{g}(y_i, s_{ij}, \mathbf{x}_{ij})\},$$
$$\phi'(s_{ij}, v_i, v_j, \mathbf{w}_{ij}) = \frac{1}{Z_\gamma} \exp\{\sum_{s_{ij} \in \mathcal{S}} \gamma \mathbf{h}(s_{ij}, v_i, v_j, \mathbf{w}_{ij})\}, \tag{3}$$

where α, β, and γ are real-valued weighting vectors and $\mathbf{f}(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$ are corresponding vectors of feature functions.

We denote $\Theta = \{\alpha, \beta, \gamma\}$ as the set of model's parameters, and concatenate all factor functions as $\Theta \mathbf{q}(y_i, s_{ij}) = \alpha \mathbf{f}(y_i, v_i, \mathbf{m}_i) + \beta \mathbf{g}(y_i, s_{ij}, \mathbf{x}_{ij}) + \gamma \mathbf{h}(s_{ij}, v_i, v_j, \mathbf{w}_{ij})$, the joint probability distribution shown in the above table can be rewritten as

$$P_{(\mathbf{y},\mathbf{s})|\mathcal{G}} = \prod_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \Phi(y_i, s_{ij}) = \frac{1}{Z} \exp\{\sum_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \Theta \mathbf{q}(y_i, s_{ij})\}, \tag{4}$$

where $Z = Z_\alpha Z_\beta Z_\gamma$ is the partition function of our MLRG model.

Figure 1 shows an example social network ($\mathbf{V} = \{v_1, v_2, v_3, v_4, v_5, v_6\}$, $\mathcal{Y} = \{\text{active}, \text{idle}\}$, $\mathcal{S} = \{\text{friend}, \text{colleague}, \text{family}\}$) and the corresponding 3D graphical representation of the MLRG model. The functions $\mathbf{f}(\cdot)$ model dependencies of social actions in the bottom part, and the functions $\mathbf{h}(\cdot)$ model dependencies of social ties in the upper part. More importantly, the functions $\mathbf{g}(\cdot)$ capture mutual influence and dependencies between social actions and social ties. As we will see, this modeling offers a natural formalism for exploiting bi-directional dependencies and interactions between social actions and social ties to capture their mutual influence, as well as a great flexibility to incorporate a large collection of arbitrary, overlapping and nonindependent features.

3.5 Discussion

Noticeably, our proposed MLRG model is essentially different from the standard exponential-family random graph models (ERGMs) and the prior models discussed in Section 2 mainly in two aspects. Firstly, compared to the standard ERGMs, the MLRG model defines latent factors to assume mutual and dynamical interaction between social ties and social actions. Secondly, compared to the prior models such as (Ma *et al.*, 2009) and (Tang *et al.*, 2011), MLRG provides a single unified framework to address both social action prediction and social tie inference simultaneously while enjoying the benefits of both sources of evidence.

Importantly, we give an analytical explanation on the mutual nature of our model in terms of a random walk (Lovász, 1996) perspective. A random walk on the graph $\mathcal{G}$ is a reversible Markov chain on the vertexes $\mathbf{V}$. The social influence propagation procedure occurs through information diffusion in the social graph $\mathcal{G}$. More specifically, a user v_i will propagate his/her influence to other related users, and will propagate more to the user which has a stronger relation (e.g., friendship) with v_i. The influence propagation will stop when the social graph reaches an equilibrium state, in which both social actions and social ties are mutually reinforced. Interestingly, this process is consistent with the homophily phenomenon that a user in the social network tends to be similar to his/her connected neighbors.

4 Learning and Inference

4.1 Mutual Optimization

The goal of learning MLRG model is to estimate a parameter configuration $\Theta = \{\alpha, \beta, \gamma\}$ such that the log-likelihood of observation is maximized. We define the log-likelihood objective function $\mathcal{O}(\Theta)$ of the

Algorithm 1: The Mutual Gradient Descent (MGD) algorithm

Input: The social graph $\mathcal{G}$, number of iterations n, and the learning rate η.
Output: Optimized parameters $\Theta^* = \{\alpha^*, \beta^*, \gamma^*\}$.
while *equilibrium states or a threshold number of iterations are not reached* **do**
  repeat
    Choose a random example $(y_i, s_{ij}) \in \mathcal{G}$ as a sample;
    Optimize social action parameters α **and** β:
      Compute the approximated gradients $\frac{\partial \mathcal{O}'}{\partial \alpha}$ and $\frac{\partial \mathcal{O}'}{\partial \beta}$ according to Eq. (6), Eq. (7) and stochastic approximation;
      Update α and β with learning rate η: $\alpha \leftarrow \alpha - \eta \cdot \frac{\partial \mathcal{O}'}{\partial \alpha}, \quad \beta \leftarrow \beta - \eta \cdot \frac{\partial \mathcal{O}'}{\partial \beta}$.
      `// Explore social tie influence`
    Optimize social tie parameters γ **and** β:
      Compute the approximated gradients $\frac{\partial \mathcal{O}'}{\partial \gamma}$ and $\frac{\partial \mathcal{O}'}{\partial \beta}$ according to Eq. (8), Eq. (7) and stochastic approximation;
      Update γ and β with learning rate η: $\gamma \leftarrow \gamma - \eta \cdot \frac{\partial \mathcal{O}'}{\partial \gamma}, \quad \beta \leftarrow \beta - \eta \cdot \frac{\partial \mathcal{O}'}{\partial \beta}$.
      `// Explore social action influence`
  until *converge*;
end
return α^*, β^*, *and* γ^*

observation given the graph $\mathcal{G}$ as

$$\mathcal{O}(\Theta) = \log P_{(\mathbf{y},\mathbf{s})|\mathcal{G}} - \log \Omega(\Theta) = \log[\exp\{\sum_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \Theta \mathbf{q}(y_i, s_{ij})\}] - \log Z - \log \Omega(\Theta), \tag{5}$$

where $\Omega(\Theta)$ is regularization to reduce over-fitting and a common choice is a spherical Gaussian prior with mean 0 and covariance $\delta^2 I$. $\Omega(\Theta) = \sum_{y_i \in \mathcal{Y}} \frac{\alpha^2}{2\sigma^2} + \sum_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \frac{\beta^2}{2\sigma^2} + \sum_{s_{ij} \in \mathcal{S}} \frac{\gamma^2}{2\sigma^2}$.

We propose a mutual gradient descent (MGD) algorithm based on the stochastic gradient descent (SGD) ((Lecun *et al.*, 1998), (Bottou, 2004)) framework, for estimating the parameters efficiently in a mutual and collaborative manner. Once we have optimized the social action parameters α and β, the influence and hypotheses of social action can aid the learning of the social tie parameters γ and β, and vice versa. As shown in Algorithm 1, β is coupled parameter vector for both actions and ties, and is updated twice in each iteration of MGD. By doing so, MGD not only allows learning of social action parameters to capture social tie influence, but it also optimizes social tie parameters to alleviate social action influence. This training procedure runs iteratively until converge to boost both the optimization of social actions and social ties.

Each iteration of the MGD algorithm consists of drawing an example at random and applying parameter updates by moving in the direction defined by the stochastically approximated gradient of the loss function (e.g., $\frac{\partial \mathcal{O}'}{\partial \alpha}$). We update each parameter with a learning rate η. Ideally, each parameter should have its own learning rate. If shared parameter weights are used, the best learning rate of a weight should be inversely proportional to the square root of the number of connection sharing that weight (Bottou, 2004). In our MGD implementation, for simplicity we use the same learning rate for all the parameters. We select a small subset of training data and try various learning rates on the subset, then pick the one that most reduces the loss and use it on the full dataset. We summarize the partial derivatives of the log-likelihood function $\mathcal{O}$ with respect to the parameter vectors α, β and γ as follows:

$$\frac{\partial \mathcal{O}}{\partial \alpha} = \sum_{y_i \in \mathcal{Y}} \mathbf{f}(y_i, v_i, \mathbf{m}_i) - \sum_{y_i \in \mathcal{Y}} \mathbf{f}(y_i, v_i, \mathbf{m}_i) \times P_{(\mathbf{y},\mathbf{s})|\mathcal{G}} - \sum_{y_i \in \mathcal{Y}} \frac{\alpha}{\sigma^2}, \tag{6}$$

$$\frac{\partial \mathcal{O}}{\partial \beta} = \sum_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \mathbf{g}(y_i, s_{ij}, \mathbf{x}_{ij}) - \sum_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \mathbf{g}(y_i, s_{ij}, \mathbf{x}_{ij}) \times P_{(\mathbf{y},\mathbf{s})|\mathcal{G}} - \sum_{y_i \in \mathcal{Y}, s_{ij} \in \mathcal{S}} \frac{\beta}{\sigma^2}, \tag{7}$$

$$\frac{\partial \mathcal{O}}{\partial \gamma} = \sum_{s_{ij} \in \mathcal{S}} \mathbf{h}(s_{ij}, v_i, v_j, \mathbf{w}_{ij}) - \sum_{s_{ij} \in \mathcal{S}} \mathbf{h}(s_{ij}, v_i, v_j, \mathbf{w}_{ij}) \times P_{(\mathbf{y},\mathbf{s})|\mathcal{G}} - \sum_{s_{ij} \in \mathcal{S}} \frac{\gamma}{\sigma^2}. \tag{8}$$

It is worth noting that the MGD algorithm computes approximations of the gradients, due to the intractability of the normalizing constant Z in the log-likelihood of our MLRG model. Our proposed

MGD algorithm is a generalized extension and it distinguishes from the standard SGD algorithm in two aspects: (1) MGD optimizes three types of parameters simultaneously, thus MGD is much more general than SGD, and it is more scalable and applicable to real-world problems. (2) MGD performs mutual and collaborative optimization to enable mutual influence between social actions and social ties, whereas SGD does not take such influence into account.

4.2 Complexity Analysis

Given several conditions including a suitable choice of the learning rate and a convex or pseudo-convex objective function, the MGD algorithm converges almost surely to a global optimum, otherwise it converges almost surely to a local optimum. In our experiments, this algorithm has good performance even if it does not reach the global optimum. Let D be the number of samples in the social graph $\mathcal{G}$, n be the number of iterations, and $\overline{p}$ be the average number of non-zero attributes (features) per sample, the computational complexity of our MGD algorithm takes $O(nD\overline{p})$. As can be seen, this algorithm is computationally efficient, and convergence is very fast when the training examples are redundant since only a few examples are needed to sample. Furthermore, this algorithm is online and scale sub-linearly with the amount of training data, making it very attractive for large-scale datasets.

4.3 Inference

The objective of inference is to find the most likely types of actions $\mathbf{y}^*$ and corresponding social tie labels $\mathbf{s}^*$, that is, to find $(\mathbf{y}^*, \mathbf{s}^*) = \arg\max_{(\mathbf{y},\mathbf{s})} P_{(\mathbf{y},\mathbf{s}|\mathcal{G})}$. The inference procedure is straightforward. Based on the learned parameters $\Theta^* = \{\alpha^*, \beta^*, \gamma^*\}$, we firstly predict the label of each social action y_i by finding a labeling assignment that maximizes $P_{y_i|\mathcal{G}}$ as $y_i^* = \arg\max_{y_i \in \mathcal{Y}} P_{y_i|\mathcal{G}}$. We then infer the social tie label s_{ij} such that $s_{ij}^* = \arg\max_{s_{ij} \in \mathcal{S}} P_{s_{ij}|(y_i,\mathcal{G})}$.

5 Experiments

5.1 Foursquare Data

We crawled one dataset from Foursquare[1], a popular location-based mobile social networking site for mobile devices (e.g., smartphones) for our experimental evaluation. Foursquare allows a user to check in at a physical location via his cellphone, and then let his online friends know where he is by publishing such check-in action online. Users check-in at venues using a mobile website, text messaging or a device-specific application by selecting from a list of venues the application locates nearby. Location is based on GPS hardware in the mobile device or network location provided by the application, and the map is based on data from the OpenStreetMap project. Each check-in awards the user points and sometimes badges. Figure 2 illustrates a snapshot of the Foursquare application interface on smartphones.

To alleviate the data sparsity problem for better evaluation, we selected check-in venues which have been visited by at least two distinct users, and users who have checked in at least 10 distinct venues. The resulting dataset contains 12,368 distinct users, 186,745 venues, 1,425,664 check-in behaviors and 56,395 social connections from January 2012 to December 2012. Table 1 lists the more detailed statistical information on our dataset, where the "Avg. Num. of Check-ins" is the average number of check-ins per user, and "Max. Num. of Check-ins" is the maximal number of check-ins among users (similarly for "Avg. Num. of Friendships" and "Max. Num. of Friendships"). The "Average Clustering Coefficient (ACC)" is a measure of the degree to which users in the Foursquare network tend to cluster together, and "Diameter" is the longest shortest path in the network. All user and venue information has been anonymized. Each check-in has a unique id as well as the user id and the venue id, and each social connection consists of two users represented by two unique ids.

5.2 Task

Mobile phones have become an important tool for communication and they are an ideal platform for understanding social influence and social dynamics. Using our Foursquare dataset, we can investigate the mutual influence between social actions and social ties. More specifically, we can investigate how

[1] https://foursquare.com/

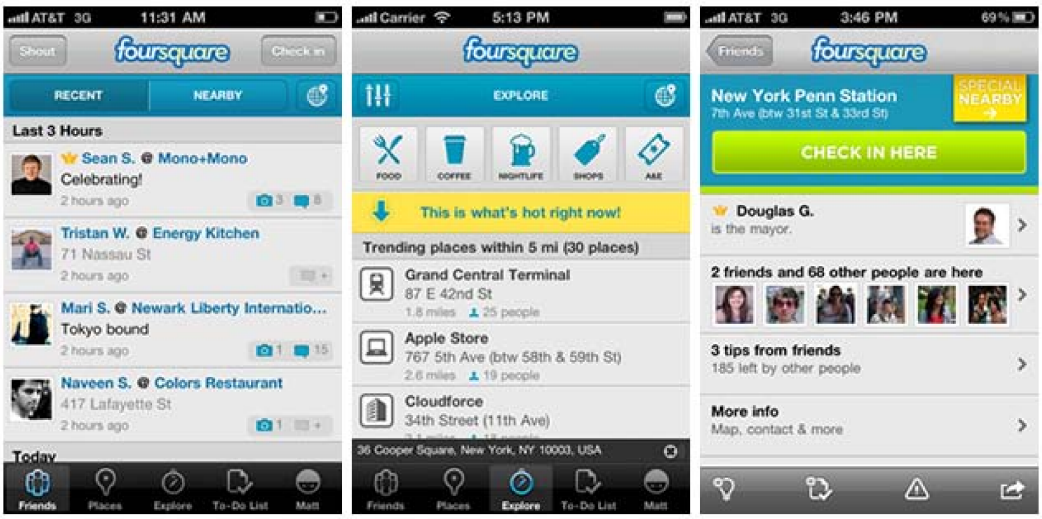


Figure 2: A snapshot of the Foursquare application interface on smartphones.

Duration	Jan 2012 to Dec 2012	Num. of Users	12,368
Num. of Check-ins	1,425,664	Num. of Friendships	56,395
Avg. Num. of Check-ins	115.27	Max. Num. of Check-ins	657
Avg. Num. of Friendships	4.56	Max. Num. of Friendships	265
Average Clustering Coefficient (ACC)	0.42	Diameter	12

Table 1: Statistical information of our Foursquare dataset.

the friendship relations affect users' check-in behaviors, and how users' check-in behaviors affect their friendships. Figure 3 gives an illustrative example of social action prediction and social tie discovery tasks in our Foursquare dataset. Given an unseen Foursquare social network dataset, our objective is to predict whether the users have check-in behaviors and whether there are friendship relations between these users. In the right figure, we list the predicted check-in behaviors (in red color) and the inferred friendship relations between users (in green color). The probabilities associated with the predictions represent corresponding confidence scores.

5.3 Evaluation Methodology

We exploited a wide range of important features to define the factors $\phi(\cdot)$, and $\phi'(\cdot)$, including temporal and social features such as the number of check-ins and number of new check-ins in a user's history, number of friends of a user, the check-in information from a user's friends, etc. For the coupled latent factor $\phi_h(\cdot)$, we incorporated social tie evidences and hypotheses as features to capture social actions, and we also incorporated social action evidences and hypotheses as features to leverage social ties.

For quantitative performance evaluation, we used the standard measures of Precision (P), Recall (R), and F-measure (the harmonic mean of P and R: $\frac{2PR}{P+R}$) for both social action prediction and social tie inference. We performed four-fold cross-validation on this dataset, and took the average performance. We compared our approach with the following alternative methods for predicting social actions and inferring social ties:

–**SVM**: This model views social action prediction and social tie inference as two separate classification problems, and solves them independently. We used the SVM-light[2] package for this model.

–**ERGM**: This is the traditional exponential-family random graph model without the latent factor $\phi_h(\cdot)$ incorporated for social action prediction and social tie inference. Similar to SVM, this model also performs them separately.

–**DCRF**: This model is a dynamical and factorial CRF (Sutton *et al.*, 2007) used to jointly solve the two tasks. This model was originally proposed for labeling and segmenting sequence data, and we directly

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

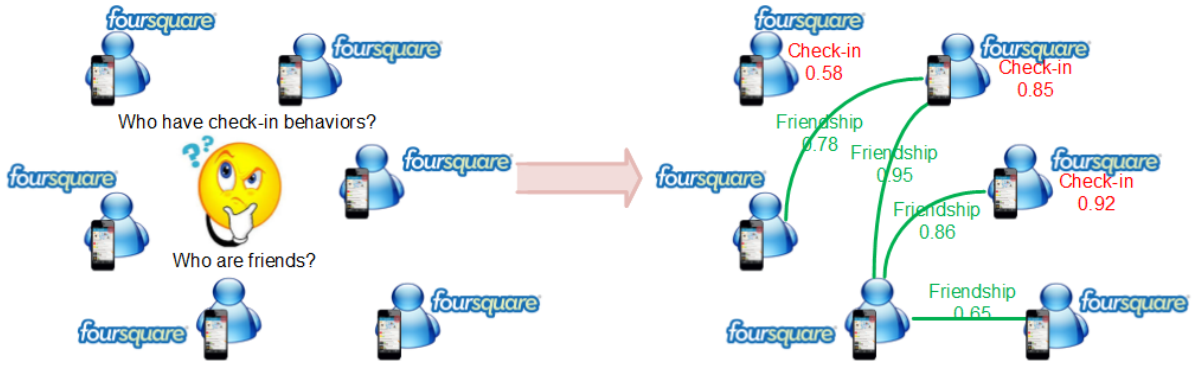


Figure 3: An illustrative example of social action prediction and social tie discovery tasks in our Foursquare dataset. The left is the input of our problem, and the right is the output of the two tasks.

Models	Precision	Recall	F-measure
SVM	73.75	64.54	68.84
ERGM	80.69	79.70	80.19
DCRF	**89.45**	82.32	85.74
MLRG	89.03	**87.89**	**88.46**

Table 2: Comparative performance of different models for social action prediction. The best results are printed in boldface.

Models	Precision	Recall	F-measure
SVM	70.75	61.57	65.84
ERGM	78.85	77.39	78.11
DCRF	82.45	76.56	79.40
MLRG	**84.33**	**83.89**	**84.11**

Table 3: Comparative performance of different models for social tie inference. The best results are printed in boldface.

applied it for our tasks in social network analysis.

All these models exploited standard parameter learning and inference algorithms in our experiments. To avoid over-fitting, penalization techniques on likelihood were also performed. All experiments were performed on the Linux workstation, with 24 2.5GHz Intel Xeon E5-2640 CPUs and 16 GB of memory.

5.4 Performance

Table 2 shows the performance on social action prediction and Table 3 shows the performance on social tie inference of different models, respectively. The best Precision, Recall and F1-measure of these results are highlighted. Our method consistently outperforms other comparative methods on the F-measure. The improvement is statistically and significantly better according to McNemar's paired tests. These results not only imply that there exists high correlation and mutual influence between social actions and social ties, but also demonstrate the feasibility and effectiveness of our model for exploring them.

The SVM model solves social action prediction and social tie inference independently without considering mutual influence and benefits between them, thus leading to the worst performance. The ERGM outperforms SVM by capturing social network structures. However, the performance of this model is still limited and there is a large room for improving. The DCRF model easily outperforms both SVM and ERGM by modeling social actions and social ties jointly in a single framework. However, compared to our MLRG model, there are still some shortcomings of DCRF. DCRF was proposed to label and segment sequence data, such as POS tagging and NP chunking (Sutton *et al.*, 2007). The graphical structure of DCRF is not well suited for social networks to capture mutual influence. The merits of our proposed MLRG model over other models principally come from (1) appropriate graphical structure for social network modeling, especially the coupled latent factor to exploit mutual influence simultaneously, and (2) the mutual and collaborative learning algorithm MGD to reinforce the optimization of both social actions and social ties.

5.5 Effect of Mutual Influence and Analysis

We also examined the nature and effectiveness of the associated latent factors on the mutual influence, and Figure 4 demonstrates their feasibility in our modeling. Note that if we do not incorporate the latent factors, our MLRG model becomes the traditional ERGM baseline approach. It shows that the

latent factors consistently enhance Precision, Recall, and F-measure for both social action prediction and social tie inference tasks. For example, the latent factors significantly improve the F-measure by 8.27% (from 80.19 to 88.46) for social action prediction, and improve the F-measure by 6.0% (from 78.11 to 84.11) for social tie discovery, respectively. These results not only illustrate that social actions and social ties influence each other to a large extent, but also demonstrate the feasibility and effectiveness of our latent factors for exploring them.

We performed an in-depth error analysis to provide gains of our MLRG model and some insights on the influence between users' check-in behaviors and users' friendship relations. By carefully investigating our Foursquare dataset, we found that approximately 75% users tend to cluster together to create tightly knit groups characterized by a relatively high density of friendship relations or ties, and the remaining 25% users loosely or seldom connect with each other through the friendship relations. In other words, 75% users form high density of relationship ties and the average clustering coefficient (ACC) is high (0.61). However, the tie density of the remaining 25% users is much lower, since the ACC of these users is only 0.18. Compared to the baseline methods (especially the SVM and ERGM methods), the performance improvement of our MLRG model mainly comes from 75% users with high density of friendship ties. In particular, about 20% prediction errors (including social action and social tie prediction errors) of such users made by the SVM model can be corrected by our MLRG model. This finding shows that, the mutual influence between users' check-in behaviors and users' friendship relations increases with the density growth of the friendship relations of these users. This finding is intuitively correct and is consistent with the homophily theory. More interestingly, this finding also implies the gains and merits of our MLRG model for exploiting mutual influence, especially when the users in the Foursquare network cluster together tightly with high density of ties.

5.6 Efficiency

A number of learning algorithms can be applied for parameter optimization of our MLRG model. Table 4 summarizes the efficiency of several alternative optimization algorithms for learning our model's parameters. We compared the learning time (hr.) and inference time (sec.) of the MGD algorithm to loopy belief propagation (LBP), Markov chain Monte Carlo(MCMC) Gibbs sampling (Geman and Geman, 1984), and variational mean-field (VMF) approximation algorithms (Wainwright and Jordan, 2008). Both Sutton *et al.* (2007) and Tang *et al.* (2011) used LBP for parameter estimation. LBP is inherently unstable and may cause convergence problems. When the graph has large tree-width as in our case, the LBP algorithm is inefficient, and is slow to converge. In Gibbs sampling, the candidate sample is always accepted with the probability of 1, lacking the capability of measuring quality of samples and eliminating low grade samples. The VMF approach aims to minimize the Kullback-Leibler (KL) divergence between an approximated distribution Q and the target distribution P by finding the best distribution Q from some family of distributions for which an inference is feasible. The MGD algorithm we proposed is very efficient. It is particularly notable that our MGD algorithm takes much less time than other three algorithms for learning. In particular, our proposed algorithm is over orders of magnitude faster than the LBP for running.

6 Conclusions and Future Work

Finally, we answer the questions in Section 1 to draw the conclusions of this paper as follows:

Is there any dynamics or mutual influence between social actions and social ties? Doubtlessly, social actions and social ties are highly correlated and mutually reinforced. We propose a single unified framework, mutual latent random graph (MLRG), to exploit homophily for simultaneous social action prediction and social tie discovery. The MLRG model incorporates coupled latent factors to capture dynamics and mutual influence between social actions and social ties. Moreover, we propose the mutual gradient descent (MGD) algorithm to perform mutual and collaborative optimization to reinforce both social actions and social ties. By coupling actions with ties jointly in a single coherent framework, MLRG achieves significantly better performance on both social action prediction and social tie inference on our collected Foursquare dataset, compared to several state-of-the-art existing models.

To what extent do they influence each other? We perform an in-depth analysis to show the gains and merits of our MLRG model, as well as some insights on the influence between users' check-in behaviors and users' friendship relations. The finding on our real-world Foursquare data demonstrates that social actions (users' check-in behaviors) and social ties (users' friendship relations) influence each other to a considerable degree when the users connect each other tightly with high density of ties in the network. Experimental results also illustrate the feasibility and effectiveness of our latent factors for exploring the mutual influence. In particular, the latent factors in our model significantly improve the F-measure by 8.27% (from 80.19 to 88.46) for social action prediction, and improve the F-measure by 6.0% (from 78.11 to 84.11) for social tie discovery, respectively.

Two directions of future work appear attractive: Inferring fine-grained and multiple relationships between users (such as friendship, family, colleague, and advisor-adviser, etc.) on complex social networks and extending our established optimization algorithms for parallel and distributed learning based on the Hadoop MapReduce framework to handle large scale social networks involving billions of users.

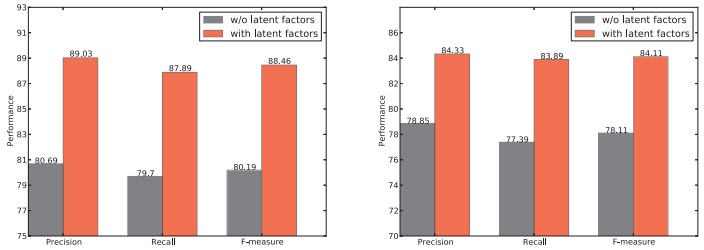


Figure 4: Contribution of latent factors on social action prediction (left) and social tie inference (right).

Algorithms	Learning	Inference
LBP	8.67	8
MCMC	3.45	124
VMF	2.39	7
MGD	0.45	6

Table 4: Efficiency comparison of different optimization algorithms on learning time (hr.) and inference time (sec.).

References

Fabrício Benevenuto, Tiago Rodrigues, Meeyoung Cha, and Virgílio A. F. Almeida. Characterizing user behavior in online social networks. In *Proceedings of Internet Measurement Conference*, pages 49–62, 2009.

Léon Bottou. Stochastic learning. In Olivier Bousquet and Ulrike von Luxburg, editors, *Advanced Lectures on Machine Learning*, Lecture Notes in Artificial Intelligence, LNAI 3176, pages 146–168. Springer Verlag, Berlin, 2004.

Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

Ove Frank and David Strauss. Markov graphs. *Journal of the American Statistical Association (JASA)*, 81(395):832–842, 1986.

Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. Modeling temporal effects of human mobile behavior on location-based social networks. In *Proceedings of CIKM-13*, pages 1673–1678, San Francisco, CA, USA, 2013.

Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transitions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

Jeongwoo Ko, Luo Si, and Eric Nyberg. A probabilistic graphical model for joint answer ranking in question answering. In *Proceedings of SIGIR-07*, pages 343–350, Amsterdam, The Netherlands, 2007.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a social network or a news media? In *Proceedings of WWW-10*, pages 591–600, Raleigh, North Carolina, USA, 2010.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.

Yan Liu, Alexandru Niculescu-Mizil, and Wojciech Gryc. Topic-link LDA: joint models of topic and author community. In *Proceedings of ICML-09*, pages 665–672, Montreal, Canada, 2009.

L. Lovász. Random walks on graphs: A survey. *Combinatorics, Paul Erdős is Eighty*, 2:353–398, 1996.

Hao Ma, Irwin King, and Michael R. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of SIGIR-09*, pages 203–210, Boston, MA, USA, 2009.

Hao Ma, Irwin King, and Michael R. Lyu. Learning to recommend with explicit and implicit social relations. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–19, 2011.

Miller McPherson, Lynn Smith-Lovin, and James M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

Garry Robins, Tom A. B. Snijders, Peng Wang, Mark Handcock, and Philippa Pattison. Recent developments in exponential random graph (p^*) models for social networks. *Social Networks*, 29(2):192–215, 2007.

Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *Journal of Machine Learning Research*, 8:693–723, 2007.

Chenhao Tan, Jie Tang, Jimeng Sun, Quan Lin, and Fengjiao Wang. Social action tracking via noise tolerant time-varying factor graphs. In *Proceedings of KDD-10*, pages 1049–1058, Washington, DC, USA, 2010.

Wenbin Tang, Honglei Zhuang, and Jie Tang. Learning to infer social ties in large networks. In *Proceedings of ECML/PKDD-11*, pages 381–397, Athens, Greece, 2011.

Jie Tang, Tiancheng Lou, and Jon Kleinberg. Inferring social ties across heterogeneous networks. In *Proceedings of WSDM-12*, pages 743–752, Seattle, WA, USA, 2012.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

Stanley Wasserman and Philippa Pattison. Logit models and logistic regressions for social networks: I. An introduction to Markov graphs and p^*. *Psychometrika*, 61(3):401–425, 1996.

Stanley Wasserman, Katherine Faust, Dawn Iacobucci, and Mark Granovetter. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

Xiaofeng Yu and Wai Lam. An integrated probabilistic and logic approach to encyclopedia relation extraction with multiple features. In *Proceedings of COLING-08*, pages 1065–1072, Manchester, United Kingdom, 2008.

Xiaofeng Yu and Wai Lam. Bidirectional integration of pipeline models. In *Proceedings of AAAI-10*, pages 1045–1050, Atlanta, Georgia, USA, 2010.

Xiaofeng Yu and Wai Lam. Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *Proceedings of COLING-10*, pages 1399–1407, Beijing, China, 2010.

Xiaofeng Yu and Wai Lam. Probabilistic joint models incorporating logic and learning via structured variational approximation for information extraction. *Knowledge and Information Systems (KAIS)*, 32:415–444, 2012.

Xiaofeng Yu, Wai Lam, and Bo Chen. An integrated discriminative probabilistic approach to information extraction. In *Proceedings of CIKM-09*, pages 325–334, Hong Kong, China, 2009.

Xiaofeng Yu, Irwin King, and Michael R. Lyu. Towards a top-down and bottom-up bidirectional approach to joint information extraction. In *Proceedings of CIKM-11*, pages 847–856, Glasgow, Scotland, UK, 2011.

Xiaodong Zeng, Derek F. Wong, Lidia S. Chao, and Isabel Trancoso. Graph-based semi-supervised model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL-13*, pages 770–779, Sofia, Bulgaria, 2013.

Discovering Topical Aspects in Microblogs

Abhimanyu Das
Microsoft Research
abhidas@microsoft.com

Anitha Kannan
Microsoft Research
ankannan@microsoft.com

Abstract

We address the problem of discovering topical phrases or "aspects" from microblogging sites like Twitter, that correspond to key talking points or buzz around a particular topic or entity of interest. Inferring such topical aspects enables various applications such as trend detection and opinion mining for business analytics. However, mining high-volume microblog streams for aspects poses unique challenges due to the inherent noise, redundancy and ambiguity in users' social posts. We address these challenges by using a probabilistic model that incorporates various global and local indicators such as "uniqueness", "diversity" and "burstiness" of phrases, to infer relevant aspects. Our model is learned using an EM algorithm that uses automatically generated noisy labels, without requiring manual effort or domain knowledge. We present results on three months of Twitter data across different types of entities to validate our approach.

1 Introduction

Microblogging sites such as Twitter and Weibo are evolving into the social platforms of choice for users to express and discuss, in real-time, their thoughts and ideas on a plethora of subject matters. It is thus important to use these microblog streams to identify the "buzz" or "talking points" regarding any topic or entity of interest, including organizations, products and social issues. This has several applications: For businesses, identifying what its customers are mostly talking about allows them to better engage with their customer base (Burton and Soboleva, 2011; Patino et al., 2012), fine-tune brand awareness and marketing campaigns (Popescu and Jain, 2011), and provide real-time feedback about customer preferences and complaints. Similarly, policy makers and think tanks would benefit from understanding the buzz around various socio-cultural or environmental issues, that could enable them to make well-informed choices and decisions. Infering such key talking points in social media also enables higher layer social-analytics applications such as trend detection, event tracking, and fine-grained opinion and sentiment analysis.

The goal of this paper is to automatically infer such entity-specific buzz in social media, which we represent using key phrases identified from microblog posts about the entity. Following past literature (Kobayashi et al., 2007; Mukherjee and Liu, 2012), we call these topical phrases as *aspects*. Thus, given a stream of microblog posts about an entity of interest, we devise an algorithm that *automatically discovers a ranked list of the top aspects that succinctly represent the buzz or key talking points among users about the entity.* As an illustrating example, Figure 1 shows the top 10 aspects discovered for each month by our aspect discovery algorithm for the Microsoft Surface tablet using 6 month of Twitter data. For each month we depict the key events and news stories (below the timeline) related to the Surface, along with the set of discovered aspects (above the timeline). As seen from the figure, several of the top aspects do not reflect product features or attributes, but instead capture the buzz among Twitter users around recent events or news related to the Surface. For example, the aspects "surface pricing" and "surface preorders" in October refer to the discussions on Twitter following a press release providing details of the Surface pricing and preorder dates. Similarly, the aspect "Oprah tweets" in November corresponds

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 860–871, Dublin, Ireland, August 23-29 2014.

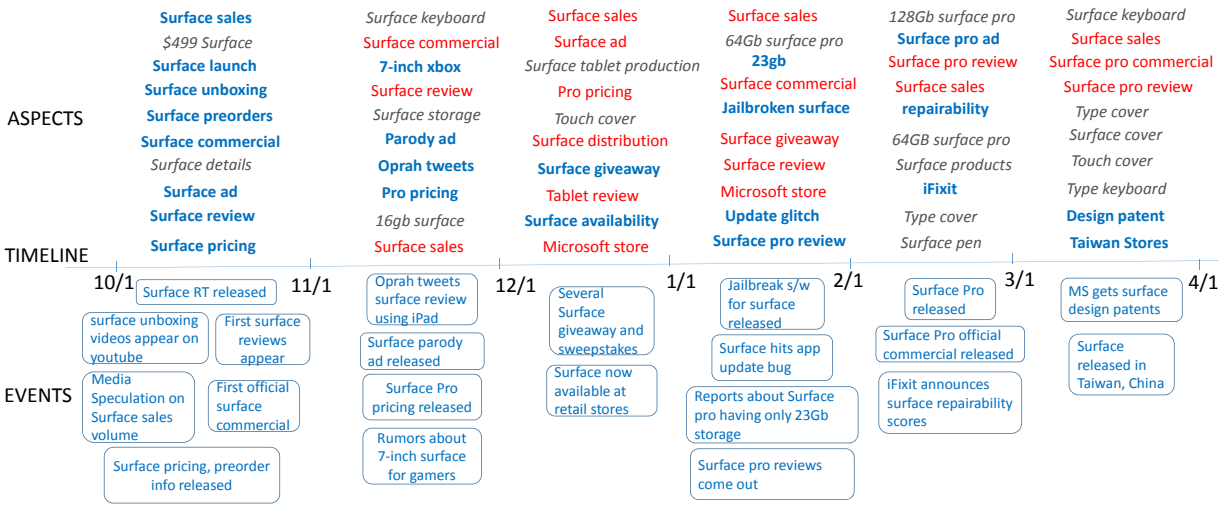


Figure 1: Temporal evolution of top monthly aspects for 'Microsoft Surface' over 6 months (October 2012 to March 2013). The aspects identified by our algorithm (for each month) is shown above the time line, while key events regarding the product is shown below. Aspects that directly relate to the events are shown in bold blue, while aspects that have no bearing to the news are shown in italics. The aspects that were related to an event in previous months but has persisted as an aspect are shown in red using normal font.

to discussions around media coverage of how Oprah Winfrey tweeted a review of the Surface tablet from her iPad. The aspects "iFixit" and "repairability" refer to the unveiling of the iFixit repairability report for the Surface in February. On the other hand, we also see more traditional product feature or attributes that are not correlated to external events but are key discussion points across multiple months, such as "keyboard" or "touch cover".

While there exists a rich line of work (refer to (Liu, 2012) for a comprehensive survey) in aspect identification from customer reviews, blogs or discussion forums, mostly for fine-grained opinion mining for products, there has been little work in the context of aspect discovery from large scale microblog posts. Microblogs pose a unique set of challenges that makes it difficult to directly apply existing methods from prior work. For example, in several papers, frequently occurring noun phrases is used as the building block for detecting aspects (Hu and Liu, 2004a; Hu and Liu, 2004b; Ku et al., 2006). However, for microblogs, frequency of a noun phrase alone is an insufficient indicator of an aspect, due to the inherent noise (unlike reviews, microblog posts are short and often not as focused) and redundancy (*e.g.,* due to retweeting in the context of Twitter). Yet another challenge unique to microblog streams is that the brevity of the posts provide inadequate context and structure. In addition, they are also noisy, with a single tweet often containing both relevant and irrelevant content for a given entity. Due to these reasons, well-known probabilistic approaches (*e.g.,* Topic Models (Mei et al., 2007; Titov and McDonald, 2008) or Conditional Random Fields (Jakob and Gurevych, 2010)), that work well for aspect identification from reasonably long and syntactically well-formed documents such as reviews and blogs, becomes immediately inappropriate in the microblog setting. Additionally, the high volume and velocity of social media streams calls for a scalable, fully automated approach that seamlessly works for a variety of entities and requires no domain-specific knowledge.

We address these challenges inherent in aspect discovery from microblog streams in a principled way: we propose quantifiable indicator measures of "uniqueness", "diversity" and "burstiness" based on insights that are fairly intuitive and yet are generic enough to model the characteristics of relevant aspects for a range of diverse entities. We represent candidate phrases in terms of these three indicators. We propose a probabilistic model for scoring the candidate phrases (§ 2.3) corresponding to an entity of interest. For every entity, the model automatically clusters the indicators and for each cluster, learns relative importance between the indicators for scoring the candidate aspects. Given a collection of <candidate aspects, noisy label> pairs where the noisy label reflects if the corresponding candidate is an aspect

(albeit, noisly), we use an Expectation-Maximization algorithm for training the model. We also present an approach to leverage web search engine results (§ 2.4) to automatically obtain noisy labelled data for any entity. While being entity specific, our approach is highly scalable, entity-agnostic and does not require any manual effort. We validate our results on diverse entities, using *all* tweets from Twitter corresponding to a three month period from January 2013 to March 2013.

Related Work: To the best of our knowledge, the only works related to aspect discovery from microblog posts are (Spina et al., 2012) and (Zhao et al., 2011). In (Spina et al., 2012), four information retrieval functions were compared for identifying aspects from a set of tweets about companies. They showed that a TF-IDF based approach performed the best. Their experiments were however not performed across multiple domains, and used a very small number of tweets for each company. Furthermore, our 'uniqueness' based ranking (§ 2.2) that we use as one baseline is quite similar to their TF-IDF approach, and in our large scale evaluation over diverse domains, we show that TF-IDF or uniqueness alone is not sufficient for efficient aspect discovery (§ 3). The work by (Zhao et al., 2011) proposes an unsupervised approach for keyphrase ranking based on measures of "interestingness" (which is similar to our uniqueness indicator) and "relevance". However, as we show in our experiments (§ 3), the performance of this method is entity–dependent and does not naturally scale to all entities.

The rest of the paper is organized as follows. We describe our algorithm in § 2, including the various indicators that we use to characterize an aspect, the automatic label generation, and our probabilistic model. In § 3, we present experimental results and evaluation of our algorithm along with other baselines on the three month Twitter data set. We conclude in § 4 with remarks on future work.

2 Approach

We formulate the problem of identifying aspects as follows: **Problem statement:** Let e be an entity and s be a time period of interest. We use $\mathcal{T}^s$ to denote the set of all tweets in time period s, and $\mathcal{T}_e^s \subset \mathcal{T}^s$ to be the set of all tweets about e in time period s [1]. Then, we wish to identify the set of k phrases from $\mathcal{T}_e^s$ which are most likely to be valid aspects of e.

Solution overview: Given e, we first identify a set of candidate phrases for aspects from $\mathcal{T}_e^s$ (§2.1). For each phrase, we compute a global indicator, *uniqueness*, that measures how strongly the phrase is correlated with e by comparing its occurrence in $\mathcal{T}_e^s$ and $\mathcal{T}^s$. We also compute two local indicators, *diversity* that measures how diversely the phrase is used in $\mathcal{T}_e^s$, and *burstiness* that measures the temporal activity around the phrase usage in $\mathcal{T}_e^s$ (§ 2.2). We train a probability model (§ 2.3) that captures non-linear relationships between the indicators using a combination of linear decision surfaces. The training labels are obtained using a completely automated approach (§ 2.4). The model is trained independently for each entity, and subsequently used in inferring aspects for the entity during the time period of interest.

2.1 Candidate Aspects

We expect an aspect of an entity to be a phrase on which users can say something subjective. This intuitive requirement is enforced by restricting candidate aspects to be noun phrases (Hu and Liu, 2004b; Popescu and Etzioni, 2007) that are qualified with an adjective within short proximity (around four words) in at least one tweet (Blair-Goldensohn et al., 2008). We use a Twitter-specific part-of-speech tagger (Owoputi et al., 2013) to identify a candidate set of noun phrases in $\mathcal{T}_e^s$ that are used in conjunction with an adjective. After resolving plural nouns to their singular forms, this results in a few thousand candidate phrases per entity for a month of tweets.

2.2 Indicators

We represent a phrase using measurements across three dimensions that captures "diversity", "uniqueness" "burstiness" of usage. These are described in detail below.

[1] While accurately classifying microblog posts to extract posts relevant to an entity is a research problem in itself, this is outside the scope of this work. In this work, we use keyword based classifiers for our entities.

2.2.1 Diversity

Intuitively, a genuine aspect of an entity is more likely to have been discussed on Twitter in the context of that entity, compared to other noun phrases. While one can consider a metric like occurrence frequency (*e.g.,* (Liu, 2012)) to capture this intuition, in microblog settings like Twitter, this can overestimate the importance of a phrase because of redundancy of content due to (a) simple retweeting by followers, (b) multiple users posting the same or very similar content, especially when talking about news and events, and (c) same user posting multiple versions of the same tweets due to automated tweet applications. As an example, the most frequently used noun phrase on Twitter for the entity 'Microsoft Surface' during March was "tablet-a-day giveaway". However, all the tweets containing this phrase referred to a lottery contest that required users to tweet a pre-specified sentence about the Surface. Hence, this phrase cannot be considered a relevant aspect.

We propose factoring out such redundancy by using a notion of "diversity" of content about that aspect. To this effect, for each candidate aspect, its "Diversity" indicator is obtained by computing a score based on the amount of diverse content in the set of tweets about the aspect. To efficiently compute this diversity score, we use the Simhash algorithm (Charikar, 2002) based on Locality Sensitive Hashing (Indyk and Motwani, 1998). Simhash measures the similarity of two tweets t_1 and t_2 by hashing them into small f-bit fingerprints (we use $f = 128$), and comparing the Hamming distance between them. The Locality Sensitive Hash function H used by Simhash ensures that

$$Pr[H(t_1) = H(t_2)] = Sim(t_1, t_2),$$

where $Sim(t_1, t_2)$ is the cosine similarity between t_1 and t_2.

Thus, it suffices to compute a diversity score on the (much smaller) set of 128-bit fingerprints of the tweets containing the aspect. We define this score as the cardinality of the largest subset $S \subset \mathcal{T}_e^s$ of tweets such that the Hamming distance d between the fingerprints of any pair in S is at most 90% of the fingerprint length. While this is a combinatorially hard problem, we use a greedy heuristic to approximate this score using the following steps: 1) Initialize S to a random tweet $r \in T_e^s$. 2) At each iteration, let $t \in \mathcal{T}_e^s \setminus S$ maximize $D(S, t)$. (Here we define $D(S, t) = \min_{x \in S} d(H(x), H(t))$). If $D(S, t) > 0.9$, add t to S. Else return $|S|$.

2.2.2 Uniqueness

Another property of a relevant aspect for an entity is a notion of "uniqueness" to that entity. Intuitively, an aspect should have a higher propensity of being used in tweets about that entity, compared to a generic set of tweets. For example, in the case of Microsoft Surface, several commonly used noun-phrases might have a high frequency of occurrence or a high diversity score such as "news" or "store". However such phrases are arguably too generic to be considered as an aspect of Microsoft Surface. Hence we need to evaluate a candidate noun phrase in terms of its frequency in the set of tweets for that entity, versus its frequency across all tweets in the same time window. In particular, we define the uniqueness indicator of a phrase p in a time period s as:

$$\text{uniqueness}_e^s[p] = \frac{\sum_{t \in \mathcal{T}_e^s} I[\text{p} \in t]}{\sum_{t \in \mathcal{T}^s} I[\text{p} \in t] + \theta}, \tag{1}$$

where $I[\text{p} \in t]$ is an indicator that evaluates to 1 if the tweet t contains the phrase p, and 0 otherwise. θ enforces minimal support (θ tweets from $\mathcal{T}_e^s$) required for p to be unique. We used $\theta = 10$.

Note that this is reminiscent of the tf-idf metric in information retrieval and also used in (Spina et al., 2012); The numerator corresponds to the notion of term-frequency and the denominator to document frequency. This can also be interpreted probabilistically, by considering a bernoulli variable Z that models how unique the phrase is to e. Then the above definition is similar to a maximum-likelihood estimate of Z using a Beta distribution with θ as the prior.

2.2.3 Burstiness

Another indicator of a relevant aspect of an entity is a noun phrase that has an unexpected surge in its frequency of occurrence among tweets of the entity, in a short period of time. This could be due to an

emerging news story, event or talking point about the entity and hence indicate that the phrase is strongly related to the entity, even if the overall frequency of the phrase over a larger time period might be low.

We capture this notion using the "burstiness" indicator. For each candidate noun-phrase, we create a time-series of its occurrences in tweets of the entity within the specific time window. We then use the burst model due to Kleinberg (Kleinberg, 2002) to extract a burstiness score for the noun-phrase. Kleinberg's model uses a finite-state automaton with different states corresponding to different emission frequencies, where state transitions from a low-frequency state to a high-frequency state signify the onset of a burst. We use an R-implementation (url, 2014) of this algorithm on the time series of occurrences of a noun-phrase to identify the corresponding burst levels, and define the burstiness score of the noun-phrase as the sum of these burst levels. For example, the aspect "shipping lanes" detected by our algorithm for the entity "Global Warming" has relatively low frequency of occurrence overall, however it was a topic of intense discussion on Twitter during a week when mainstream news media reported on a PNAS article discussing opening up of new shipping lanes through the Arctic ocean due to global warming(url, 2013).

2.3 Probabilistic model for aspect identification

Given a candidate phrase and its measurements of indicators, we would like to rank these based on a learned model that takes into account these varied interactions between the indicators. One approach is to directly train a linear classifier such as logistic regression using a training set of <phrase,binary label> pairs. As we show in § 3, this approach does not capture the non-linear dependencies among the indicators and the label, resulting in poor performance. In this paper, we jointly model the space of indicator variables and their labels, which we describe next.

2.3.1 Model specification

A candidate phrase is represented by a three-dimensional continuous-valued random variable $\mathbf{x}$, where x_1 corresponds to 'Diversity', x_2 to 'Uniqueness' and x_3 to'Burstiness'. The relationship between these indicators is captured by a probabilistic Gaussian mixture model. Let c be a random variable with discrete distribution over m components. Then,

$$p(\mathbf{x}, c) = p(c)p(\mathbf{x}|c) = \pi_c \mathcal{N}(\mathbf{x}|\mu_c, \Phi_c), \tag{2}$$

where $p(c)$ is a Multinomial distribution with probability π_c for the c^{th} component such that $\sum_c \pi_c = 1$ and $p(\mathbf{x}|c)$ is a Gaussian distribution for c^{th} component with mean vector μ_c and covariance matrix, Φ_c. Let y be the Bernoulli random variable representing whether a phrase is an aspect, such that:

$$p(y = 1|\mathbf{x}, c) = \frac{1}{1 + \exp(-(\mathbf{w}_c^T \mathbf{x} + b_c))}. \tag{3}$$

Then, the joint distribution over the variables is

$$p(\mathbf{x}, y, c) = p(c)p(\mathbf{x}|c)p(y|\mathbf{x}, c) \tag{4}$$

Note that unlike a mixture of logistic regressors (Bishop, 2007), this formulation captures $p(\mathbf{x}|c)$ which is central to modeling the correlation between the indicators. One can view our formulation as a variant of mixture of experts (Jacobs et al., 1991) wherein the gating functions are represented using the posterior over the mixture model components, as opposed to the soft-max function typically used.

2.3.2 Learning

Given a set of training examples,$[\mathbf{X}, \mathbf{y}] = \{\mathbf{x}_n, y_n\}_{n=1}^N$, the model parameters $\{\mu_c, \Phi_c, \pi_c, \mathbf{w}_c, b_c\}_{c=1}^K$ are learned so as to maximize the probability of observations, $p(\mathbf{X}, \mathbf{y})$. Assuming the training examples are independent and identically distributed, we use an Expectation-Maximization algorithm to learn parameters that maximize the probability of observations, $p(\mathbf{X}, \mathbf{y}) = \prod_{n=1}^N p(\mathbf{x}_n, y_n)$ or equivalently, its log:

$$\log p(\mathbf{X}, \mathbf{y}) = \sum_n \sum_c p(c|\mathbf{x}_n, y_n) \log \frac{p(c_n, \mathbf{x}_n, y_n)}{p(c|\mathbf{x}_n, y_n)} \tag{5}$$

$$= \sum_n \sum_c p(c|\mathbf{x}_n, y_n) \log \frac{p(c)p(\mathbf{x}_n|c)p(y_n|\mathbf{x}_n, c)}{p(c|\mathbf{x}_n, y_n)}, \tag{6}$$

where $p(c|\mathbf{x}_n, y_n)$ is the posterior distribution over c. The parameters are learned using the EM algorithm. by iterating between Expectation(E)-step in which $p(c|\mathbf{x}_n, y_n)$ is estimated for each training instance, and the Maximization(M)-step in which parameters of the model are estimated:
E-step: In this step, $p(c|\mathbf{x}_n, y_n)$ is computed for each training instance by taking derivative of eq. 6 with respect to $p(c|\mathbf{x}_n, y_n)$ and setting to zero, so that $p(c = j|\mathbf{x}_n, y_n) \propto p(\mathbf{x}_n, y_n, c = j)$.
M-step: The mixture component parameters (means and covariances) are updated as weighted averages and deviations from the mean, weighted by the posterior computed in the E-step:

$$\mu_c = \frac{\sum_{n=1}^{N} p(c = j|\mathbf{x}_n, y_n)\mathbf{x}_n}{\sum_{n=1}^{N} p(c = j|\mathbf{x}_n, y_n)} \qquad \Phi_c = \frac{\sum_{n=1}^{N} p(c = j|\mathbf{x}_n, y_n)(\mathbf{x}_n - \mu_c)(\mathbf{x}_n - \mu_c)^T}{\sum_{n=1}^{N} p(c = j|\mathbf{x}_n, y_n)} \quad (7)$$

The weight vector,$\mathbf{w}_c$, for each of the logistic component is estimated using the re-weighted least squares (IRLS) algorithm (Bishop, 2007):

$$\mathbf{w}_c = \arg\max_{\mathbf{w}} \sum_n p(c|\mathbf{x}_n, y_n) \log p(y_n|\mathbf{x}_n, c). \quad (8)$$

Scoring Function: Once the model is trained, it is used to score a candidate phrase for being an aspect. For any phrase with indicator vector $\mathbf{x}$, the probability of it being an aspect is given by the convex combination (weighted by $p(c|\mathbf{x})$) of the outputs from all the regressors: $p(y|\mathbf{x}) = \sum_c p(c|\mathbf{x})p(y|\mathbf{x}, c)$

Choice of number of components: The number of components m is a free parameter in our model, and its value is a function of the training dataset. We use Bayesian information criteria (BIC) (Schwarz, 1978) to choose the optimal number of components for training. In particular, we train models by varying K and pick the one with the largest BIC given by $\log p(\mathbf{X}, \mathbf{y}|\theta_m) - \frac{|\theta_m|}{2} \log N$ where θ_m is the model with m components having $|\theta_m|$ parameters, N is the number of data points and $\log p(\mathbf{X}, \mathbf{y}|\theta_m)$ for fixed θ_m is given by eq. 6.

2.4 Automatic generation of training data

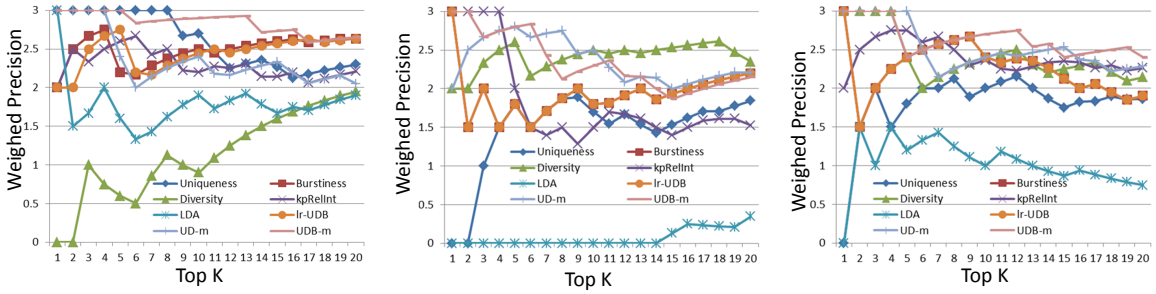
We use a fully automated approach to (noisily) label candidate phrases. The approach is based on the premise that a phrase that is related to the entity and is also popular on the web is more likely to be a potential aspect for the entity. We operationalize this by issuing each phrase to be labeled as a query to a web search engine and retrieve top 50 results. Then, we label it as an aspect if, at least 10% of the top 50 web results have web page titles that are relevant to the entity (determined by the same rules that is used for tweet classification (§ 3.1)) and all the unigrams in the phrase is contained in them.

This approach can result in noisy labels since a candidate phrase that have huge web presence may not be an aspect, and vice versa. In spite of this, we observed reasonable correlation between the propensity of a phrase on Twitter to be a true aspect and the quality of web search result that we can retrieve. Thus, this approach results in generating large noisily labeled datasets, which can often be more effective than a small dataset with high quality labels (Fuxman et al., 2009).

3 Evaluation

We compare our algorithm described in § 2.3 (which we denote *UDB-m*) against the following algorithms: (1) *kpRelInt*: the keyphrase ranking algorithm of (Zhao et al., 2011) applied to our candidate aspects, (2) *lr-UDB*: ranking based on probabilities obtained using a trained, single-component logistic regression model using uniqueness, diversity and burstiness indicators as features, (3) *UD-m*: ranking based on our probabilistic mixture model of § 2.3 where we only used Uniqueness and Diversity indicators but not Burstiness and (4) *LDA*: an approach based on training a 50 component Latent Dirichlet Allocation (Blei et al., 2003) on tweets of that entity, from which we then manually constructed aspects from the best 20 topics.

We also consider rankings based solely on the various indicator scores themselves: uniqueness (*U*), diversity (*D*) and burstiness (*B*), to understand the effectiveness of each of these indicators. Note that *U* is a stronger baseline than the TF-IDF metric (Spina et al., 2012).



(a) Windows 8 (b) Global Warming (c) Microsoft Surface

Figure 2: Weighted Precision across entities

3.1 Dataset

We studied six entities from varied domains including products, environmental issues and personalities:"Windows 8", "Microsoft Surface", "Hyundai", "Organic Food", "Global Warming", and "Tiger Woods". We obtained the set of all English language tweets posted in a three month time period from Jan 1, 2013 to March 31, 2013. For each entity, we classified the tweets pertaining to that entity by using simple keyword-based classifiers. For instance, for the entity 'windows 8', the keywords corresponded to the set {"windows 8", "win8", "windows8", "win 8", "#win8", "#windows8", "#microsoftwindows8"}. In total, we obtained about three million English tweets across all the six entities that we used, with around $100,000$ to $800,000$ tweets for each entity.

Train/Test split: We used data from January to train *UDB-m*, *UD-m* and *lr-UDB* algorithms. We evaluated all algorithms on data from February and March, and obtained qualitatively similar results for both months. We present results only from March, due to space constraints.

3.2 Precision analysis of inferred aspects

The goal of this experiment is to obtain a precision measure for the various algorithms in inferring relevant aspects. Since it is impractical to manually create a ground truth test set of aspects for each entity by inspecting all the tweets, we take an approach used in (Spina et al., 2012). For each entity and month, we pooled together the top 20 aspects identified by all the algorithms under consideration. We used three judges in our organization as human assessors who manually annotated these candidate aspects on a 4-point relevance scale (with '3' being most relevant and '0' being irrelevant to the entity).

Metrics: Let S be the list of top K phrases identified as aspects by an algorithm, with $S[i]$ being the i^{th} phrase. For every phrase $p \in S$, let $R(p) \in [0,3]$ be the average of the relevance rating provided by the three judges. Then, $\text{WeightedPrecision @K}$ of the algorithm at the top K rank is given by $\frac{\sum_{i=1}^{K} R(S[i])}{K}$ (Sakai, 2007). Note that *Weighted Precision @ K* lies in the range [0,3] with higher values indicating that the list of top K aspects is more precise.

Results: Figure 3 shows the Weighted Precision at top K ranks ($K = 1, \ldots, 20$) for each algorithm, averaged across all the entities. For each algorithm and value of K, the marker size of each point in the plot is proportional to the variance in the algorithm's weighted precision. Observe that *UDB-m* consistently has high Weighted Precision scores across all values of K and has the lowest variance, showing its efficacy in discovering aspects with high precision across all entities. Contrast this with the relatively poorer performance of *lr-UDB* that uses a simpler logistic regression model on the same three indicators. This highlights the importance of using a multiple-component mixture model (as opposed to a single component) to capture the non-linear dependencies among the three indicators for an entity. We discuss this further in § 3.4.

The next closest contender after *UDB-m* is *UD-m* that uses only the uniqueness and diversity indicators. The non-trivial gap between *UDB-m* and *UD-m* indicates the importance of incorporting burstiness. The *kpRelInt* algorithm of Zhao et al. (Zhao et al., 2011) actually performs quite poorly. We observed two reasons for this: first, the interestingness score in *kpRelInt* that is based on the ratio of retweets to

tweets does not capture key aspects that may have been frequently used by tweets (but not necessarily retweeted often), and secondly it gives undue importance to words in tweets that are meant to be retweeted by design (for example, as part of a contest, announcement or giveaway). Indeed, the former reason is precisely addressed by our diversity indicator, whose importance is be seen from the fact that among all the three indicators, *D* performs the best.

We note that methods that use only one of the indicators (*U*, *D* and *B*) have large variance in their performance across entities emphasizing the entity-specific nature of these algorithms (we comment on this shortly) making them ill-suited for large scale domain-agnostic applications. Finally, we see that *LDA* performs the worst among our baselines, due to the inherent brevity, ambiguity and noise in tweets.

Entity-specific analysis: Consider Figure 2 that compares entity-specific performances of the algorithms considered. Figure 2a shows their performance for 'Windows 8'. For this entity, *U*, *UD-m* and *UDB-m* all perform equally well for small values of K, however the performance of *UDB-m* stays stable even for large values of K, while that of *U* and *UD-m* deteriorate. Contrast the relatively poor performance of *B* for Windows 8 with its performance for 'Global Warming' in Figure 2b. We see that the precision of *UDB-m*, which is still higher than most of the other algorithms, aligns with that of *B* for small K. This is due to the inherent nature of this entity, for which much of the chatter on Twitter tends to revolve around major news events. We discuss this in more detail in § 3.5. *UD-m*, which performed very well for the Windows 8 entity, does not have as good precision in this case, because it does not factor in this important effect of burstiness. Figure 2c shows the performance of the algorithms for the 'Microsoft surface' tablet. Again, *UDB-m* mostly outperforms the other methods across the range of K values, but is matched by *UD-m* and, to a lesser extent, by *D* for small K. Burstiness no longer plays such an important role - the tweets for Surface in March tend to be mostly comments on the features, commercials and accessories related to the product, and not so much related to news.

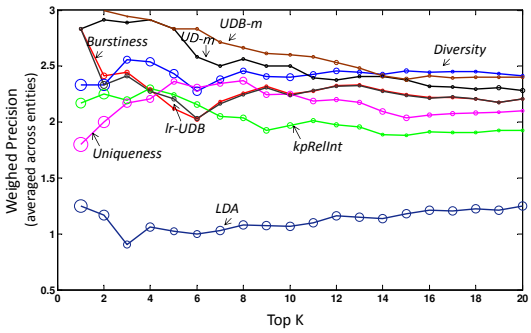


Figure 3: Average and variance (over all entities) for Weighted Precision for various algorithms. (Best viewed in color)

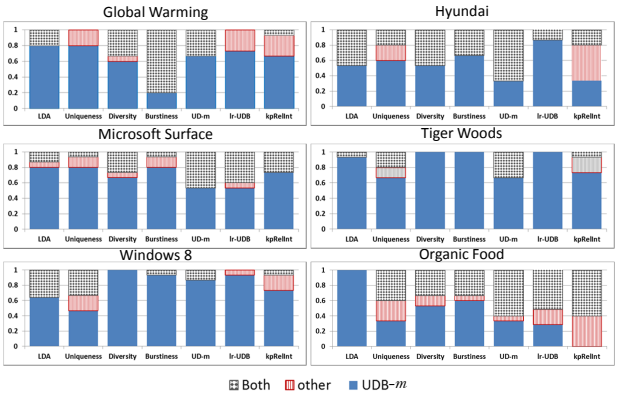
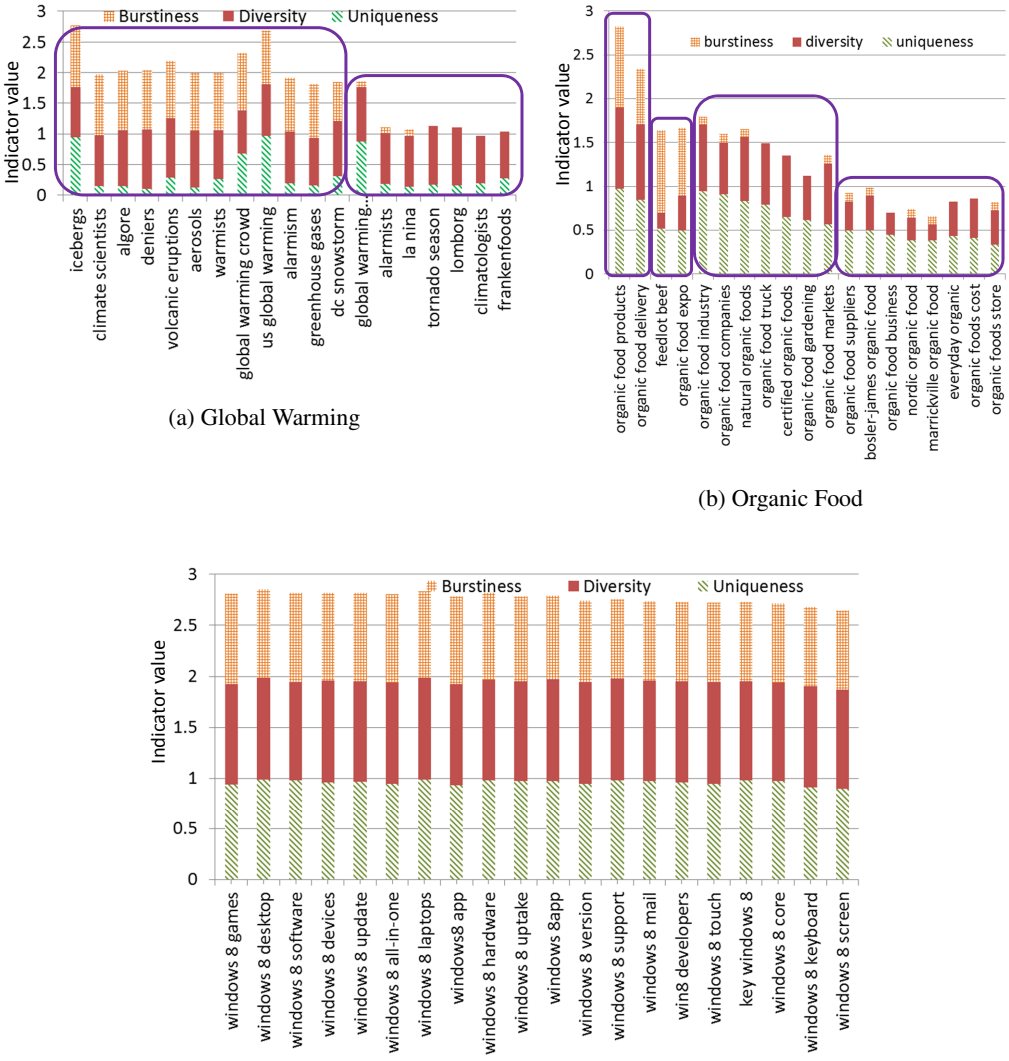


Figure 4: Pairwise preferences (at top 10) between *UDB-m* and other algorithms studied

3.3 Pairwise Preference comparison of inferred aspects

Here, we quantify the overall precision of the ranked list of aspects identified by various algorithms. We conduct pairwise evaluation using Amazon Mechanical Turk. Each Human Intelligence Tasks (HIT) consists of a pair of top 20 ranked aspect lists for an entity, with one list from *UDB-m* and the other chosen from one of the baseline algorithms. For each pair, we randomly permuted the order for each HIT (considered 5 random orderings). Each pair was judged by 5 judges, resulting in 25 judgments for each <entity, *UDB-m*, baseline-algorithm> triplet. Each judge was asked to study the two lists and specify which of the two was more relevant (or choose "Both are comparable"). Since the judges do not have access to the tweets, they were given instructions to perform a web search using the aspect and the entity name as a query string, restricted to the appropriate month. They were then asked to use the search results to guide them in determining which of the ranked lists was more relevant. We computed the Fleiss-κ inter-annotator agreement across each entity and method to be 0.68 on average, showing substantial agreement among the judges.



(a) Global Warming

(b) Organic Food

(c) Windows 8

Figure 5: Strengths of indicators for top 20 aspects (sorted according to the components)

Results: Figure 4 plots the results of the pairwise preference evaluations for all the entities for the month of March. Each of the seven bars corresponds to the preference results obtained by comparing *UDB-m* versus one of *lr-UDB*, *kpRelInt*, *LDA*, *UD-m*, *U*, *D* and *B*. For each pair, we plot the fraction of the 25 judges that 1) voted for *UDB-m* 2) voted for the other algorithm, or 3) answered "both are comparable". We observe that for almost all the entities, the fraction of votes for *UDB-m* was higher than the votes for the other algorithms. The only exception was organic food for which *kpRelInt* performed better than *UDB-m*. There were some cases for which a majority of judges said both aspect lists were comparable. These cases mostly involved comparisons of *UDB-m* versus *UD-m* which suggests that after *UDB-m* the next best performing algorithm was *UD-m* (We had observed the same effect in precision analysis § 3.2). Another case where both aspect lists are comparable was *UDB-m* vs *B* for the entity 'global warming'. As described in § 3.5, the buzz around this entity is often centered around news events and hence the top 10 aspects identified by our algorithm coincides closely with burstiness of the phrases. Hence, *UDB-m* and *B* perform quite comparably.

3.4 Importance of Multiple Components

UDB-m uses multiple components to model the data, with actual number of components inferred using BIC. Here, we demonstrate the importance of using varied number of components for each entity. Figure 5 shows the top 20 aspects identified for three of the entities. For ease of exposition, we group

Global Warming	Hyundai	Microsoft Surface	Tiger Woods	Windows 8	Organic Food
volcanic eruptions	hyundai santafe	surface keyboard	tiger woods commercial	windows 8 games	organic food delivery
deniers	hyundai sonata	surface sales	us skier	windows 8 desktop	organic food truck
aerosols	starex	surface pro commercial	tiger woods #2	windows 8 software	certified organic foods
tornado season	fuel cell	surface pro review	cadillac championship	windows 8 devices	organic food business
al gore	hyundai genesis	type cover	arnold palmer invitational	update windows 8	everyday organic
lomborg	tucson	surface review	tiger woods #3	windows 8 all-in-one	organic food gardening
global warming awareness	r-spec	touch cover	tiger woods number	windows 8 laptops	organic foods cost
us global warming	i-deal	type keyboard	tiger woods video	windows8 app	cafe bahrain
dc snowstorm	elantra	benchmarks surface	tiger woods house	windows 8 hardware	nordic organic food
icebergs	entourage	surface tablet line	skier lindsey vonn	windows 8 uptake	organic food industry

Table 1: Top 10 aspects identified by our algorithm for various entities

aspects list based on which component they belonged to (the one with largest posterior probability). For each aspect, we show a stacked bar representing the values for the three indicators (hence the maximum length of the bar is 3).

Consider Figure 5a corresponding to 'Global Warming'. Here, only a two component model was trained: one to model large values for diversity and burstiness, and another to model large values for diversity. While highly bursty and diverse aspects such as 'volcanic eruptions' are explained by the component that captures large diversity and burstiness values, aspects such as "tornado season" and "lomborg" that are widely discussed in diverse contexts but are not bursty are captured in another component.

Contrast this with Figure 5b for 'organic food'. This entity was automatically trained using a six component model out of which four participated in identifying the top 20 aspects. The aspect 'organic food products' from the first component has large values for all three indicators. In contrast, the aspects 'organic food truck' and 'organic food gardening' from the third component have high uniqueness and diversity values but low values for burstiness, indicating that they are consistently talked about through the month. The aspect 'feedlot beef' in the second component has low values for diversity, but has large values for burstiness indicating a spike in chatter around feedlot beef in the context of organic food.

Figure 5c shows the corresponding plot for 'Windows 8'. All the top aspects come from a single component. While one may be tempted to use only one component for this entity, our model used six components in order to explain the high variance in the training data. The remaining components were useful in weeding out noise. This can also be seen from the improved performance of *UDB-m* in comparison to *lr-UDB* which uses only a linear classifier (Figure 2a).

3.5 Qualitative Results

Table 1 shows the top 10 aspects identified by our algorithm for the month of March 2013. Consider the entity, 'global warming'. In Twitter, we found that discussions around this entity were highly news (or event) driven and this is reflected in the identified aspects. For instance, 'volcanic eruptions' and 'aerosols' corresponds to news reports of a study that showed how aerosols from modest volcanic eruptions may mask global warming effects. The aspect 'lomborg', referring to Bjorn Lomborg was the subject of much discussion in March; With his article in WSJ on heavy carbon–di–oxide emissions from electric cars charging, Lomborg created a stir among environmentalists.

In contrast, the top ranking aspects for Hyundai on Twitter corresponded mostly to chatter about various car models. Hyundai's announcement in March of their intention to offer fuel-cell cars in the US led to a lot of buzz around this topic, as aptly identified by the aspect 'fuel cells'.

There were three major events in March about Tiger Woods that created buzz on Twitter (and mainstream news media): his Cadillac Championship performance that led to his regaining the number one spot in golf, his relationship with the US skier Lindsey Vonn, and his rivalry with Graeme McDowell during the Cadillac championship. All these events are identified as aspects for the entity 'Tiger Woods'.

4 Concluding Remarks

In this paper, we studied the problem of inferring the key talking points or *aspects* about entities from microblog streams. We presented a probabilistic model to automatically infer these aspects from microblog streams for any specified domain, with *no* manual effort or domain knowledge about the entity.

We presented indicators such as "uniqueness", "diversity" and "burstiness" to capture characteristics of aspects in the microblog context. Our large scale empirical evaluation over three months of Twitter data for entities from various categories validated the efficacy of our approach.

A key direction for future work is the problem of clustering semantically similar aspects pertaining to an entity (*e.g.,* 'volcanic eruptions' and 'aerosols' for 'global warming') to get a more succinct representation of the aspects. Another line of work is to leverage the temporality of these aspects in building temporal aspect discovery models.

References

[Bishop2007] Christopher Bishop. 2007. *Pattern Recognition and Machine Learning*. Springer.

[Blair-Goldensohn et al.2008] Sasha Blair-Goldensohn, Kerry Hannan, Ryan McDonald, Tyler Neylon, George A Reis, and Jeff Reynar. 2008. Building a sentiment summarizer for local service reviews. In *WWW Workshop on NLP in the Information Explosion Era*.

[Blei et al.2003] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

[Burton and Soboleva2011] Suzan Burton and Alena Soboleva. 2011. Interactive or reactive? marketing with twitter. *Journal of Consumer Marketing*, 28(7):491–499.

[Charikar2002] Moses S Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM.

[Fuxman et al.2009] Ariel Fuxman, Anitha Kannan, Andrew B Goldberg, Rakesh Agrawal, Panayiotis Tsaparas, and John Shafer. 2009. Improving classification accuracy using automatically extracted training data. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1145–1154. ACM.

[Hu and Liu2004a] Minqing Hu and Bing Liu. 2004a. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

[Hu and Liu2004b] Minqing Hu and Bing Liu. 2004b. Mining opinion features in customer reviews. In *AAAI*, volume 4, pages 755–760.

[Indyk and Motwani1998] Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM.

[Jacobs et al.1991] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

[Jakob and Gurevych2010] Niklas Jakob and Iryna Gurevych. 2010. Extracting opinion targets in a single-and cross-domain setting with conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1035–1045. Association for Computational Linguistics.

[Kleinberg2002] Jon Kleinberg. 2002. Bursty and hierarchical structure in streams.

[Kobayashi et al.2007] Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *EMNLP-CoNLL*, pages 1065–1074.

[Ku et al.2006] Lun-Wei Ku, Yu-Ting Liang, and Hsin-Hsi Chen. 2006. Opinion extraction, summarization and tracking in news and blog corpora. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 100–107.

[Liu2012] Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1).

[Mei et al.2007] Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of the 16th international conference on World Wide Web*, pages 171–180. ACM.

[Mukherjee and Liu2012] Arjun Mukherjee and Bing Liu. 2012. Aspect extraction through semi-supervised modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 339–348. Association for Computational Linguistics.

[Owoputi et al.2013] Olutobi Owoputi, Brendan OConnor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL-HLT*, pages 380–390.

[Patino et al.2012] Anthony Patino, Dennis A Pitta, and Ralph Quinones. 2012. Social media's emerging importance in market research. *Journal of Consumer Marketing*, 29(3):233–237.

[Popescu and Etzioni2007] Ana-Maria Popescu and Orena Etzioni. 2007. Extracting product features and opinions from reviews. In *Natural language processing and text mining*, pages 9–28. Springer.

[Popescu and Jain2011] Ana-Maria Popescu and Alpa Jain. 2011. Understanding the functions of business accounts on twitter. In *Proceedings of the 20th international conference companion on World wide web*, pages 107–108. ACM.

[Sakai2007] Tetsuya Sakai. 2007. On the reliability of information retrieval metrics based on graded relevance. *Information Processing & Management*, 43(2):531–548.

[Schwarz1978] Gideon Schwarz. 1978. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464.

[Spina et al.2012] Damiano Spina, Edgar Meij, Maarten de Rijke, Andrei Oghina, Minh Thuong Bui, and Mathias Breuss. 2012. Identifying entity aspects in microblog posts. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1089–1090. ACM.

[Titov and McDonald2008] Ivan Titov and Ryan McDonald. 2008. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM.

[url2013] 2013. `http://phys.org/news/2013-03-global-unexpected-shipping-routes-arctic.html`.

[url2014] 2014. `http://cran.r-project.org/web/packages/bursts/index.html`.

[Zhao et al.2011] Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee Peng LIM, and Xiaoming Li. 2011. Topical keyphrase extraction from twitter. In *ACL*. ACM.

Utilizing Microblogs for Automatic News Highlights Extraction

Zhongyu Wei
The Chinese University of Hong Kong
Shatin, N.T.
Hong Kong
zywei@se.cuhk.edu.hk

Wei Gao
Qatar Computing Research Institute
Qatar Foundation
Daha, Qatar
wgao@qf.org.qa

Abstract

Story highlights form a succinct single-document summary consisting of 3-4 highlight sentences that reflect the gist of a news article. Automatically producing news highlights is very challenging. We propose a novel method to improve news highlights extraction by using microblogs. The hypothesis is that microblog posts, although noisy, are not only indicative of important pieces of information in the news story, but also inherently "short and sweet" resulting from the artificial compression effect due to the length limit. Given a news article, we formulate the problem as two rank-then-extract tasks: (1) we find a set of indicative tweets and use them to assist the ranking of news sentences for extraction; (2) we extract top ranked tweets as a substitute of sentence extraction. Results based on our news-tweets pairing corpus indicate that the method significantly outperform some strong baselines for single-document summarization.

1 Introduction

People in this era are overloaded by their daily exposure to large amount of online information. To make life easier, some news websites like CNN.com and USAToday.com provide "Story Highlights" in their news articles for readers to get the gist of story quickly. The highlights of an article typically contain 3-4 summary sentences in bullet-points form that are representative of and shorter than the original new sentences in the article. An example of story highlights of an article is shown in Figure 1 (marked in red rectangle) that are written in a compact, almost telegraphic style. In contrast to the original content of the article, significant compression is obtained by shortening and paraphrasing.

Unfortunately, the production of such good-quality highlights needs to be done manually which is very expensive. Existing methods face grand technical challenges for automating the process. The task is complex in nature due to a broad range of linguistic constraints which ultimately requires wide-coverage of language understanding beyond the capabilities of current NLP technology (Woodsend and Lapata, 2010). Most automatic systems simplify the problem using extractive approach. By using linguistic or statistical information or both, the key units or concepts can be identified from sentences or across multiple documents, and then the sentences are scored and extracted according to their informativeness with the presence of the key components.

The extractive approach has two salient problems: (1) it is commonly ineffective to locate key sentences, meaning that the presence of linguistically and/or statistically important units does not necessarily indicate a highlight sentence. This is evidenced by the fact that sophisticated systems for Document Understanding Conference (DUC) summarization task cannot significantly outperform a trivial baseline that simply selects first n sentences of the document (Nenkova, 2005); (2) sentence extracts as highlights are extraordinarily verbose in general, which need to be post-processed for substantial compression. But sentence compression may breach the readability or grammaticality (Clarke and Lapata, 2008).

With the popularity of social media, online news providers are moving towards offering more interaction with news readers via microblogging service like Twitter. Many Twitter users also post tweets

Work conducted at Qatar Computing Research Institute (QCRI) when the first author was employed as an intern

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 872–883, Dublin, Ireland, August 23-29 2014.

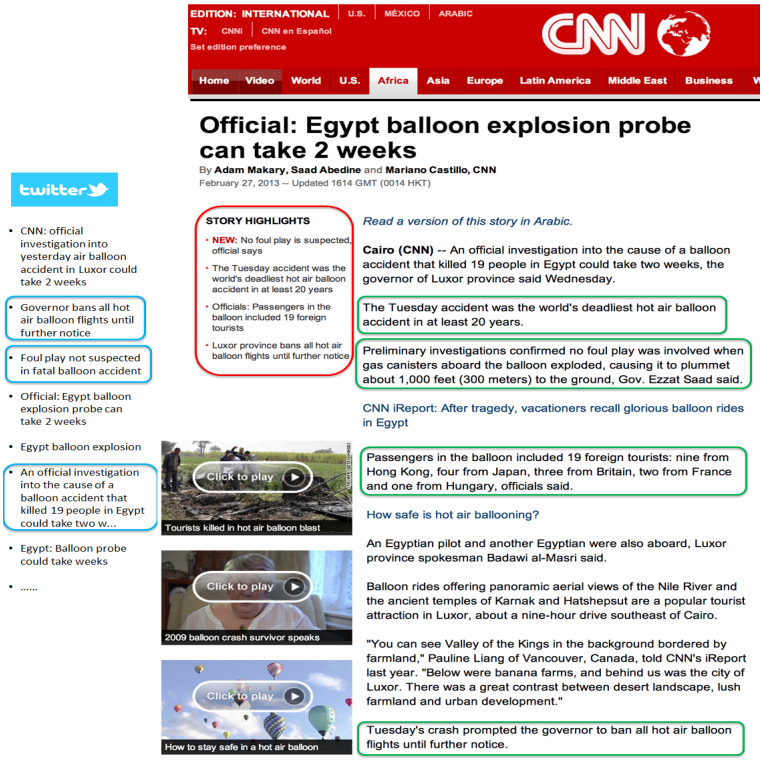


Figure 1: A CNN news article with story highlights (Highlights are marked by red rectangle, and the news sentences related to the highlights are enclosed in green rectangles) and some relevant tweets one can observe independently on Twitter (marked by light blue rectangles on the left)

about news together with their URLs. Such increased cross-media interaction recasts the role of different information sources that are useful for this task in a sense that interesting correlations between the news and relevant microblogs could be captured and leveraged to boost the performance.

To address these considerations, we make two hypotheses based on our observation that can be crucial to highlights extraction. (1) *Indicative effect:* microblog users' mentioning about the pieces of news is indicative of the importance of the corresponding sentences; (2) *Human compression effect:* important portions of a news article have been rewritten by microblog users in a more condensed style owing to length limit. Accordingly, we formulate our problem as two independent rank-then-extract tasks: firstly, we find a set of indicative tweets and use them to assist the ranking of news sentences for extraction; secondly, we extract top-ranked tweets (with the help of news sentences) as a substitute of sentences extraction since they are typically shorter. Based on our news-tweets pairing corpus, the results of experiments following both directions indicate that our methods outperform some strong baselines for single-document summarization.

2 Related Work

Our work intersects the summarization of single document and microblogs. Single-document summarization has been studied for years starting from Luhn and Peter (1958). Based on local content information of a document (Wong et al., 2008; Barzilay et al., 1997; Marcu, 1997), researchers proposed various statistical or semantic approaches using classification (Wong et al., 2008), Integer Linear Programming (ILP) (Li et al., 2013), sequential models (Shen et al., 2007) and graphical models (Litvak and Last, 2008; Hirao et al., 2013). For the concision of summary, sentence compression or word deletion was used (Knight and Marcu, 2002) for preprocessing. Joint models combining compression and selection of sentences were also studied (Woodsend and Lapata, 2010; Li et al., 2013).

Summarizing microblog content is to distill the large quantities of tweets into a concise and representative description of a target event. Sharifi et al. (2010) proposed a graph-based phrase reinforcement

algorithm (PRA) to generate a one-sentence summary from a collection of tweets. By using linguistic features, Judd and Kalita (2013) improved the performance of PRA. Sharifi et al. (2010) and Inouye et al. (2011) presented a hybrid TF-IDF approach for extracting tweets with the presence of important terms. More fine-grained summarization was proposed by considering sub-events and combining the summaries extracted from each sub-topic (Nichols et al., 2012; Zubiaga et al., 2012; Duan et al., 2012).

The research for coupling news and microblogs attracted much attention recently. Subašić and Berendt (2011) and Zhao et al. (2011) independently compared tweets to online news to identify features for news detection in tweets. Phelan et al. (2011) used tweets to recommend news articles based on user preferences. Gao et al. (2012) produced cross-media news summaries by capturing the complementary information from both sides. Kothari et al. (2013) and Štajner et al. (2013) investigated detecting news comments from Twitter for extending news information provided. Guo et al. (2013) proposed a graphical model to identify news for a given tweet to provide contextual support for NLP tasks.

Some work attempted to use different kinds of resources to help document summarization, such as Wikipedia and query log of search engine (Svore et al., 2007), clickthrough data (Sun et al., 2005), users' comments on news (Hu et al., 2008), and social media context of the articles (Yang et al., 2011). Our work is closely related to Svore et al. (2007) that considered incorporating third-party resource in the ranking process, but the access to query logs is extremely limited, and Wikipedia content is relatively static which cannot reflect timely information like social media.

We also share the same testbed with Woodsend and Lapata (2010). They selected and compressed news sentences with a joint model using ILP by considering phrase as basic extract element. Their method requires a large training corpus for deriving accurate salient scores of phrases, and also the feasible solution of ILP model with hard constraints does not necessarily exist.

Yang et al. (2011) proposed a unified supervised model called dual wing factor graph to simultaneously summarize Web documents and tweets based on structural mining from social context. Despite of similar motivation, our work has some key differences from theirs: (1) Our ground-truth come from standard news highlights, and our target summary keeps consistent no matter which source of information our highlights are extracted from. They built ground-truth summaries separately for each side by manually choosing no less than 5 tweets and 10 news sentences. So, our standard is more difficult to reach since our ground-truth summaries are not extracts of the original sentences or tweets; (2) Our approach is very different. We use ranking-based algorithm which is more adequate than their classification approach because there are much fewer positive candidates than negative ones, and the class distribution is very imbalanced (like information retrieval tasks). Also, they were focused on mining the implicit structural information from retweeting and user following networks, while we focus on content-based correlations.

3 Corpus Construction

There is no news-tweets coupling data set publicly available for the purpose of news highlights production[1]. We constructed the first of such corpus for this application by our own, for which an event-oriented strategy was adopted to collect the highlights-document-tweets couplings by using a social search engine. We manually identified 17 salient news events taking place in recent two years. For each event, we manually generated a set of core queries which were used to retrieve the relevant tweets via Topsy[2] search API. Then we gathered the retrieved tweets containing embedded URLs that point to the news articles on CNN and USAToday websites that provide story highlights, and extracted the content of the news articles and the associated highlights.

For each article, we collected all the tweets in the retrieved tweet set above that contain links to the article to form our highlights-document-tweets couplings based on the following rules: (1) We delete those extremely short tweets with less than 5 tokens and the tweets that are suspected copies from news title and highlights. For example, we try our best to remove all the suspectable tweets including the cases

[1]We realize the news-tweets coupling data set released recently for NLP tasks by Guo et al. (Guo et al., 2013). However, this data set is not suitable for our task for two reasons: (1) There are 12,704 news articles but only 34,888 tweets. Although part of the news are from CNN which contain story highlights, the number of tweets per article is too limited, not to mention finding useful candidates; (2) The full text of news content is not provided, with only the first few sentences of articles instead.

[2]`http://topsy.com`

	Documents	Highlights	Tweets
Total #	121	455	78,419
Sentence # per news	53.6±25.6	3.7±0.4	648.1±1161.7
Token # per news	1123.0±495.8	49.6±10.0	10364.5±24749.2
Token # per sentence	21.0±11.6	13.2±3.2	16.0±5.3

Table 1: Overview statistics on the corpus (mean and standard deviation)

Event	Doc #	Highlight #	Tweet #	Event	Doc #	Highlight #	Tweet #
Aurora shooting	14	54	12,463	African runner murder	8	29	9,461
Boston bombing	38	147	21,683	Syria chemical weapons use	1	4	331
Connecticut shooting	13	47	3,021	US military in Syria	2	7	719
Edward Snowden	5	17	1,955	DPRK Nuclear Test	2	8	3,329
Egypt balloon crash	3	12	836	Asiana Airlines Flight 214	11	42	8,353
Hurricane Sandy	4	15	607	Moore Tornado	5	19	1,259
Russian meteor	3	11	6,841	US Flu Season	7	23	6,304
Chinese Computer Attacks	2	8	507	Williams Olefins Explosion	1	4	268
cause of the Super Bowl blackout	2	8	482	Total	121	455	78,419

Table 2: Distribution of documents, highlights and tweets with respect to different events

like "RT @someone HIGHLIGHT URL"; (2) If there are more than 100 tweets linked to an article, the article is kept, otherwise the artcile is removed. Note that using explicit hyperlinks is not the only way for identifying the couplings but the most straightforward one. Here we simply resort to this straightforward method to build the corpus for verifying our two hypotheses raised in Section 1. Thorough investigation on the construction of an enhanced highlights-oriented coupling corpus is left for our future work.

The statistics of the resulted corpus are given in Table 1 which is also made accessible[3]. As shown in the table, the average number of relevant tweets to a document is about 648. Since some of the events are much more popular than others, the standard deviation of the number of tweets associated with a document is as high as 1,162. The highlights are characterized as high compression rate compared to the length of news articles. In addition, a single highlight sentence on average is only 2/3 the length of a news sentence, and more interestingly the average length of tweets is very close to that of highlight sentences, which suggests that the relevant tweets can be a reasonable source of candidates for extraction.

Table 2 shows the distribution of documents, highlights and tweets with respect to the 17 news events we collected.

4 Our Approach

Given a news article containing n sentences $S = \{s_1, s_2, ...s_n\}$ and a set of m relevant tweets $T = \{t_1, t_2, ..., t_m\}$, we aim to extract x sentences from the set S or the same number of tweets from set T as highlights covering the main theme of the article. We define the two tasks as follows:

- **Task 1 – sentences extraction:** Given auxiliary T, extract x elements $H(S) = \{s^{(1)}, s^{(2)}, ..., s^{(x)} | s^{(i)} \in S, 1 \leq i \leq x\}$ from S as highlights.
- **Task 2 – tweets extraction:** Given auxiliary S, extract x elements $H(T) = \{t^{(1)}, t^{(2)}, ..., t^{(x)} | t^{(i)} \in T, 1 \leq i \leq x\}$ from T as highlights.

Most single-document summarization methods (Woodsend and Lapata, 2010; Yang et al., 2011) treat the extraction as a classification problem which assigns either positive or negative label to the extract candidates. We argue that it is more adequate to model it as a ranking problem because there is far more unsuitable candidates than suitable ones for being the highlights. Such kind of imbalanced class distribution makes classification a secondary solution.

Our model learns to rank all the candidate sentences in task 1 or candidate tweets in task 2, and then extracts the top-x ranked instances as output highlights. We adopt an effective pair-wise ranking model RankBoost (Freund et al., 2003) for that using the RankLib package[4]. RankBoost takes pairs of instances

[3] http://www1.se.cuhk.edu.hk/~zywei/data/hilightextraction.zip

[4] http://sourceforge.net/p/lemur/wiki/RankLib/

Category	Name	Description
Local Sentence Feature (LSF)	IsFirst	Whether s is the first sentence in the news
	Pos	The position of s in the news
	TitleSimi	Token overlap between s and news title
	ImportUnigram	Importance score of s according to the unigram distribution in the news
	ImportBigram	Importance score of s according to the bigram distribution in the news
Local Tweet Feature (LTF)	Length	Token number in t
	HashTag	HashTag related features (presence and count)
	URL	URL related features (count)
	Mention	Mention related features (presence and count)
	ImportTFIDF	Importance score of t based on unigram Hybrid TF-IDF algorithm (Sharifi et al., 2010)
	ImportPRA	Importance score of t based on phrase reinforcement algorithm (Sharifi et al., 2010)
	TopicNE	Named entity related features (NE count and seven binary values indicating the presence of each category)
	TopicLDA	LDA-based topic model features (maximum relevance with sub-topics, etc.)
	QualityOOV	Out-of-vocabulary words related features (count and percentage)
	QualityLM	Quality score of t according to language model (Unigram, bigram and trigram)
	QualityDepend	Quality score of t according to dependency bank (Han and Baldwin, 2011)
Cross-Media Feature (CCF)	MaxCosine	Maximum cosine value between the target instance and auxiliary instances
	MaxROUGE1F	Maximum ROUGE-1 F score between the target instance and auxiliary instances
	MaxROUGE1P	Maximum ROUGE-1 precision value between the target instance and auxiliary instances
	MaxROUGE1R	Maximum ROUGE-1 recall value between the target instance and auxiliary instances
	LeadSenSimi*	ROUGE-1 F score between leading news sentences and t
	TitleSimi*	ROUGE-1 F score between news title and t
	MaxSenPos*	The position of sentences that obtain maximum ROUGE-1 F score with t
	SimiUnigram	Similarity based on the distribution of (local) unigram frequency in the auxiliary resource
	SimiUniTFIDF	Similarity based on the distribution of (local) unigram TF-IDF in the auxiliary resource
	SimiTopEntity	Similarity based on the (local) presence and count of most frequent entities in the auxiliary resource
	SimiTopUnigram	Similarity based on the (local) presence and count of most frequent unigrams in the auxiliary resource

Table 3: Feature description (t: a tweet; s: a news sentence; *: features used in task 2 only)

(I_i, I_j) as input for training and their preference order as labels. In our case, instance pair can be the pair of sentences or tweets, and the pairwise order is determined by the salient score of each instance that is the maximum ROUGE-1 (Lin, 2004) F-value between the instance and the corresponding ground-truth highlight sentences. Given the gold standard highlights $H^g = \{h_1, h_2, ..., h_x\}$, the salient score of an instance is calculated as $score(I_i) = max_k\{\text{ROUGE-1}(I_i, h_k)\}$.

Note that in task 2 the number of tweets pairs generated in training can be extremely large because of the number of tweets in popular topical news articles (see Table 2) that may degrade the efficiency of training. Some ad-hoc workaround is employed to make the problem tractable. As opposed to using all the possible pairs, we divide the tweets into b bins, where the bins are bounded by continuous ranges of salient scores. We fix the length of different ranges by fitting the distributions of salient score values. Tuned on a subset with 20% randomly selected training instances, the value of b is determined as 4. Then, the pairs are formed across these brackets.

5 Feature Design

The feature space of the two tasks are designed to intersect at the cross-media correlation part. The local features describe the instance to be ranked (i.e., either a news sentence or a tweet), and the cross-media correlation features capture the similarity of the instance with the counterparts in the auxiliary resource.

The features consist of three subsets of informativeness measures including local sentence features (LSF), local tweet features (LTF) and cross-media correlation features (CCF). In task 1, we can use LSF or both LSF and CCF for rank learning; and in task 2, we can use LTF or combine LTF and CCF. The full feature list is described in Table 5. For local sentence features, we implement the 5 document features defined in (Svore et al., 2007) for single-document summarization task. This is for the ease of comparison with the existing approach. In this section, we will only describe the local tweet features and the cross-media correlation features in more detail.

5.1 Local Tweet Features

Local tweet features are proposed to capture the importance of a tweet based on local information in three aspects, including twitter-specific, topic-related, and writing-quality measures.

5.1.1 Twitter-specific measures

Twitter-specific features indicate the basic content-based characteristics of a tweet such as length, the characteristics specifically provided by Twitter platform such as hashtags, mentions and embedded urls, and two scoring functions used by state-of-the-art tweet summarization algorithms including Hybrid TF-IDF (Sharifi et al., 2010) and PRA (Sharifi et al., 2010). Hybrid TF-IDF is a variant of traditional TF-IDF weighting for tweets collection which treats each tweet as a document when computing IDF while the whole tweets set as a document when computing TF. We calculate the feature *ImportTFIDF* of a tweet based on the TF and IDF values of its tokens. PRA is a phrase reinforcement algorithm that can produce a one-sentence summary for a given tweets set. We follow the idea of PRA to generate the token graph of our tweets set and compute the weight for each token node. We then measure the importance of a tweet by summating the weights of all its tokens, which becomes the *ImportPRA* feature.

5.1.2 Topic-related measures

Topic-related features are used to capture important tweets based on the topical information embodied by named entities (NE) or latent topic semantics. *TopicNE* is proposed to utilize NE as indicator for describing an event. We resort to Stanford Name Entity Recognizer[5] to extract seven types of named entities including time, location, organization, person, money, percent and date. Based on that, we count entities in the tweet, and then obtain seven additional binary values indicating the presence of each category. *TopicLDA* is used to capture sub-topics. Intuitively, if a tweet is highly related to some sub-topic in the event, it is more important. We use LDA (Blei et al., 2003) to identify the sub-topics in the tweets set. Based on the resulted sub-topics and term distribution, we first calculate the maximum relevance value between the tweet and all sub-topics as a feature. Then, we obtain the distribution of relevance values of the tweet with respect to all sub-topics and compute the entropy of this distribution as another feature. The lower the entropy is, the higher the degree of topical concentration for the tweet. We use the default setting of the toolkit mallet[6] and set the number of sub-topics as 10 empirically.

5.1.3 Writing-quality measures

Writing-quality features indicate if a tweet is written in a formal way. Intuitively if more formally a tweet is written, it is more likely to be extracted. *QualityOOV* measures to what extent a tweet contains out-of-vocabulary (OOV) tokens. We simply calculate the number and the percentage of the OOV words in the tweet as features[7]. *QualityLM* measures writing quality of a tweet based on language model. We train uni-gram, bi-gram and tri-gram language models using maximum-likelihood estimation. By summating the probabilities of all the tokens in the tweet regarding the three different language models, we obtain three n-gram-based writing-quality features. *QualityDepend* measures the writing quality based on dependency relation. The dependency feature is generated following Han et al. (2011). Instead of using the technique for normalizing tweet text, we apply it for assessing the grammaticality of tweets[8].

5.2 Cross-media Correlation Features

We observe that Twitter users like to quote or rewrite the important pieces of new content in the posts. If a news sentence is referred or paraphrased by many tweets, it is assumed to be indicated as more important. On the other hand, a tweet, besides its local importance indicator, may be more important if it is similar to the theme of the news content. Therefore, cross-media correlation features are designed to incorporate the auxiliary information source for helping instance ranking. In task 1, news articles are local content and the corresponding tweets are considered auxiliary, and in task 2 their roles are reversed.

5.2.1 Instance-level similarities

Instance-level similarities indicate if there are auxiliary instances similar to the current local instance and to what extent they are similar. These features reveal if the current instance has strong correlation

[5] `http://nlp.stanford.edu/software/CRF-NER.shtml`

[6] `http://mallet.cs.umass.edu/index.php`

[7] The words not found in a common English dictionary, GNU aspell dictionary v0.60.6, are treated as OOV

[8] Both dependency bank and language model here are based on New York Times corpus (`http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2008T19`)

across the media boundary. We use four general metrics including cosine, ROUGE-1 F-value, ROUGE-1 precision score and ROUGE-1 recall score to measure the surface similarity between news sentence and tweet. And the other three features, namely *LeadSenSimi*, *TitleSimi* and *MaxSenPos* are only used in task 2 for ranking tweets when news sentences are considered as auxiliary. This is because leading sentences and title of news are considered as the most informative content. The more similar a tweet to them, the more important it can be. Also, position information is often used for document summarization. We borrow the position of the most similar sentence as bridge to measure the importance of a given tweet.

5.2.2 Semantic-space-level similarities

Semantic-space-level similarities reflect the importance of the current local instance based on the distribution of its semantic units in the auxiliary resource. We propose two features to represent the distribution of the semantic units that are based on unigram frequency and unigram TF-IDF, and named as *SimiUnigram* and *SimiUniTFIDF*, respectively. We first obtain a unigram distribution on the auxiliary space, and compute the similarity of a local instance by summing over the probabilities of all its unigrams in the distribution. Additionally, we also identify some most frequent named entities and unigrams in the auxiliary information source, and then compute the presence and the count of them in the current local instance as additional features, which are named as *SimiTopEntity* and *SimiTopUnigram*.

6 Experiments and Results

6.1 Setup

Task 1 extracts highlights from *news articles*. For comparison, we use the following approaches: (1) *Lead sentence* chooses the first x sentences from the given news article, which is a strong baseline that no DUC system could beat with large margin (Nenkova, 2005); (2) *Phrase ILP* (Woodsend and Lapata, 2010) generates highlights from news with the joint model combining sentence compression and selection, which treats phrases and clauses as extract unit; (3) *Sentence ILP* (Woodsend and Lapata, 2010) is a variant of *Phrase ILP* that treats sentence as extract unit; (4) *LexRank (news)* summarizes the given news using the typical multi-document summarization algorithm LexRank (Erkan and Radev, 2004); (5) *Ours (LSF)* is our ranking method based on the local sentence features which are equivalent to the features used by Svore et al. (2007); (6) *Ours (LSF+CCF)* is our method combining LSF and CCF.

Task 2 extracts highlights from *tweets* where we use the following approaches: (1) *LexRank (tweets)* uses LexRank (Erkan and Radev, 2004) with tweets as the mere input; (2) *Ours (LTF)* is our ranking method based on local tweet features; (3) *Ours (LTF+CCF)* is our method combining LTF and CCF.

Unlike single news document where redundant sentences are rare, the redundancy of tweets is serious. Many summarization algorithms are sensitive to redundancy in the input. It is thus problematic for tweets as the source of extraction. Hence we apply Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998) for reducing tweets redundancy in task 2. The parameter in MMR used to gauge the threshold of redundancy is tuned based on 20% randomly selected training data. Overall, we conduct 5-fold cross-validation for evaluation. The highlights of each news article are used as ground truth. In the output, we fix the number of highlights extracted x as 4. We report ROUGE-1 and ROUGE-2 scores with ROUGE-1 as the major evaluation metric.

6.2 Results

The overall performance can be seen in Table 4, from which we have the following findings:

– Indeed, *Lead sentence* is a very strong baseline that performs much better than most of other methods. It is only a little worse than *LexRank (news)* and much worse than *Ours (LSF+CCF)*.

– *LexRank (news)* performs the second best in task 1. However, the performance of *LexRank (tweets)* is the worst in task 2. This is because LexRank is proposed for summarizing regular documents and its performance is affected seriously by the short, noisy texts like tweets.

– *Sentence ILP* and *Phrase ILP* perform similarly and do not show clear advantage over other baselines. This is different from what Woodsend and Lapata (2010) has obtained. This implies that their model is sensitive to the size of training data where the ILP model may be undertrained here with the

Approach	ROUGE-1			ROUGE-2		
	F	P	R	F	P	R
Lead sentence	0.263	0.211	0.374	0.101	0.080	0.147
LexRank (news)	*0.264*	0.226	0.332	0.088	*0.074*	0.112
Sentence ILP	0.238	0.209	0.293	0.068	0.058	0.088
Phrase ILP	0.236	*0.215*	0.281	0.069	0.061	0.086
Ours (LSF)	0.256	0.214	0.345	0.093	0.076	0.129
Ours (LSF+CCF)	**0.292**	**0.239**	**0.398**	**0.110**	**0.089**	**0.155**
LexRank (tweets)	0.212	0.204	0.226	0.064	0.061	0.068
Ours (LTF)	0.264	0.280	0.274	0.095	0.106	0.098
Ours (LTF+CCF)	**0.295**	**0.320**	**0.295**	**0.105**	**0.118**	**0.105**

Table 4: Overall performance (**Bold**: best performance of the task; Underlined: significance ($p < 0.01$) compared to our best model; *Italic*: significance ($p < 0.05$) compared to our best model)

amount of training data available. In addition, we find there are lots of infeasible solutions for the ILP model, indicating that the hard constraints are not relaxed enough for the relatively small data set.

– *Ours (LSF+CCF)* and *Ours (LTF+CCF)* achieve the best performance on task1 and task2, respectively, and they significantly outperform all other methods in terms of ROUGE-1 F-score based on the result of paired two-tailed t-test. By incorporating CCF, we improve the performance of local features significantly. This justifies that cross-media correlations are indeed useful for improving the quality of exaction from both directions.

– Comparing *Ours (LSF+CCF)* and *Ours (LTF+CCF)*, although their ROUGE-1 F-scores are comparable, the former is better on ROUGE-1 recall and the ROUGE-1 precision of the latter is much higher. This is because news sentences are usually longer than tweets. So the highlights extracted from news article cover more highlight tokens than those from tweets. The length of generated summary and ground truth can be seen in Table 5, where tweet extracts are much closer to the ground-truth highlights. And tweets appear to be a more suitable source for highlights extraction because of the human compression effect on the tweets.

	Tokens # per sentence	Tokens # per summary
Ground-truth highlights	13.2±3.2	49.6±10.0
Ours (LSF+CCF)	24.3±11.8	91.3±18.4
Ours (LTF+CCF)	16.1±5.4	55.3±16.1

Table 5: Comparison of the length of extracted highlights and that of ground truth

6.3 Analysis

Table 6 shows an example for analyzing our extracted highlights compared to the ground-truth. In example 1 (left column), with the help of tweets, *Ours (LSF+CCF)* can output good highlight sentences N2 and N3 which cannot be extracted by *Ours (LSF)*. On the side of tweets, T2 is newly extracted by *Ours (LTF+CCF)* after considering CCF. Furthermore, highlights extracted from tweets also bring extra good highlight T3 which is similar to H1. We find that H1 is rewritten from an original sentence which is three times longer, so it is difficult for extractive method to locate the original sentence in the article. Even if the sentence could be identified, the information was verbose still. Interestingly, some Twitter user produces a tweet like T3 by paraphrasing and shortening which is captured by the algorithm.

Although cross-media correlations are helpful, two out of four ground-truth highlight sentences are covered by the extracted good highlights in example 1. Also, the good extracts from different sources may not cover the same set of ground-truth. Therefore, maybe we can try to combine the extracts from both sides for further improvement.

1: Positive example	2: Negative example
H1. Luxor province bans all hot air balloon flights until further notice H2. The Tuesday accident was the world's deadliest hot air balloon accident in at least 20 years H3. Officials: Passengers in the balloon included 19 foreign tourists H4. No foul play is suspected, official says	HH1. Snowden grew up in Elizabeth City, N.C., but family moved to Ellicott City, Md. HH2. In 2003, he enlisted in the Army, but broke both his legs during Special Forces training HH3. His first NSA job was as a security guard at an agency facility at the University of Maryland
N1. Cairo An official investigation into the cause of a balloon accident that killed 19 people in Egypt could take two weeks, ... N2. [+] **The Tuesday accident was the world's deadliest hot air balloon accident in at least 20 years.** N3. [+] **Tuesday's crash prompted the governor to ban all hot air balloon flights until further notice.** N4. How safe is hot air ballooning?	NN1. A 29-year-old former CIA employee who admitted responsibility Sunday for one of the most extraordinary ... NN2. He told the newspaper he is willing to stand behind his actions in public because "I know I have done nothing wrong." NN3. He told the newspaper that the NSA "routinely lies" to Congress about the scope of its surveillance in the United States. NN4. [+] I can't in good conscience allow the U.S. government to destroy privacy, internet freedom and basic... NN. [-] **His first NSA job was as a security guard at an agency facility at the University of Maryland in College Park, ...**
T1. CNN: official investigation into yesterday air balloon accident in Luxor could take 2 weeks T2. [+] **Governor bans all hot air balloon flights until further notice.** T3. **Foul play not suspected in fatal balloon accident** T4. Official: Egypt balloon explosion probe can take 2 weeks	TT1. I can't in good conscience allow the U.S. government to destroy privacy, Snowden told the Guardian. TT2. whistleblower Edward Snowden: I do not expect to see home again, though that is what I want. TT3. More on ex CIA Snowden: I have done nothing wrong TT4. Ex-CIA employee: Obama advanced surveillance policies, not reformed them.

Table 6: Examples of extracted highlights (H&HH items are the ground-truth highlights, N&NN items are the highlights extracted from news by *Ours (LSF+CCF)*, and T&TT items are the highlights extracted from tweets by *Ours (LTF+CCF)*; Bold: Good highlight; [+]: Newly extracted highlights using correlation features; [-]: Lost highlights after adding correlation features)

Example 2 (right column) shows tweets may not be always useful. *Ours (LSF+CCF)* adds a bad highlight NN4 but removes a good one NN. We find that NN4 is very similar to TT1. So the introduction of NN4 is believed as the result of influence from TT1. NN is squeezed out of the summary since we find it lack of tweets in our set similar to NN. Currently, we only use explicit links for tweets-document couplings. It might be helpful if we could expand the set to cover more informative tweets.

6.4 Contribution of Features

We further investigate the contribution of different features in our feature set (see Table 5) to the learned ranking models. We choose the best models from the two tasks, i.e., *Ours (LSF+CCF)* and *Ours (LTF+CC)*, and find out the top-10 weighted features for each model. To get the feature weights, for each feature we aggregate the weight values of its corresponding weak ranker selected during the iteration in RankBoost training, that is, for a weak ranker repeatedly selected in different rounds, its weights obtained from those rounds are added up to obtain as the feature weight. Table 7 lists the top-10 features and their corresponding weight values.

Cross-media correlation features, which are underlined, appear overwhelmingly important to the sentences extraction task with the model *Ours (LSF+CCF)*, where they take eight places in the top-10 feature list. This confirms the indicative effect of tweets. In tweets extraction task, the model *Ours (LTF+CCF)* does not seem to be so dependent on the cross-media correlation features, but still there are five of them appearing important in the list. In particular, the similarities between tweets and the leading news sentences such as *SimiTopUnigram* and *LeadSenSimi* are shown very helpful. This is because the leading part of the article can be more indicative of important tweets. Besides, the writing-quality measures of tweets are also very useful as it is shown that all the three quality-related features are among the top ten.

7 Conclusion and Future work

In this paper, we explore to utilize microblogs for automatic highlights extraction from two perspectives using learning-based ranking models. Firstly, we extract important sentences from news article by using a set of relevant tweets that provide indicative support for the informativeness of candidate sentences; Secondly, we extract important tweets from the relevant tweets set associated with the given article by taking the advantage of the fact that tweets are comparably concise as highlights. The results show that our methods significantly outperform state-of-the-art baseline approaches for single-document sum-

Task 1: Ours (LSF+CCF)		Task 2: Ours (LTF+CCF)	
Feature	Weight	Feature	Weight
ImportUnigram	4.7912	SimiTopUnigram (count)	1.9300
MaxROUGE1R	2.1049	LeadSenSimi (third)	1.8367
MaxROUGE1F	0.6511	QualityLM (Bigram)	1.4513
SimiTopUnigram (count)	0.6260	MaxROUGE1R	1.1925
SimiUnigram	0.5424	QualityLM (Unigram)	0.9441
MaxROUGE1P	0.1922	LeadSenSimi (second)	0.9224
SimiTFIDF	0.1534	QualityDepend	0.8306
SimiTopEntity (count)	0.0311	TopicNE (person)	0.7937
SimiTopEntity (presence)	0.0051	ImportTFIDF	0.7423
TitleSimi	0.0050	LeadSenSimi (fourth)	0.6072

Table 7: Top 10 features and their weights resulting from the best ranking models in the two tasks (underline: Cross-media correlation features)

marization. Our feature study further discovers that the cross-media correlations are overwhelmingly important to sentence extraction, and for tweets extraction the quality-related features are comparably important as cross-media correlation measures. Also, tweets extraction appears more suitable for producing highlights owing to the human compression effect of tweets.

For the future work, we plan to enlarge the relevant tweets collection by including relevant tweets not linked by URLs; we can combine the extracts from both sides for further improvement; we can also strengthen our model by capturing some deeper or latent linguistic and semantic correlations with deep learning formalism.

Acknowledgments

This work is supported by the QCRI-MIT collaboration program. We appreciate the helpful discussions with Regina Barzilay, Karthik Narasimhan and Lanjun Zhou at early stage of the project. Zhongyu Wei is partially financed by the General Research Fund of Hong Kong (417112) and the Shenzhen Fundamental Research Program (JCYJ20130401172046450) of China. We thank anonymous reviewers for their insightful comments.

References

Regina Barzilay, Michael Elhadad, et al. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, number 1, pages 10–17.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336. ACM.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 34:637–674.

Yajuan Duan, Zhimin Chen, Furu Wei, Ming Zhou, and Heung-Yeung Shum. 2012. Twitter topic summarization by ranking tweets using social influence and content quality. In *Proceedings of COLING*, pages 763–780.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969.

Wei Gao, Peng Li, and Kareem Darwish. 2012. Joint topic modeling for event summarization across news and social media streams. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 1173–1182. ACM.

Weiwei Guo, Hao Li, Heng Ji, and Mona Diab. 2013. Linking tweets to news: A framework to enrich short text data in social media. In *Proceedings of the ACL*, pages 239–249.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a# twitter. In *Proceedings of ACL*, pages 368–378.

Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. 2013. Single-document summarization as a tree knapsack problem. In *Proceedings of EMNLP*, pages 1515–1520.

Meishan Hu, Aixin Sun, and Ee-Peng Lim. 2008. Comments-oriented document summarization: understanding documents with readers' feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 291–298. ACM.

David Inouye and Jugal K Kalita. 2011. Comparing twitter summarization algorithms for multiple post summaries. In *Proceedings of 2011 IEEE Third International Conference on Social Computing (SocialCom)*, pages 298–306. IEEE.

Joel Judd and Jugal Kalita. 2013. Better twitter summaries? In *Proceedings of NAACL-HLT*, pages 445–449.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

Alok Kothari, Walid Magdy, Ahmed Mourad Kareem Darwish, and Ahmed Taei. 2013. Detecting comments on news articles in microblogs. In *Proceedings of ICWSM*, pages 293–302.

Chen Li, Xian Qian, and Yang Liu. 2013. Using supervised bigram-based ilp for extractive summarization. In *Proceedings of ACL*, pages 1004–1013.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on multi-source multilingual information extraction and summarization*, pages 17–24. Association for Computational Linguistics.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165.

Daniel Marcu. 1997. From discourse structures to text summaries. In *Proceedings of ACL*, pages 82–88.

Ani Nenkova. 2005. Automatic text summarization of newswire: lessons learned from the document understanding conference. In *Proceedings of the 20th International Conference on Artificial Intelligence*, pages 1436–1441. AAAI Press.

Jeffrey Nichols, Jalal Mahmud, and Clemens Drews. 2012. Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*, pages 189–198. ACM.

Owen Phelan, Kevin McCarthy, Mike Bennett, and Barry Smyth. 2011. Terms of a feather: Content-based news recommendation and discovery using twitter. In *Advances in Information Retrieval*, pages 448–459. Springer.

Beaux Sharifi, M-A Hutton, and Jugal K Kalita. 2010. Experiments in microblog summarization. In *Proceedings of 2010 IEEE Second International Conference on Social Computing (SocialCom)*, pages 49–56. IEEE.

Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. 2007. Document summarization using conditional random fields. In *Proceeding of IJCAI*, pages 2862–2867.

Tadej Štajner, Bart Thomee, Ana-Maria Popescu, Marco Pennacchiotti, and Alejandro Jaimes. 2013. Automatic selection of social media responses to news. In *Proceedings of the 19th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining*, pages 50–58. ACM.

Ilija Subašić and Bettina Berendt. 2011. Peddling or creating? investigating the role of twitter in news reporting. In *Advances in Information Retrieval*, pages 207–213. Springer.

Jian-Tao Sun, Dou Shen, Hua-Jun Zeng, Qiang Yang, Yuchang Lu, and Zheng Chen. 2005. Web-page summarization using clickthrough data. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 194–201. ACM.

Krysta Marie Svore, Lucy Vanderwende, and Christopher JC Burges. 2007. Enhancing single-document summarization by combining ranknet and third-party sources. In *Proceedings of EMNLP-CoNLL*, pages 448–457.

Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 985–992. Association for Computational Linguistics.

Kristian Woodsend and Mirella Lapata. 2010. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 565–574. Association for Computational Linguistics.

Zi Yang, Keke Cai, Jie Tang, Li Zhang, Zhong Su, and Juanzi Li. 2011. Social context summarization. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 255–264. ACM.

Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval*, pages 338–349. Springer.

Arkaitz Zubiaga, Damiano Spina, Enrique Amigó, and Julio Gonzalo. 2012. Towards real-time summarization of scheduled events from twitter streams. In *Proceedings of the 23rd ACM Conference on Hypertext and Social Media*, pages 319–320. ACM.

A Step Towards Usable Privacy Policy: Automatic Alignment of Privacy Statements

Fei Liu Rohan Ramanath Norman Sadeh Noah A. Smith
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{feiliu, rrohan, sadeh, nasmith}@cs.cmu.edu

Abstract

With the rapid development of web-based services, concerns about user privacy have heightened. The privacy policies of online websites, which serve as a legal agreement between service providers and users, are not easy for people to understand and therefore offer an opportunity for natural language processing. In this paper, we consider a corpus of these policies, and tackle the problem of aligning or grouping segments of policies based on the privacy issues they address. A dataset of pairwise judgments from humans is used to evaluate two methods, one based on clustering and another based on a hidden Markov model. Our analysis suggests a five-point gap between system and median-human levels of agreement with a consensus annotation, of which half can be closed with bag of words representations and half requires more sophistication.

1 Introduction

Privacy policies are legal documents, authored by privacy lawyers to protect the interests of companies offering services through the web. According to a study conducted by McDonald and Cranor (2008), if every internet user in the U.S. read the privacy notice of each new website she visited, it would take the nation 54 billion hours annually to read privacy policies. It is not surprising that they often go unread (Federal Trade Commission, 2012).

Users, nonetheless, might do well to understand the implications of agreeing to a privacy policy, and might make different choices if they did. Researchers in the fields of internet privacy and security have made various attempts to standardize the format of privacy notices, so that they are easier to understand and to allow the general public to have better control of their personal information. An early effort is the Platform for Privacy Preferences Project (P3P), which defines a machine-readable language that enables the websites to explicitly declare their intended use of personal information (Cranor, 2002). Many other studies primarily focus on the qualitative perspective of policies and use tens of carefully selected privacy notices. For example, Kelley et al. (2010) proposed a "nutrition label" approach that formalizes the privacy policy into a standardized table format. Breaux et al. (2014) map privacy requirements encoded in text to a formal logic, in order to detect conflicts in requirements and trace data flows (e.g., what data might be collected, to whom the data will be transferred and for what purposes).

The need for automatically or semi-automatically generating simple, easy-to-digest privacy summaries is further exacerbated by the emergence of the mobile Web and the Internet of Things, with early efforts in this area including the use of static analysis to identify sensitive data flows in mobile apps (Lin et al., 2012) and the development of mobile app privacy profiles (Liu et al., 2014).

Increased automation for such efforts motivates our interest in privacy policies as a text genre for NLP, with the general goal of supporting both user-oriented tools that interpret policies and studies of the contents of policies by legal scholars.

In this paper, we start with a corpus of 1,010 policies collected from widely-used websites (Ramanath et al., 2014),[1] and seek to automatically align segments of policies. We believe this is a worthwhile first

[1] http://www.usableprivacy.org/data

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 884–894, Dublin, Ireland, August 23-29 2014.

Amazon.com Privacy Notice	Walmart Privacy Policy
... **What About Cookies?** Cookies are unique identifiers that we transfer to your device to enable our systems to recognize your device and to provide features such as 1-Click purchasing, Recommended for You, personalized advertisements on other Web sites... ...Because cookies allow you to take advantage of some of Amazon.com's essential features, we recommend that you leave them turned on. For instance, if you block or otherwise reject our cookies, you will not be able to add items to your Shopping Cart, proceed to Checkout, or use any Amazon.com products and services that require you to Sign in...	... **Information We Collect** ...We use "cookies" to recognize you as you use or return to our sites. This is done so that we can provide a continuous and more personalized shopping experience for you. A cookie is a small text file that a website or email may save to your browser and store on your hard drive... **Your Choices** ...You may exercise choices related to our online operations and advertising. For instance, you can choose to browse our websites without accepting cookies. Please know that cookies allow us to recognize you from page to page, and they support your transactions with us. Without cookies enabled, you will still be able to browse our websites, but will not be able to complete a purchase or take advantage of certain website features...

Table 1: Example privacy statements from Amazon.com (left) and Walmart.com (right). The statements are concerned with the websites' cookie policy. The top-most level section subtitles are shown in bold.

step toward interpretation of the documents of direct interest here, and also that automatic alignment of a large set of similarly-constructed documents might find application elsewhere.

Consider the example in Table 1, where we show privacy statements from Amazon.com[2] and Walmart.com.[3] These statements are concerned with the usage of cookies—small data files transferred by a website to the user's computer hard drive—often used for tracking a user's browsing behavior. Cookies are one issue among many that are addressed by privacy policies; by aligning segments by issue, across policies, we can begin to understand the range of policy approaches for each issue.

We contribute pairwise annotations of segment pairs drawn from different policies, for use in evaluating the quality of alignments, an analysis of the inter-annotator reliability, and an experimental assessment of three alignment methods, one based on clustering and two based on a hidden Markov model. This paper's results refine the findings of Ramanath et al. (2014). Our key finding is that these unsupervised methods reach far better agreement with the consensus of crowdworkers than originally estimated, and that the gap between these methods and the "median" crowdworker is about half due to the greedy nature of such methods and about half due to the bag of words representation.

2 Privacy Dataset and Annotations

For completeness, we review the corpus of privacy policies presented by Ramanath et al. (2014), and then present the new annotations created for evaluation of alignment.

2.1 Corpus

We collected 1,010 unique privacy policy documents from the top websites ranked by Alexa.com.[4] These policies were collected during a period of six weeks during December 2013 and January 2014. They are a snapshot of privacy policies of mainstream websites covering fifteen of Alexa.com's seventeen categories (Table 2).[5]

Finding a website's policy is not trivial. Though many well-regulated commercial websites provide a "privacy" link on their homepages, not all do. We found university websites to be exceptionally unlikely to provide such a link. Even once the policy's URL is identified, extracting the text presents the usual challenges associated with scraping documents from the web. Since every site is different in its placement of the document (e.g., buried deep within the website, distributed across several pages, or mingled together with Terms of Service) and format (e.g., HTML, PDF, etc.), and since we wish to preserve as much document structure as possible (e.g., section labels), full automation was not a viable solution.

[2] https://www.amazon.com/gp/help/customer/display.html?nodeId=468496

[3] http://corporate.walmart.com/privacy-security/walmart-privacy-policy

[4] http://www.alexa.com

[5] The "Adult" category was excluded; the "World" category was excluded since it contains mainly popular websites in different languages, and we opted to focus on policies in English in this first stage of research, though multilingual policy analysis presents interesting challenges for future work.

Category	Sections		Paragraphs		Category	Sections		Paragraphs	
	Count	Length	Count	Length		Count	Length	Count	Length
Arts	11.1	254.8	39.2	72.1	Recreation	11.9	218.8	38.5	67.4
Business	10.0	244.2	37.6	65.1	Reference	9.7	179.4	26.2	66.3
Computers	10.5	213.4	34.4	65.4	Regional	10.2	207.7	36.0	59.1
Games	10.0	244.1	34.9	70.1	Science	8.7	155.0	22.1	61.0
Health	9.9	228.2	32.4	69.4	Shopping	11.9	213.9	39.3	64.8
Home	11.6	201.5	32.4	72.0	Society	9.8	230.8	32.6	69.3
Kids and Teens	9.6	231.5	32.3	68.6	Sports	10.1	217.1	29.1	75.6
News	10.3	248.4	35.5	72.4	**Average**	**10.4**	**221.9**	**34.1**	**68.0**

Table 2: Fifteen website categories, average number of sections and paragraphs per document in that category, and average length in word tokens.

We therefore crowdsourced the privacy policy document collection using Amazon Mechanical Turk. For each website, we created a HIT in which a worker was asked to copy and paste the following privacy policy-related information into text boxes: (i) privacy policy URL; (ii) last updated date (or effective date) of the current privacy policy; (iii) privacy policy full text; and (iv) the section subtitles in the top-most layer of the privacy policy. To identify the privacy policy URL, workers were encouraged to go to the website and search for the privacy link. Alternatively, they could form a search query using the website name and "privacy policy" (e.g., "Amazon.com privacy policy") and search in the returned results for the most appropriate privacy policy URL. Each HIT was completed by three workers, paid \$0.05, for a total cost of \$380 (including Amazon's surcharge). After excluding dupliates, the dataset contains 1,010 unique documents.[6]

Given the privacy policy full text and the section subtitles, we partition the full privacy document into different sections, delimited by the section subtitles. To generate paragraphs, we break the sections by lines, and consider each line as a paragraph. We require a paragraph to end with a period, if not, it will be concatenated with the next paragraph. Using this partition scheme, sections contain 12 sentences on average; and paragraphs contain 4 sentences on average. More statistics are presented in Table 2.

2.2 Pairwise Annotations

Ramanath et al. (2014) described an evaluation method in which pairs of privacy policy sections were annotated by crowdworkers.[7] A sample of section pairs from different policies was drawn, stratified by cosine similarity of unigram tfidf vectors. In a single task, a crowdworker was asked whether two sections broadly discussed the same topic. The question was presented alongside three answer options, essentially a strong yes, a yes, and a no. In that initial exploration, each item was annotated at least three times, and up to fifteen, until an absolute majority was reached.

The annotations conducted for this study were done somewhat differently. Our motivations were to enable a more careful exploration of inter-annotator agreement, which was complicated in the earlier work by the variable number of annotations per pair, from three to fifteen. We also sought to explore a more fine-grained problem at the paragraph level.

We sampled 1,000 document pairs from each of the 15 categories, then generated pairs (separately of sections and of paragraphs) by choosing one at random from each document. In total, 1,278,204 section pairs and 7,968,487 paragraph pairs were produced. These pairs were stratified by cosine similarity intervals: [0, 0.25], (0.25, 0.5], (0.5, 0.75], (0.75, 1], as in Ramanath et al. (2014). We sampled 250 pairs from each interval, resulting in 1,000 pairs each of sections and paragraphs.

These pairs were annotated on Amazon Mechanical Turk. The crowdworkers were instructed to carefully read the privacy statements and answer a "yes/no" question, indicating whether the two texts are discussing the same privacy issue or not. Several key privacy issues are provided as examples, including

[6]Note that different websites may be covered by the same privacy policy provided by the parent company. For example, `espn.go.com`, `abc.go.com`, and `marvel.com` are all covered under the Walt Disney privacy policy.

[7]Another evaluation, based on text selected by humans in a separate, unrelated task, was also explored. Because such an evaluation seems less broadly applicable, we did not pursue it here.

	Sections					Paragraphs				
Cosine similarity:	[0, .25]	(.25, .5]	(.5, .75]	(.75, 1]	All	[0, .25]	(.25, .5]	(.5, .75]	(.75, 1]	All
5 workers agree	36.4	**12.4**	28.0	85.2	40.5	42.4	**12.0**	32.8	77.6	41.2
4 workers agree	42.8	**42.4**	42.0	13.6	35.2	39.6	**36.8**	35.6	17.6	32.4
3 workers agree	20.8	**45.2**	30.0	1.2	24.3	18.0	**51.2**	31.6	4.8	26.4
Consensus-yes	4.4	**45.2**	87.2	99.2	59.0	9.2	**66.0**	88.8	98.0	65.5
Consensus-no	95.6	**54.8**	12.8	0.8	41.0	90.8	**34.0**	11.2	2.0	34.5

Table 3: Inter-annotator agreement of section and paragraph pairs.

collection of personal information, sharing of information with third parties, cookies and other tracking techniques, data security, children policies, and contact of the websites. To encourage the crowdworkers to carefully read the privacy statements, we also asked them to copy and paste 1–3 keywords from each section/paragraph, before answering the question.[8] Each section/paragraph pair was judged by five crowdworkers and was rewarded $0.05. In total, $550 was spent on the annotations.

On average, it took a crowdworker 2.15 minutes to complete a section pair and 1.67 minutes for a paragraph pair. Interestingly, although a section is roughly three times the length of a paragraph (see Table 2), the time spent on annotation is not proportional to the text length.

In Table 3, we present the inter-annotator agreement results for section and paragraph pairs, broken down by cosine-similarity bin and by the majority answer. 75.7% (73.6%) of section (paragraph) pairs were agreed upon by four or more out of five annotators. Unsurprisingly, disagreement is greatest in the $(.25, .5]$ similarity bin. Cosine similarity is a very strong predictor of the consensus answer (Pearson correlation 0.72 for section pairs, 0.67 for paragraphs, on this stratified sample).

Ramanath et al. (2014) considered only sections. A different method was used to obtain consensus annotations; we simply kept adding annotators to a pair until consensus was reached. For a fair comparison with the new data, we calculated pairwise agreement among three annotators per item, randomly selected if there were more than three to choose from. On the old section-level data, this was 60.5%; on the new data, it was 71.3% (using five annotators). Although a controlled experiment in the task setup was not conducted, we take this as a sign that our binary question with keywords led to a higher quality set of annotations than the three-way question in the older data. Our experiments in this paper use only the new data.

2.3 Discussion

We had expected higher agreement at the paragraph level, since paragraphs are shorter, presumably easier to read and compare, and presumably more focused on a smaller number of issues. This was not borne out empirically, though a slightly different analysis presented in §4.2 suggests that, among crowdworkers who completed ten or more tasks, paragraphs were easier to agree on.

Privacy policies are generally written by attorneys with expertise in privacy law, though there are automatic generation solutions available that allow a non-expert to quickly fill in a template to create a policy document.[9] Example 1 in Table 4 shows a case of very high text overlap (five out of five annotators agreed on a "yes" answer for this pair). While this kind of localized alignment is not our aim here, we believe that such "boilerplate" text, to the extent that it occurs in large numbers of policies, will make automatic alignment easier.

A case where annotators seem not to have understood, or not taken care to read carefully, is illustrated by Example 2 in Table 4. Both sections describe "opt-out" options for unsubscribing from mailing lists that send promotional messages, though the first is more generally about "communications" and the second only addresses email. Three out of five crowdworkers labeled this example with"no." Achieving better consensus might require more careful training of annotators about a predefined set of concepts at the right granularity.

[8] We have not used these keywords for any other purpose.

[9] For example: `http://www.rendervisionsconsulting.com/blog/wp-content/uploads/2011/09/Privacy-policy-solutions-list_rvc.pdf`

Example 1	Example 2
Policy excerpt from Urban Outfitters website: To serve you better, we may combine information you give us online, in our stores or through our catalogs. We may also combine that information with publicly available information and information we receive from or cross-reference with our Select Partners and others. We use that combined information to enhance and personalize the shopping experience of you and others with us, to communicate with you about our products and events that may be of interest to you, and for other promotional purposes. **Policy excerpt from Williams-Sonoma website:** To serve you better and improve our performance, we may combine information you give us online, in our stores or through our catalogs. We may also combine that information with publicly available information and information we receive from or cross-reference with select partners and others. By combining this information we are better able to communicate with you about our products, special events and promotional purposes and to personalize your shopping experience.	**Policy excerpt from IKEA website:** What if I prefer not to receive communications from IKEA? If you prefer not to receive product information or promotions from us by U.S. Mail, please click here. To unsubscribe from our email list, please follow the opt-out instructions at the bottom of the email you received, or click here and update your profile by deselecting "Please send me: Inspirational emails and updates." **Policy excerpt from Neiman Marcus website:** Emails. You will receive promotional emails from us only if you have asked to receive them. If you do not want to receive email from Neiman Marcus or its affiliates you can click on the "Manage Your Email Preferences" link at the bottom of any email communication sent by us. Choose "Unsubscribe" at the bottom of the page that opens. Please allow us 3 business days from when the request was received to complete the removal, as some of our promotion s may already have been in process before you submitted your request.

Table 4: Privacy policy excerpts. Example 1 (a pair of paragraphs) illustrates the likely use of boilerplate; identical text is marked in gray. Example 2 shows a pair of sections where our intuitions disagree with the annotations.

3 Problem Formulation

Given a collection of privacy policy documents and assuming each document consists of a sequence of naturally-occurring text segments (e.g., sections or paragraphs), our goal is to automatically group the text segments that address the same privacy issue, without pre-specifying the set of such issues. We believe this exemplifies many scenarios where a collection of documents follow a similar content paradigm, such as legal documents and, in some cases, scientific literature. Our interest in algorithms that characterize each individual document's parts in the context of the corpus is inspired by biological sequence alignment in computational biology (Durbin et al., 1998).

In our experiments, we consider a hidden Markov model (HMM) that captures local transitions between topics. The motivation for the HMM is that privacy policies might tend to order issues similarly, e.g., the discussion on "sharing information to third parties" appears to often follow the discussion of "personal information collection." If each of these corresponds to an HMM state, then the regularity in ordering is captured by the transition distribution, and each state is characterized by its emission distribution over words. In this section, we discuss the HMM and two estimation procedures based on Expectation-Maximization (EM) and variational Bayesian (VB) inference.

3.1 Hidden Markov Model

Assume we have a sequence of observed text segments[10] $O = [O_1, O_2, ..., O_T]$, and each O_t represents a text segment in a privacy document ($t \in \{1, 2, ..., T\}$). We denote $O_t = [O_t^1, O_t^2, ..., O_t^{N_t}]$, where each O_t^j corresponds to a word token in the tth text segment; N_t is the total number of word tokens in the segment; T represents the total number of segments in the observation sequence. Each text segment O_t is associated with a hidden state S_t ($S_t \in \{1, 2, \ldots, K\}$, where K is the total number of states). Given an observation sequence O, our goal is to decode the corresponding hidden state sequence S.

We employ a first-order hidden Markov model where the next state depends only on the previous state. A notable difference from the familiar HMM used in NLP (e.g., as used for part-of-speech tagging) is that we allow multiple observation symbols to be emitted from each hidden state. Each symbol corresponds to a word token in the text segment. Hence the likelihood for a single document can be written as:

$$L(\theta, \phi) = \sum_{S \in \{1,...,K\}^T} p(O, S \mid \theta, \phi) = \sum_{S \in \{1,...,K\}^T} \prod_{t=1}^{T+1} \theta_{S_t|S_{t-1}} \prod_{j=1}^{N_t} \phi_{O_t^j|S_t} \quad (1)$$

[10]We use *segments* to refer abstractly to either sections or paragraphs. In any given instantiation, one or the other is used, never a blend.

E-step:

$$\textit{Forward pass:} \quad \alpha_1(\cdot) = 1; \quad \alpha_t(k) = \sum_{k'=1}^{K} \alpha_{t-1}(k') \cdot \theta_{k|k'} \cdot \prod_{j=1}^{N_t} \phi_{O_t^j|k}, \;\; \forall t \in \{2, \ldots, T\}, \forall k \in \{1, \ldots, K\} \tag{2}$$

$$\textit{Backward pass:} \quad \beta_{T+1}(\cdot) = 1; \quad \beta_t(k) = \sum_{k'=1}^{K} \theta_{k'|k} \cdot \prod_{j=1}^{N_t} \phi_{O_t^j|k'} \cdot \beta_{t+1}(k'), \;\; \forall t \in \{T, \ldots, 1\}, \forall k \in \{1, \ldots, K\} \tag{3}$$

$$\textit{Likelihood:} \quad p(O \mid \theta, \phi) = p(O_1, O_2, ..., O_T \mid \theta, \phi) = \sum_{k=1}^{K} \alpha_t(k) \cdot \beta_t(k) \textit{ (for any t)} \tag{4}$$

$$\textit{Posteriors:} \quad \gamma_t(k) = p(S_t = k \mid O, \theta, \phi) = \frac{\alpha_t(k) \cdot \beta_t(k)}{p(O \mid \theta, \phi)} \tag{5}$$

$$\textit{Pair posteriors:} \quad \xi_t(k, k') = p(S_t = k, S_{t+1} = k' \mid O, \theta, \phi) = \frac{\alpha_t(k) \cdot \theta_{k'|k} \cdot \left(\prod_{j=1}^{N_{t+1}} \phi_{O_{t+1}^j|k'} \right) \cdot \beta_{t+1}(k')}{p(O \mid \theta, \phi)} \tag{6}$$

M-step (in EM):

$$\textit{Transitions:} \quad \theta_{k'|k} = \frac{\sum_{t=1}^{T} \xi_t(k, k')}{\sum_{t=1}^{T} \sum_{k''=1}^{K} \xi_t(k, k'')}; \qquad \textit{Emissions:} \quad \phi_{v|k} = \frac{\sum_{t=1}^{T} \gamma_t(k) \cdot \sum_{j=1}^{N_t} \mathbf{1}\{O_t^j = v\}}{\sum_{t=1}^{T} \gamma_t(k) \cdot N_t} \tag{7}$$

Variational update (in VB):

$$\theta_{k'|k} = \frac{\exp \Psi \left(\sum_{t=1}^{T} \xi_t(k, k') + \lambda \right)}{\exp \Psi \left(\sum_{t=1}^{T} \sum_{k''=1}^{K} \xi_t(k, k'') + \lambda \cdot K \right)}; \qquad \phi_{v|k} = \frac{\exp \Psi \left(\sum_{t=1}^{T} \gamma_t(k) \cdot \sum_{j=1}^{N_t} \mathbf{1}\{O_t^j = v\} + \lambda' \right)}{\exp \Psi \left(\sum_{t=1}^{T} \gamma_t(k) \cdot N_t + \lambda' \cdot V \right)} \tag{8}$$

Table 5: Equations for parameter estimation of the HMM with multiple emissions at each state and a single sequence. K is the number of states, V is the emission vocabulary size, and T is the length of the sequence in sections. $\Psi(\cdot)$ is the digamma function.

$\theta_{k'|k}$ denotes the probability of transitioning to state k' given that the preceding state is k. $\phi_{v|k}$ denotes the probability that a particular symbol emitted during a visit to state k is the word v. As in standard treatments, we assume an extra final state at the end of the sequence that emits a stop symbol.

Ramanath et al. (2014) considered three variants of the HMM, with different constraints on the transitions, such as a "strict forward" variant that orders the states and only allows transition to "later" states than the current one. In the evaluation against direct human judgments, they found a slight benefit from such constraints, but they increased performance variance considerably. Here we only consider an unconstrained HMM.

3.2 EM and VB

We consider two estimation methods, neither novel. Both are greedy hillclimbing methods that locally optimize functions based on likelihood under the HMM.

The first method is EM, adapted for the multiple emission case; the equations for the E-step (forward-backward algorithm and subsequent posterior calculations) and the M-step are shown in Table 5.

We also consider Bayesian inference, which seeks to marginalize out the parameter values, since we are really only interested in the assignment of sections to hidden states. Further, Bayesian inference has been found favorable on small datasets (Gao and Johnson, 2008). We assume symmetric Dirichlet priors on the transition and emission distributions, parameterized respectively by $\lambda = 1$ and $\lambda' = 0.1$. We apply mean-field variational approximate inference as described by Beal (2003), which amounts to an EM-like procedure. The E-step is identical to EM, and the M-step involves a transformation of the expected counts, shown in Table 5. (We also explored Gibbs sampling; performance was less stable but generally similar; for clarity we do not report the results here.)

3.3 Implementation Details

In modeling, the vocabulary excludes 429 stopwords,[11] words whose document frequency is less than ten, and a set of terms specific to website privacy polices: *privacy*, *policy*, *personal*, *information*, *service*, *web*, *site*, *website*, *com*, and *please*. After lemmatizing, the vocabulary contains $V = 2{,}876$ words. We further exclude sections and paragraphs that contain less than 10 words. Many of these are not meaningful statements, e.g., "return to top." This results in 9,935 sections and 27,594 paragraphs in the experiments.

During estimation, we concatenate all segments into a single sequence, delimited by a special boundary symbol. This does not affect the outcome (due to the first-order conditions; it essentially conflates "start" and "stop" states), but gave some efficiency gains in our implementation.

EM or VB iterations continue until one of two stopping criteria is met: either 100 iterations have passed, or the relative change in log-likelihood (or the variational bound in the case of VB) falls below 10^{-4}; this consistently happens within forty iterations.

After estimating parameters, we decode using the Viterbi algorithm.

4 Experiments

Our experiments compare three methods for aligning segments of privacy policies, at both the paragraph and the section level:

- A greedy divising clustering algorithm, as implemented in CLUTO.[12] The algorithm performs a sequence of bisections until the desired number of clusters is reached. In each step, a cluster is selected and partitioned so as to optimize the clustering criterion. CLUTO demonstrated robust performance in several related NLP tasks (Zhong and Ghosh, 2005; Lin and Bilmes, 2011; Chen et al., 2011).

- The Viterbi state assignment from the EM-estimated HMM. We report averaged results over ten runs, with random initialization.

- The Viterbi state assignment after VB inference, using the mean field parameters. We report averaged results over ten runs, with random initialization.

Our evaluation metrics are precision, recall, and F-score on the identification of section or paragraph pairs annotated "yes."

4.1 Results

In Figure 1, we present performance of different algorithms using a range of hidden state values $K \in \{1, 2, \ldots, 20\}$. The top row shows precision, recall and F-scores on section pairs, the bottom row on paragraph pairs.

The algorithms mostly perform similarly. At the section level, we find the clustering algorithm to perform better in terms of F-score than the HMM with larger K; at $K = 10$ the two are very close.[13] CLUTO's best performance, 85%, was achieved by $K = 14$.

At the paragraph level, the HMMs outperform clustering in the $K \in [5, 15)$ range, and this is where the peak F-score is obtained (87%). We do not believe these differences among algorithms are especially important, noting only that the HMM's advantage is that it does not require pairwise similarity calculations between all section pairs.

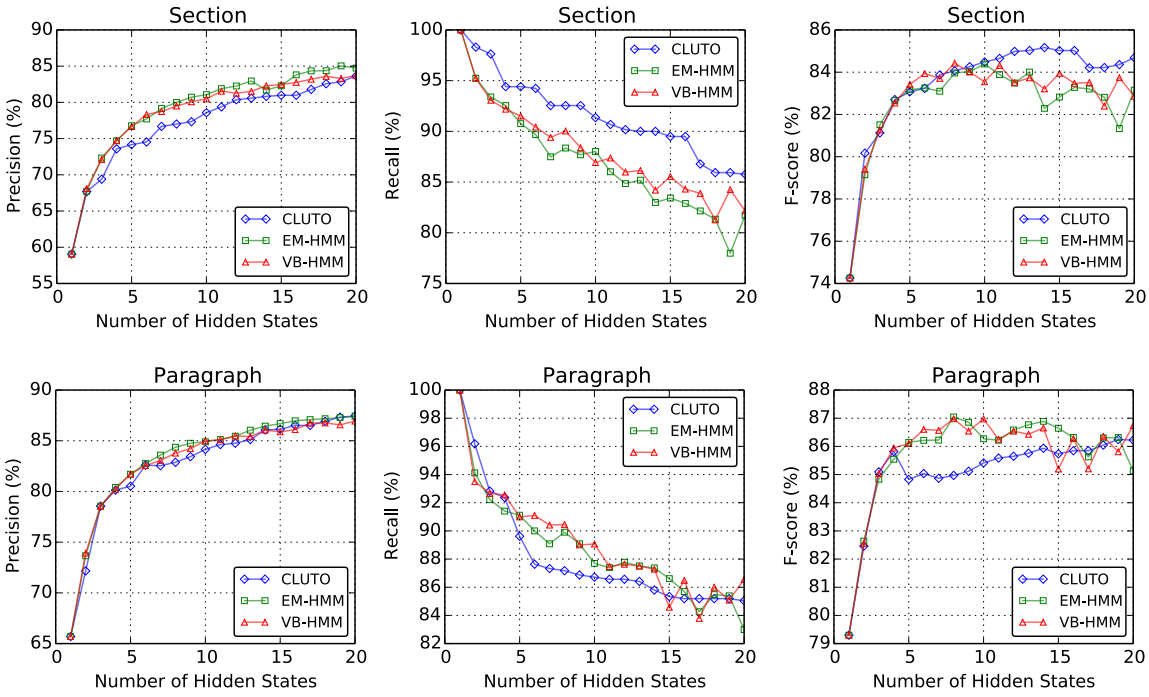


Figure 1: Performance results against pairwise annotations when using different number of hidden states $K \in \{1, \ldots, 20\}$. The top row is at the section level, the bottom row at the paragraph level.

4.2 Upper Bounds

How do these automatic alignment methods compare with the levels of agreement reached among crowdworkers? We consider the agreement rate of each method, at varying values of K, with the majority vote of the annotators. Note that this is distinct from the positive-match–focused precision, recall, and F-score measures presented in §4.1. For each crowdworker who completed ten tasks or more, and therefore for whom we have hope of a reliable estimate, we calculated her agreement rate with the majority. For sections, this set included 65 out of 162 crowdworkers; for paragraphs, 76 out of 197.

In Figure 2 we show the three quartile points for this agreement measure, across the pool of ten-or-more-item crowdworkers, in comparison to the various automatic methods. For sections, our systems perform on par with the 25% of crowdworkers just below the median. For paragraphs, which show a generally higher level of agreement among this subset of crowdworkers, our systems are on par with the lowest 25% of workers. We take all of this to suggest that there is room for improvement in methods overall.

Given the observation in §2 that cosine similarity of two segments' tfidf vectors is a very strong predictor of human agreement on whether they are about the same issue, we also consider a threshold on cosine similarity for deciding whether a pair is about the same issue. This is not a complete solution to the problem of alignment, since pairwise scores only provide groupings if they are coupled with a transitivity constraint. The clustering and HMM methods can be understood as greedy approximations to such an approach. We therefore view cosine similarity thresholding as an upper bound for bag of words representations on the pairwise evaluation task. Figure 2 includes agreement levels for oracle cosine similarity thresholding.[14]

[11] `http://www.lextek.com/manuals/onix/stopwords1.html`

[12] `http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview`

[13] Ramanath et al. (2014) only considered $K = 10$ and found a $K = 10$ HMM to outperform clustering at the section level; the scores reported there, on the earlier dataset, are much lower and not comparable to those reported here. There are numerous differences between the setup here and the earlier one. The most important, we believe, are the improved quality of the dataset and greater care given to preprocessing, most notably the pruning of documents and vocabulary, in the present experiments.

[14] For comparison with the results in §4.1, we found that, for sections, oracle thresholding (at 0.3) achieved F-score of 0.87, and for paragraphs, oracle thresholding (at 0.2) achieved 0.90.

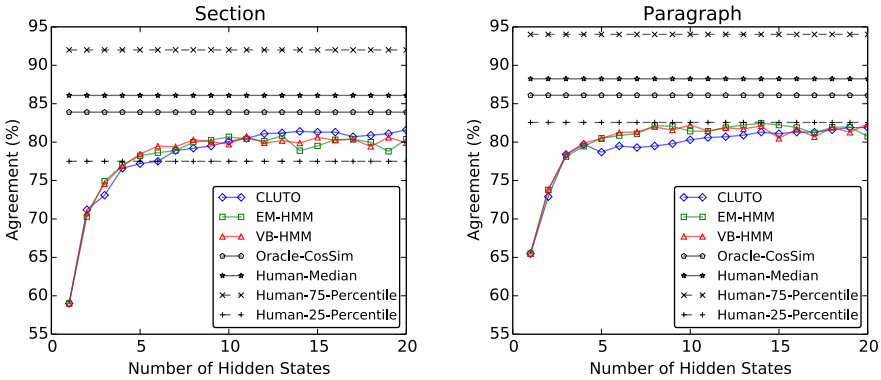


Figure 2: Agreement rates, as compared to crowdworkers and a cosine similarity oracle.

Taken together, this analysis suggests that—in principle—an automated approach based on word-level similarity could close about half of the gap between our methods and median crowdworkers, and further gains would require more sophisticated representations or similarity measures.

5 Related Work

There has been little work on applying NLP to privacy policies. Some have sought to parse privacy policies into machine-readable representations (Brodie et al., 2006) or extract sub-policies from larger documents (Xiao et al., 2012). Machine learning has been applied to assess certain attributes of policies (Costante et al., 2012; Costante et al., 2013), e.g., compliance of privacy policies to legal regulations (Krachina et al., 2007) or simple categorical questions about privacy policies (Ammar et al., 2012; Zimmeck and Bellovin, 2014).

Our alignment-style analysis is motivated by an expectation that many policies will address similar issues,[15] such as collection of a user's contact, location, health, and financial information, sharing with third parties, and deletion of data. This expectation is supported by recommendation by privacy experts (Gellman, 2014) and policymakers (Federal Trade Commission, 2012); in the financial services sector, the Gramm-Leach-Bliley Act *requires* these institutions to address a specific set of issues. Sadeh et al. (2013) describe our larger research initiative to incorporate automation into privacy policy analysis.

Methodologically, the HMM used above is very similar to extensive previous uses of HMMs for POS tagging (Merialdo, 1994), including with Bayesian inference (Goldwater and Griffiths, 2007; Johnson, 2007; Gao and Johnson, 2008). Bayesian topic models (Blei et al., 2003) are a related set of techniques, and future exploration might consider their use in automatically discovering document sections (Eisenstein and Barzilay, 2008), rather than fixing section or paragraph boundaries.

6 Conclusion

This paper presents an exploration of alignment-by-paragraph and -section of website privacy policies. We contribute an improved annotated dataset for pairwise evaluation of automatic methods and an exploration of clustering and HMM-based alignment methods. Our results show that these algorithms achieve agreement on par with the lower half of crowdworkers, with about half of the difference from the median due to the bag of words representation and half due to the inherent greediness of the methods.

Acknowledgments

The authors gratefully acknowledge helpful comments from Lorrie Cranor, Joel Reidenberg, Florian Schaub, and several anonymous reviewers. This research was supported by NSF grant SaTC-1330596.

[15] Personal communication, Joel Reidenberg.

References

Waleed Ammar, Shomir Wilson, Norman Sadeh, and Noah A. Smith. 2012. Automatic categorization of privacy policies: A pilot study. Technical Report CMU-LTI-12-019, Carnegie Mellon University.

Matthew J. Beal. 2003. *Variational Algorithms for Approximate Bayesian Inference*. Ph.D. thesis, Gatsby Computational Neuroscience unit, University College London.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*.

Travis D. Breaux, Hanan Hibshi, and Ashwini Rao. 2014. Eddy, A formal language for specifying and analyzing data flow specifications for conflicting privacy requirements. *Requirements Engineering Journal*.

Carolyn A. Brodie, Clare-Marie Karat, and John Karat. 2006. An empirical study of natural language parsing of privacy policy rules using the SPARCLE policy workbench. In *Proceedings of the Second Symposium on Usable Privacy and Security (SOUPS)*.

Harr Chen, Edward Benson, Tahira Naseem, and Regina Barzilay. 2011. In-domain relation discovery with meta-constraints via posterior regularization. In *Proceedings of ACL-HLT*.

Elisa Costante, Yuanhao Sun, Milan Petković, and Jerry den Hartog. 2012. A machine learning solution to assess privacy policy completeness. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society*.

Elisa Costante, Jerry Hartog, and Milan Petkovi. 2013. What websites know about you. In Roberto Pietro, Javier Herranz, Ernesto Damiani, and Radu State, editors, *Data Privacy Management and Autonomous Spontaneous Security*, volume 7731 of *Lecture Notes in Computer Science*, pages 146–159. Springer Berlin Heidelberg.

Lorrie Faith Cranor. 2002. *Web Privacy with P3P*. O'Reilly & Associates.

Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of ACL*.

Federal Trade Commission. 2012. Protecting consumer privacy in an era of rapid change: Recommendations for businesses and policymakers. Available at `http://www.ftc.gov/reports/protecting-consumer-privacy-era-rapid-change-recommendations-businesses-policymakers`.

Jianfeng Gao and Mark Johnson. 2008. A comparison of Bayesian estimators for unsupervised Hidden Markov model POS taggers. In *Proceedings of EMNLP*.

Robert Gellman. 2014. Fair information practices: a basic history (v. 2.11). Available at `http://www.bobgellman.com/rg-docs/rg-FIPShistory.pdf`.

Sharon Goldwater and Thomas L. Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of ACL*.

Mark Johnson. 2007. Why doesnt EM find good HMM POS-taggers? In *Proceedings of EMNLP–CoNLL*.

Patrick Gage Kelley, Lucian Cesca, Joanna Bresee, and Lorrie Faith Cranor. 2010. Standardizing privacy notices: An online study of the nutrition label approach. In *Proceedings of CHI*.

Olga Krachina, Victor Raskin, and Katrina Triezenberg. 2007. Reconciling privacy policies and regulations: Ontological semantics perspective. In Michael J. Smith and Gavriel Salvendy, editors, *Human Interface and the Management of Information. Interacting in Information Environments*, pages 730–739. Springer.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of ACL*.

Jialiu Lin, Shahriyar Amini, Jason I. Hong, Norman Sadeh, Janne Lindqvist, and Joy Zhang. 2012. Expectation and purpose: Understanding users' mental models of mobile app privacy through crowdsourcing. In *Proceedings of ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*.

Bin Liu, Jialiu Lin, and Norman Sadeh. 2014. Reconciling mobile app privacy and usability on smartphones: Could user privacy profiles help? In *Proceedings of WWW*.

Aleecia M. McDonald and Lorrie Faith Cranor. 2008. The cost of reading privacy policies. *I/S: A Journal of Law and Policy for the Information Society*.

Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–171.

Rohan Ramanath, Fei Liu, Norman Sadeh, and Noah A. Smith. 2014. Unsupervised alignment of privacy policies using hidden Markov models. In *Proceedings of ACL*.

Norman Sadeh, Alessandro Acquisti, Travis Breaux, Lorrie Cranor, Aleecia McDonald, Joel Reidenberg, Noah Smith, Fei Liu, Cameron Russel, Florian Schaub, and Shomir Wilson. 2013. The usable privacy policy project: Combining crowdsourcing, machine learning and natural language processing to semi-automatically answer those privacy questions users care about. Technical Report CMU-ISR-13-119, Carnegie Mellon University.

Xusheng Xiao, Amit Paradkar, Suresh Thummalapenta, and Tao Xie. 2012. Automated extraction of security policies from natural-language software documents. In *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*.

Shi Zhong and Joydeep Ghosh. 2005. Generative model-based document clustering: a comparative study. *Knowledge and Information Systems*, 8(3):374–384.

Sebastian Zimmeck and Steven M. Bellovin. 2014. Privee: An architecture for automatically analyzing web privacy policies. In *Proceedings of the 23rd USENIX Security Symposium*.

An Off-the-shelf Approach to Authorship Attribution

Jamal Abdul Nasir
Dep. of Computer Science
LUMS Lahore
Pakistan
jamaln@lums.edu.pk

Nico Görnitz
Dep. of Computer Science
TU Berlin
Germany
goernitz@tu-berlin.de

Ulf Brefeld
Dep. of Computer Science
TU Darmstadt
Germany
brefeld@cs.tu-darmstadt.de

Abstract

Authorship detection is a challenging task due to many design choices the user has to decide on. The performance highly depends on the right set of features, the amount of data, in-sample vs. out-of-sample settings, and profile- vs. instance-based approaches. So far, the variety of combinations renders off-the-shelf methods for authorship detection inappropriate. We propose a novel and generally deployable method that does not share these limitations. We treat authorship attribution as an anomaly detection problem where author regions are learned in feature space. The choice of the right feature space for a given task is identified automatically by representing the optimal solution as a linear mixture of multiple kernel functions (MKL). Our approach allows to include labelled as well as unlabelled examples to remedy the in-sample and out-of-sample problems. Empirically, we observe our proposed novel technique either to be better or on par with baseline competitors. However, our method relieves the user from critical design choices (e.g., feature set) and can therefore be used as an off-the-shelf method for authorship attribution.

1 Introduction

Automatically attributing a piece of text to its author is one of the oldest problems studied in linguistics (Mendenhall, 1887). Despite being an old problem, authorship attribution is still highly topical and todays applications range from plagiarism detection (Maurer et al., 2006), identifying the origin of anonymous harassments in emails, blogs, and chat rooms (Tan et al., 2013) to copyright and estate issues as well as resolving historical questions of disputed authorship (Mosteller and Wallace, 1964; Fung, 2003).

Intrinsically, the goal of authorship detection is to identify the characteristic traits of an author. The idea is that, these traits distinguish an author from others in terms of writing style, use of words, etc. Thus, prior work often focuses on designing and extracting features from text to capture these traits. There is a great deal of features proposed for authorship detection, including word or character n-grams (Burrows, 1987; Houvardas and Stamatatos, 2006), part-of-speech (Stamatatos et al., 2001), probabilistic context-free grammars (Raghavan et al., 2010), or linguistic features (Koppel et al., 2006). However, indicative features for one author do not necessarily help to characterise another. A major problem in authorship detection is therefore to find the right set of features for a given task at hand (Forman, 2003).

Algorithmically, a variety of different models have been studied in the context of authorship detection, ranging from probabilistic approaches (Seroussi et al., 2011) and similarity-based methods (Koppel et al., 2011) to vector space models (Fung, 2003; Li et al., 2006). The approaches either treat documents as independent (instance-based) or concatenate documents by the same author (profile-based). Intuitively, the latter is helpful if an author has a concise way of expressing herself so that the concatenated document allows to extract a statistic that is sufficient for capturing her style. On the other hand, instance-based approaches are better suited for expressive authors and have advantages in sparse data scenarios.

Another aspect in authorship attribution is the application scenario of the final model. In transductive (in-sample) settings, the unlabelled documents of interest are already included in the training process

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 895–904, Dublin, Ireland, August 23-29 2014.

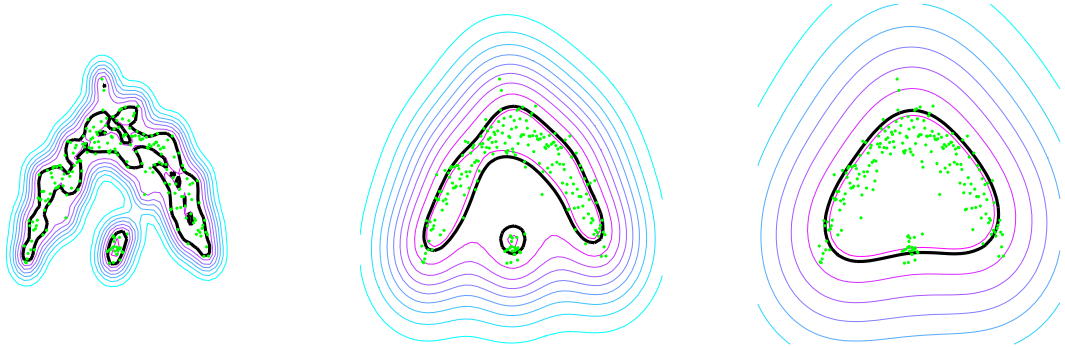


Figure 1: Three solutions of an anomaly detection problem where data is represented by RBF kernels with different band-width parameters. Combining anomaly detection with multiple kernel learning allows to include all three kernels simultaneously in the optimisation and to find the optimal linear mixture of the three (or more) kernels together automatically for a given task.

and the model does not necessarily perform well on new and unseen texts. By contrast, inductive (out-of-sample) scenarios generally allow to learn models that can be applied to any future text but require larger training samples to achieve accurate performances.

In this paper, we propose a general machine learning-based approach to authorship detection. Our approach remedies the above mentioned problems by fusing existing techniques: (i) We cast authorship attribution as an anomaly detection problem where one model is learned for every author. The idea is to identify a concise region in feature space that contains (most of) the documents of the author of interest while other documents are considered outliers. Thus, the model can be viewed as a profile-based approach in feature space while the data is treated on an instance-based level. (ii) We remedy the in-sample / out-of-sample problem by providing a semi-supervised extension of the commonly unsupervised outlier detection framework. By doing so, we may include authorship labels for the already known documents and leave the disputed ones unlabelled. (iii) Finally, we devise our model consequentially as a member of the multiple kernel learning family to automatically include a mathematically well founded feature selection framework that renders the method generally applicable. The optimal solution is given by a (possibly sparse) linear mixture of kernel functions.

Empirically, we observe that our approach consistently outperforms baseline competitors or confirms common knowledge with respect to the authorship of disputed articles. The main advantage of the method however lies in its simplicity. Practitioners do not need to take critical design choices in terms of which features to use and which not. By contrast, all features (kernels) can be used in the optimisation and the method itself finds the optimal combination for the problem at hand.

The remainder of this paper is structured as follows. Section 2 reviews related work. Our main contribution is presented in Section 3. We report on empirical results in Section 4 and Section 5 concludes.

2 Related Work

Authorship attribution using linguistic and stylistic features has a long tradition and can be dated back to the nineteenth century. As a first attempt, Mendenhall (Mendenhall, 1887) uses features based on word lengths to characterise the plays of Shakespeare. Later in the first half of the 20th century, different textual statistics, such as Zipf's distribution (Zipf, 1932) and Yule's k-statistic (Yule, 1944) have been proposed to quantify textual style. A study conducted by (Mosteller and Wallace, 1964) is one of the most influential modern work in authorship attribution. They use a Bayesian approach to analyse frequencies of a small set of function words for *The Federalist Papers*, a series of 85 political essays written by John Jay, Alexander Hamilton, and James Madison. Until the late 1990s, research in *stylometry* has been dominated by feature engineering to quantify writing style (Holmes, 1998) and about 1,000 different measures have been proposed (Rudman, 1997).

Document representation is essential for author attribution tasks. Features aim to capture characteristic traits of authors that persist across topics. Traditional stylometric features include function and high-frequency words, hapax legomena, Yules k-statistic, syllable distributions, sentence length, word length and word frequencies, vocabulary richness functions as well as syntactic features. Many studies combine features of different types using multivariate analyses. Some researchers use punctuation symbols while others experiment with n-grams (Diederich et al., 2003). Grammatical style markers with natural language processing techniques are also used to extract features from the documents.

Also in terms of technical approaches, authorship attribution has been studied with a wide range of different approaches. The deployed techniques can be broadly divided into three categories: machine learning (Diederich et al., 2003), multivariate/cluster analysis (Khmelev, 2000), and natural language processing (Stamatatos et al., 2000). Principal components analysis (PCA) is one of the widely used techinques for authorship studies, for instance, (Holmes and Crofts, 2010) apply PCA to identify the authorship of unknown articles that have been attributed to Stephen Crane. In addition, machine learning-based approaches, including neural networks (Neme et al., 2011) and support vector machines (SVMs) (Diederich et al., 2003), are frequently used to discriminate documents by different authors. An excellent survey on the diversity of approaches for authorship detection is provided by (Stamatatos, 2009).

Density level set estimation, also known as one-class learning (Tax and Duin, 1999; Schölkopf et al., 1999), is the problem of learning a representation of a single class of interest, rejecting data points that deviate from the learned model of normality. Thus, it has been proven very successful in anomaly detection scenarios such as network intrusion detection (Görnitz et al., 2009). Various extensions have been proposed, i.e. to incorporate prior and additional knowledge (Görnitz et al., 2013; Blanchard et al., 2010) in a semi-supervised fashion (Chapelle et al., 2006) and to learn linear combinations of kernels (Kloft et al., 2011; Rakotomamonjy et al., 2008) which is especially usefull whenever the right choice of representation is unknown.

3 Methodology

In this section, we cast semi-supervised anomaly detection as an instance of multiple kernel learning. The rational for this idea is shown in Figure 1. The figure shows three solutions for an anomaly detection task. The data is represented by RBF kernels with different band-width parameters. As shown in the figure, the choice of the band width parameter is crucial and leads to very different solutions. Usually, kernel parameters are thus included in the model selection and their optimisation is often time consuming. Fusing the anomaly detection with multiple kernel learning allows to include all three kernels simultaneously in the optimisation and to find the best linear mixture of the three (or more) kernels together with model parameters for the data at hand.

We briefly review anomaly detection and semi-supervised anomaly detection in Sections 3.1 and 3.2, respectively, and present our main contribution, multiple kernel learning for anomaly detection, in Section 3.3.

3.1 Preliminaries

Anomaly detection is often cast as an unsupervised one-class problem where the goal is to find a hyperplane that separates the majority of the input examples from the origin with maximum margin, so that, by definition, points not exceeding the hyperplane are considered outliers. Analogously, we aim to learn a separating hyperplane for articles by an author of interest, such that documents not exceeding the learned hyperplane are written by other authors.

Given a n articles $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ of possibly different authors, a straight forward optimisation problem that identifies the hyperplane in terms of its normal vector $\mathbf{w}$ and threshold ρ is known as one-class support vector machine (Tax and Duin, 1999; Schölkopf et al., 1999) and given by

$$\begin{aligned} \min_{\mathbf{w},\rho,\boldsymbol{\xi}\geq 0} \quad & \frac{1}{2}\| \mathbf{w} \|_2^2 - \rho + \eta_u \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad \forall_{i=1}^{n} : \quad & \langle \mathbf{w}, \phi(\boldsymbol{x}_i) \rangle \geq \rho - \xi_i. \end{aligned}$$

The hyperplane is realised by the decision function

$$f(\boldsymbol{x}) = \langle \mathbf{w}, \phi(\boldsymbol{x}) \rangle - \rho$$

and new articles are credited to the author if $f(\boldsymbol{x}) > 0$ and are considered work by someone else if $f(\boldsymbol{x}) \leq 0$. The threshold ρ can be interpreted as a measure of expressiveness of an author. E.g., authors who have a very clear and concise style realise smaller thresholds than expressive authors that may adopt to different writing styles.

3.2 Semi-supervised Anomaly Detection

Using only unlabelled data is usually leading to inaccurate models in the presence of only a few data points. We therefore extend the problem setting to include m labeled examples $(\boldsymbol{x}_{n+1}, y_{n+1}), \ldots, (\boldsymbol{x}_{n+m}, y_{n+m})$ in addition to the n unlabelled ones. Labels $y_i \in \{+1, -1\}$ are considered binary, that is in case $y_i = +1$, document $\boldsymbol{x}_i$ belongs to the author of interest. To combine sums and hence, improve readability, we introduce labels $y_i = +1 \; \forall \, i = 1, \ldots, n$ for all unlabelled examples and an indicator function $\mathbb{I}_c \equiv [c > n]$ to mask labeled examples; the function $\mathbb{I}_c$ simply returns 1 if $c > n$ and 0 otherwise.

A semi-supervised generalisation of the hypersphere model of the previous section is the convex semi-supervised anomaly detection (SSAD) (Görnitz et al., 2013) which uses an L_2-regularizer together with the hinge-loss. Let γ be the margin for the labeled examples and κ, η_u, and η_l trade-off parameters, the optimisation problem becomes

$$\begin{aligned} \min_{\mathbf{w}, \rho, \gamma \geq 0, \boldsymbol{\xi} \geq 0} \quad & \frac{1}{2} \| \mathbf{w} \|_2^2 - \rho - \kappa\gamma + \sum_{i=1}^{n+m} (\mathbb{I}_i \eta_l + (1 - \mathbb{I}_i)\eta_u)\xi_i \\ \text{s.t.} \quad & \forall_{i=1}^{n+m} : \; y_i \langle \mathbf{w}, \phi(\boldsymbol{x}_i) \rangle \geq y_i \rho + \mathbb{I}_i \gamma - \xi_i. \end{aligned}$$

The solution $\boldsymbol{w}$ admits a dual representation and can be written as

$$\boldsymbol{w} = \sum_{i=1}^{n+m} \alpha_i y_i \phi(\boldsymbol{x}_i)$$

and hence, the decision function depends only on inner products of the input examples wich paves the way for kernel functions $K_\phi(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle$ (see (Müller et al., 2001) for an introduction to kernels). It holds

$$f(\boldsymbol{x}) = \sum_{i=1}^{n+m} \alpha_i y_i \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}) \rangle - \rho = \sum_{i=1}^{n+m} \alpha_i y_i K_\phi(\boldsymbol{x}_i, \boldsymbol{x}) - \rho.$$

We omit the subscript ϕ in the remainder to not clutter notation unnecessarily.

3.3 Multiple Kernel Learning for Anomaly Detection

Learning linear combinations of multiple kernels is an appealing strategy when the right choice of representations is unknown. We therefore generalise the semi-supervised anomaly detection of the previous section as a member of the multiple kernel learning framework (Lanckriet et al., 2004). Thus, we aim to learn a weighted combination of T kernels with mixing coefficients $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_T)$:

$$\begin{aligned} K_{\text{MKL}}(\boldsymbol{x}, \boldsymbol{x}') &:= \sum_{t=1}^{T} \beta_{\text{t}} \, K_t(\boldsymbol{x}, \boldsymbol{x}') = \sum_{t=1}^{T} \beta_{\text{t}} \langle \phi_t(\boldsymbol{x}), \phi_t(\boldsymbol{x}') \rangle \\ &= \sum_{t=1}^{T} \langle \sqrt{\beta_{\text{t}}} \phi_t(\boldsymbol{x}), \sqrt{\beta_{\text{t}}} \phi_t(\boldsymbol{x}') \rangle. \end{aligned}$$

In general, properties of the mixing coefficients include (i) non-negativity, hence $\beta_{\mathrm{t}} \geq 0$ and (ii) normalisation $\| \boldsymbol{\beta} \|_p = 1$. Recent work (Kloft et al., 2011) suggests to use the more general p-norm instead of a common 1-norm (Lanckriet et al., 2004; Bach et al., 2004; Rakotomamonjy et al., 2008). The latter usually leads to sparse mixing coefficients which is not appealing in every situation whereas p-norm with $1 \leq p \leq \inf$ admits sparsity adjustments for the problem at hand and thus adds flexibility. Incorporating multiple feature representations in our model introduced in Section 3.1 leads to

$$f_{\mathrm{MKL}}(\boldsymbol{x}) = \sum_{t=1}^{T} \langle \hat{\mathbf{w}}_t, \sqrt{\beta_{\mathrm{t}}} \phi_t(\boldsymbol{x}) \rangle - \rho = \sum_{t=1}^{T} \sqrt{\beta_{\mathrm{t}}} \langle \hat{\mathbf{w}}_t, \phi_t(\boldsymbol{x}) \rangle - \rho. \tag{1}$$

Due to technical reasons, i.e. to preserve convexity, we substitute the model parameters $\mathbf{w}_t = \sqrt{\beta_t} \hat{\mathbf{w}}_t$ and arrive at the revised primal MKL-SSAD optimisation problem:

$$\begin{aligned} \min_{\{\mathbf{w}_t\}, \rho, \gamma \geq 0, \boldsymbol{\xi} \geq 0, \boldsymbol{\beta} \geq 0} \quad & \frac{\lambda}{2} \| \boldsymbol{\beta} \|_p^2 + \frac{1}{2} \sum_{t=1}^{T} \frac{1}{\beta_{\mathrm{t}}} \| \mathbf{w}_t \|_2^2 - \rho - \kappa\gamma + \sum_{i=1}^{n+m} (\mathbb{I}_i \eta_l + (1 - \mathbb{I}_i) \eta_u) \xi_i \\ \text{s.t.} \quad & \forall_{i=1}^{n+m} : \; y_i \sum_{t=1}^{T} \langle \mathbf{w}_t, \phi_t(\boldsymbol{x}_i) \rangle \geq y_i \rho + \mathbb{I}_i \gamma - \xi_i. \end{aligned} \tag{2}$$

(Kloft et al., 2011) prove the equivalence of Tikhonov and Ivanov regularisation which allows to move the regulariser on the mixing coefficients in the objective function. We will exploit this relation on various occasions in this section. Deriving the Lagrange dual problem, we arrive at the intermediate saddle point problem

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \min_{\{\mathbf{w}_t\}, \boldsymbol{\beta} \geq 0} \; & \frac{\lambda}{2} \| \boldsymbol{\beta} \|_p^2 + \frac{1}{2} \sum_{t=1}^{T} \frac{1}{\beta_{\mathrm{t}}} \| \mathbf{w}_t \|_2^2 - \sum_{i=1}^{n+m} \alpha_i y_i \sum_{t} \langle \mathbf{w}_t, \phi_t(\boldsymbol{x}_i) \rangle \\ \text{s.t.} \quad & \kappa \leq \sum_{i=1}^{n+m} \mathbb{I}_i \alpha_i \quad \text{and} \quad 0 \leq \alpha_i \leq \mathbb{I}_i \eta_l + (1 - \mathbb{I}_i) \eta_u \quad \forall\, i \end{aligned}$$

We are solving the optimisation problem in a block-coordinate descent fashion by alternating between $\mathbf{w}$ and $\boldsymbol{\beta}$. This enables us to compute the latter analytically assuming fixed variables $\mathbf{w}$ and setting the partial derivative to zero:

$$\lambda \, {\beta_{\mathrm{t}}}^{p-1} \| \boldsymbol{\beta} \|_p^{2-p} - \frac{\| \mathbf{w}_t \|_2^2}{{\beta_{\mathrm{t}}}^2} = 0.$$

Therefore, given $\Upsilon \geq 0$ we get

$$\beta_{\mathrm{t}} = \Upsilon \| \mathbf{w}_t \|_2^{\frac{2}{p+1}}.$$

Furthermore, it holds that at any optimal point $\| \boldsymbol{\beta} \|_p = 1$ and solving for Υ gives $\Upsilon = 1/(\sum_{t=1}^{T} \| \mathbf{w}_t \|_2^{\frac{2p}{p+1}})^{\frac{1}{p}}$. Putting things together, gives the analytical update rule

$$\beta_{\mathrm{t}} = \frac{\| \mathbf{w}_t \|_2^{\frac{2}{p+1}}}{(\sum_{t=1}^{T} \| \mathbf{w}_t \|_2^{\frac{2p}{p+1}})^{\frac{1}{p}}} \tag{3}$$

which, since only norms are involved, ensures non-negativity for the mixing coefficients. Substituting $\mathbf{w}_t$ using the representer theorem $\mathbf{w}_t = \beta_{\mathrm{t}} \sum_{i=1}^{n+m} \alpha_i \boldsymbol{y}_i \phi_t(\boldsymbol{x}_i)$ yields the final optimisation problem for

Algorithm 1 Proposed optimization algorithm for MKL-SSAD (2)

Require: $\boldsymbol{x}, \boldsymbol{y}, \eta_u, \eta_l, \kappa$ & $p-$ norm
  Initialize kernel mixture coefficients such that $\| \boldsymbol{\beta}^{z=0} \|_p = 1$
  while Until Convergence **do**
    Step 1: solve the convex SSAD problem as stated in Eqn. (4)
    $\boldsymbol{\alpha}^{z+1} = \text{argmax}_{\boldsymbol{\alpha}: 0 \leq \alpha_i \leq \mathbb{I}_i \eta_l + (1-\mathbb{I}_i)\eta_u} J(\boldsymbol{\alpha}, \boldsymbol{\beta}^z) \quad \text{s.t.} \quad \kappa \leq \sum_{i=1}^{n+m} \mathbb{I}_i \alpha_i$
    Step 2: optimize the weights according to Eqn. (3)
    $\boldsymbol{\beta}^{z+1} = \text{argmin}_{\boldsymbol{\beta} \geq 0} J(\boldsymbol{\alpha}^{z+1}, \boldsymbol{\beta}) \quad \text{s.t.} \quad \| \boldsymbol{\beta} \|_p^2 \leq 1$
    z=z+1
  end while
  return Trained parameter vector $\boldsymbol{\alpha}^\star$, weights $\boldsymbol{\beta}^*$

MKL-SSAD:

$$\max_{\boldsymbol{\alpha}} \min_{\boldsymbol{\beta}: \| \boldsymbol{\beta} \|_p^2 \leq 1} \overbrace{-\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \sum_{t=1}^{T} \beta_t K_t(\boldsymbol{x}_i, \boldsymbol{x}_j)}^{=: J(\boldsymbol{\alpha}, \boldsymbol{\beta})} \tag{4}$$
$$\text{s.t.} \quad \kappa \leq \sum_{i=1}^{n+m} \mathbb{I}_i \alpha_i \quad \text{and} \quad 0 \leq \alpha_i \leq \mathbb{I}_i \eta_l + (1 - \mathbb{I}_i)\eta_u \quad \forall\, i$$

As a block-coordinate descent method, we can iteratively alternate between the two optimisation blocks and every limit point of Algorithm 1 is a globally optimal point (cmp. also (Kloft et al., 2011)). Algorithm 1 summarises the proposed optimisation procedure.

4 Empirical Results

In this section, we empirically evaluate the benefit of fusing semi-supervised anomaly detection with multiple kernel learning. We experiment on two data sets, the Reuters-50-50 corpus in Section 4.1 and the Federalist Papers in Section 4.2

4.1 Reuters 50-50

We use a subset of the Reuters 50-50 data set[1] to evaluate the performance of the aforementioned approaches. The reduced data contains 1000 articles written by 10 authors, Aaron Pressman, Alan Crosby, Alexander Smith, Benjamin Kang Lim, Bernard Hickey, Brad Dorfman, Darren Schuettler, David Lawder, Edna Fernandes, and Eric Auchard.

We deploy the following four kernels to represent documents: the first kernel is made of 484 function words taken from (Koppel and Schler, 2003), the second contains part-of-speech (POS) tags, the third is assembled by 3-letter suffixes, the last one simply a bag-of-words (BOW) kernel. We split the data into training (90%) and test (10%) sets and conduct a 10-fold cross-validation on the training set for model selection. The best performing models are then evaluated on the test set. In this set of experiments, we use a transductive setting where all training instances are labeled and only holdout and test articles are unlabelled. We compare the performance of our approach with different p-norms to the SSAD which uses one kernel at a time. For our MKL-based approach we use p-norms in the set $p \in \{1, 1.7783, 3.1623, 5.6234, 10\}$.

The results in terms of averaged micro- and macro-F_1 measures are shown in Table 1. MKL consistently outperforms the single-kernel baseline for all p-norms. That is, instead of extensively experimenting with SSAD and different kernel functions and parameter selections, a single run with our MKL already leads to better performances in both metrics. The rightmost column in the table shows the result for SSAD using a sum of the input kernels. Apparently, the performance is worse than using a bag-of-words kernel alone. This result underlines the necessity of effective feature selection techniques for

[1] https://archive.ics.uci.edu/ml/datasets/Reuter_50_50

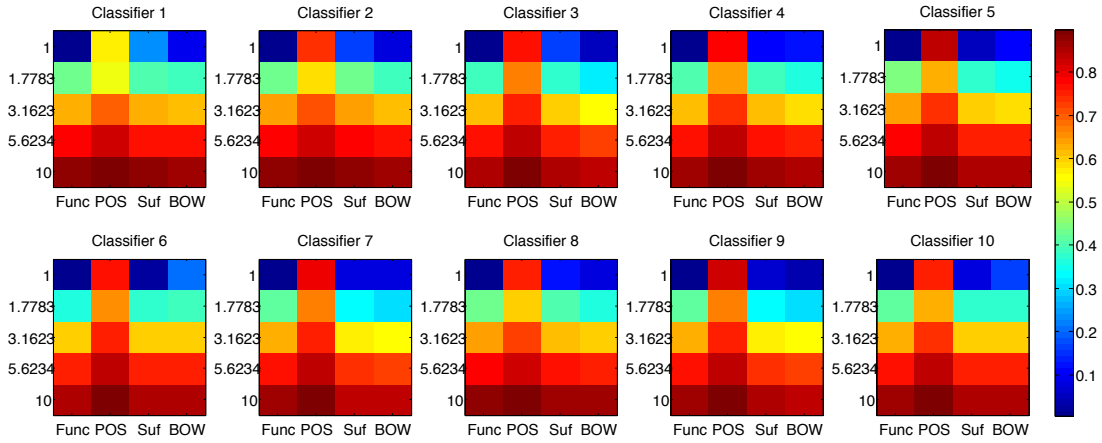


Figure 2: Kernel mixture coefficients for the 10 classes

authorship attribution. Note that our method can actually be viewed as an ensemble method that combines several models as shown in Equation (1). However, compared to traditional ensembles, our method uses a convex combination and hence returns the optimal ensemble given the data.

Table 1: F-scores for the subset of Reuters 50-50

	p-norm MKL					SSAD				
	1	1.7783	3.1623	5.6234	10	func-w	POS	Suffix-3	BOW	$\sum$
F_{micro}	73.46	73.08	73.84	73.89	74.23	63.08	54.62	70.01	72.85	61.76
F_{macro}	79.23	78.86	79.63	79.76	80.07	68.66	58.03	74.01	78.09	70.93

Figure 2 visualises the resulting mixing coefficients for the 10 authors/classifiers. While the models are very similar at first sight, small deviations indicate differences in the style of the authors. Consider for instance the top-left matrix. The contribution of the part-of-speech tag kernel (second column) to the final solution is less than for the other authors. By contrast, the importance of the Suffix-3 kernel has (slightly) more impact than for the remaining authors. This result shows that author-dependent mixtures are found that help to capture characteristic traits of the respective writing styles.

4.2 Revisiting the Federalist Papers

The Federalist Papers are a series of 85 articles and essays written during 1787–1788. They were published anonymously to persuade the citizens of the State of New York to ratify the Constitution. Later, these papers were credited to Alexander Hamilton, John Jay, and James Madison; 73 of the documents are uniquely associated with one of the three authors while the remaining 12, also known as the disputed papers, have been claimed by both, Hamilton and Madison. Three of the 73 articles are considered joint work by Hamilton and Madison. Previous studies often assign all 12 disputed papers to Madison which we assume as ground-truth in the remainder (Mosteller and Wallace, 1964; Fung, 2003).

To confirm or refuse these previous findings, we conduct an experiment using same four kernels as in the previous section, that is, a function words kernel (Koppel and Schler, 2003), a part-of-speech (POS) tag kernel, a Suffix-3 kernel, and a bag-of-words (BOW) kernel. We compare the performance of our approach (MKL) with semi-supervised anomaly detection (SSAD) (Görnitz et al., 2013), support vector data description (SVDD) (Tax and Duin, 1999), and the one-class SVM (OCSVM) (Schölkopf et al., 1999). As before, the baselines cannot use all kernels at a time and are evaluated on every kernel separately. For simplicity, we show only the MKL results for parameter $p = 2$ as all other p-norms that we tried out lead to the same result.

We randomly divide the undisputed papers into training (80%) and holdout (20%) and use the 12 disputed papers for testing. We make sure that training sets contain at least three examples of every author and two articles written jointly by Hamilton and Madison. Otherwise we discard and draw again. We repeat experiments five times with randomly drawn training and holdout sets and report on averaged accuracies for the disputed test set.

Table 2: Results for the disputed articles of the Federalist papers.

	kernel	H&M	M	J	H
MKL	(all)	0	**12**	0	0
SSAD	484fw	0	**12**	0	0
	POS	9	0	3	0
	Suffix3	0	**12**	0	0
	BoW	0	0	0	12
SVDD	484fw	12	0	0	0
	POS	12	0	0	0
	Suffix3	12	0	0	0
	BoW	12	0	0	0
OCSVM	484fw	12	0	0	0
	POS	12	0	0	0
	Suffix3	12	0	0	0
	BoW	12	0	0	0

The results are shown in Table 2. The one-class SVM and SVDD constantly credit the 12 disputed articles as joint work by Hamilton and Madison. The outcome of SSAD highly depends on the kernel function; while the part-of-speech kernel distributes the papers on Jay (3) and Hamilton and Madison (9), respectively, the bag-of-words kernel assigns all documents to Hamilton. By contrast, SVDD with function word and Suffix-3 kernels attribute the articles to Madison. The same outcome is observed for our novel MKL that also credits the 12 papers to Madison. Thus, MKL and SSAD with function words and BoW kernel confirm todays assumption that all 12 papers have been written by Madison. However, choosing SSAD as the base classifier in the absence of prior knowledge leaves much room for interpretations and the user in the need of deciding between three solutions, depending on which kernel she prefers. By using our MKL, selecting features and or kernel functions is no longer necessary as the learning algorithm itself picks the right combination of kernels for the problem at hand. Thus, the more kernels are thrown into, the richer the decision space for the MKL.

5 Conclusion

We proposed a universal method for authorship detection. Our approach built upon semi-supervised anomaly detection and generalised existing techniques to utilise multiple kernels; a requirement which is particularly beneficial for authorship attribution as features are usually tailored to tasks at hand and do not necessarily translate well to other authors. Our method is proven to converge to the optimal solution and simple to implement. Our empirical results show the robustness of our approach as it consistently outperforms baseline competitors on a subset of Reuters 50-50 or confirms common knowledge wrt the authorship of disputed articles of the Federalist Papers. The main advantage of the method however lies in its simplicity. Practitioners do not need to take critical design choices in terms of which features to use and which not. By contrast, all features (kernels) can be used in the optimisation and the method itself finds the optimal combination for the problem at hand.

Acknowledgements

Jamal Abdul Nasir is supported by a grant from the Higher Education Commission, H-9 Islamabad, Pakistan. Nico Görnitz is supported by the German Bundesministerium für Bildung und Forschung

(BMBF FKZ 01GQ0850 and 01IB001A). Ulf Brefeld is also affiliated with the German Institute for Educational Research (DIPF), Frankfurt/Main, Germany.

References

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. 2004. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proc. of the International Conference on Machine Learning*.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. 2010. Semi-Supervised Novelty Detection. *Journal of Machine Learning Research*, pages 2973–2973–3009–3009, December.

J. F. Burrows. 1987. *Computation into Criticism: A Study of Jane Austen's Novels and an Experiment in Method*. Oxford: Clarendon Press.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. 2006. *Semi-supervised learning*. {MIT} Press.

J. Diederich, J. Kindermann, E. Leopold, and G. Paass. 2003. Authorship attribution with support vector machines. *Applied Intelligence*, 19(1):109–123.

G. Forman. 2003. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305.

Glenn Fung. 2003. The disputed federalist papers: Svm feature selection via concave minimization. In *Richard Tapia Celebration of Diversity in Computing Conference*.

Nico Görnitz, Marius Kloft, and Ulf Brefeld. 2009. Active and semi-supervised data domain description. In *ECML*, pages 407–422. Springer.

Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. 2013. Toward Supervised Anomaly Detection. *Journal of Artificial Intelligence Research (JAIR)*, 46:235–262.

David I. Holmes and Daniel W. Crofts. 2010. The diary of a public man: a case study in traditional and non-traditional authorship attribution. *LLC*, 25(2):179–197.

David I. Holmes. 1998. The Evolution of Stylometry in Humanities Scholarship. *Literary and Linguistic Computing*, 13(3):111–117, September.

J. Houvardas and E. Stamatatos. 2006. N-gram feature selection for authorship identification. In *AIMSA*.

Dmitry V. Khmelev. 2000. Disputed authorship resolution through using relative empirical entropy for markov chains of letters in human language texts. *Journal of Quantitative Linguistics*, 7(3):201–207.

M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. 2011. lp-Norm Multiple Kernel Learning. *JMLR*, 12:953–997.

M. Koppel and J. Schler. 2003. Exploiting stylistic idiosyncrasies for authorship attribution. In *Proceedings of the International Joint Conference on Artificial Ingelligence*.

M. Koppel, N. Akiva, and I. Dagan. 2006. Feature instability as a criterion for selecting potential style markers: Special topic section on computational analysis of style. *Journal of the American Society for Information Science and Technology*, 57(11):1519–1525.

M. Koppel, J. Schler, and S. Argamon. 2011. Authorship attribution in the wild. *Language Resources & Evaluation*, 45(1):83–94.

G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. I. Jordan. 2004. Learning the kernel matrix with semi-definite programming. *JMLR*, 5:27–72.

J. Li, R. Zheng, and H. Chen. 2006. From fingerprint to writeprint. *Communications of the ACM*, 49(4):76–82.

Hermann Maurer, Frank Kappe, and Bilal Zaka. 2006. Plagiarism – a survey. *Journal of Universal Computer Science*, 12(8)8:1050–1084.

T. C. Mendenhall. 1887. The characteristic curves of composition. *Science*, ns-9(214S):237–246.

Frederick Mosteller and David L. Wallace. 1964. Inference and Disputed Authorship: The Federalist. Springer-Verlag, New York. 2nd Edition appeared in 1984 and was called *Applied Bayesian and Classical Inference*.

Klaus-Robert Müller, Sebastian Mika, Gunnar Rätsch, Koji Tsuda, and Bernhard Schölkopf. 2001. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201.

Antonio Neme, Blanca Lugo, and Alejandra Cervera. 2011. Authorship attribution as a case of anomaly detection: A neural network model. *Int. J. Hybrid Intell. Syst.*, 8(4):225–235.

S. Raghavan, A. Kovashka, and R. Mooney. 2010. Authorship attribution using probabilistic context-free grammars. In *Proceedings of the ACL 2010 Conference*.

Alain Rakotomamonjy, Francis R. Bach, Stephan Canu, and Yves Grandvalet. 2008. SimpleMKL. *JMLR*, 9:2491–2521.

Joseph Rudman. 1997. The state of authorship attribution studies: Some problems and solutions. *Computers and the Humanities*, 31(4):351–365.

B Schölkopf, J C Platt, J Shawe-Taylor, a J Smola, and R C Williamson. 1999. Estimating the support of a high-dimensional distribution. Technical report, July.

Y. Seroussi, I. Zukerman, and F. Bohnert. 2011. Authorship attribution with latent dirichlet allocation. In *Proceedings of the 15th International Conference on Computational Natural Language Learning*.

Efstathios Stamatatos, Nikos Fakotakis, and George K. Kokkinakis. 2000. Automatic text categorization in terms of genre and author. *Computational Linguistics*, 26(4):471–495.

E. Stamatatos, N. Fakotakis, and G. Kokkinakis. 2001. Computer-based authorship attribution without lexical measures. In *Computers and the Humanities*.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.

Enhua Tan, Lei Guo, Songqing Chen, Xiaodong Zhang, and Yihong (Eric) Zhao. 2013. Unik: Unsupervised social network spam detection. In *Proceedings of CIKM*.

D. Tax and R. Duin. 1999. Data domain description using support vectors. In *Proceedings of the European Symposium on Artificial Neural Networks*, volume 256, pages 251–256. Citeseer.

G. Udnv Yule. 1944. *The Statistical Study of Literary Vocabulary*. Cambridge University Press.

G. K. Zipf. 1932. *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press.

Automatic Prediction of Text Aesthetics and Interestingness

Debasis Ganguly
CNGL,
School of Computing,
Dublin City University,
Dublin 9, Ireland
dganguly@computing.dcu.ie

Johannes Leveling
CNGL,
School of Computing,
Dublin City University,
Dublin 9, Ireland
jleveling@computing.dcu.ie

Gareth J.F. Jones
CNGL,
School of Computing,
Dublin City University,
Dublin 9, Ireland
gjones@computing.dcu.ie

Abstract

This paper investigates the problem of automated text aesthetics prediction. The availability of user generated content and ratings, e.g. Flickr, has induced research in aesthetics prediction for non-text domains, particularly for photographic images. This problem, however, has yet not been explored for the text domain. Due to the very subjective nature of text aesthetics, it is difficult to compile human annotated data by methods such as crowd sourcing with a fair degree of inter-annotator agreement. The availability of the Kindle "popular highlights" data has motivated us to compile a dataset comprised of human annotated aesthetically pleasing and interesting text passages. We then undertake a supervised classification approach to predict text aesthetics by constructing real-valued feature vectors from each text passage. In particular, the features that we use for this classification task are word length, repetitions, polarity, part-of-speech, semantic distances; and topic generality and diversity. A traditional binary classification approach is not effective in this case because non-highlighted passages surrounding the highlighted ones do not necessarily represent the other extreme of unpleasant quality text. Due to the absence of real negative class samples, we employ the MC algorithm, in which training can be initiated with instances only from the positive class. On each successive iteration the algorithm selects new *strong negative* samples from the unlabeled class and retrains itself. The results show that the mapping convergence (MC) algorithm with a Gaussian and a linear kernel used for the mapping and convergence phases, respectively, yields the best results, achieving satisfactory accuracy, precision and recall values of about 74%, 42% and 54% respectively.

1 Introduction

Since their inception, Amazon Kindle device[1] and Apps for other general purpose hand-held devices, have led to a massive increase in the trend of reading e-books over paper printed ones. The Amazon Kindle and the Kindle Apps provide a very simple mechanism for highlighting a piece of text and sharing it on social media. The most popular highlighted pieces of text are shown in the Kindle device with an intention to help readers focus on passages that are pleasing or interesting to the greatest number of people. Every month, Kindle customers highlight millions of book passages that are meaningful to them[2]. The general trend among Kindle readers, while reading the classic English literary works, is to highlight text passages that are associated with a high aesthetic quality. An example highlighted passage is shown in Figure 1.

With the availability of such highlighted text, which may be considered as text passages which most readers find pleasing to read, an interesting research problem is to attempt automatic prediction of highlighted pieces of text. In other words, given a text passage, the objective is to

[1] https://kindle.amazon.com/

[2] https://kindle.amazon.com/most_popular

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 905–916, Dublin, Ireland, August 23-29 2014.

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair.

Figure 1: Passage from *A tale of two cities* (Charles Dickens), highlighted by 6843 Kindle readers.

determine the likelihood of it being aesthetically pleasing and interesting. Such an automated approach of identifying aesthetically pleasing text passages may potentially be used to endorse a newly released book on e-commerce websites with an aim to increase its sales. Moreover, such an approach may also, in principle, be used as a tool by an author to determine how likely it is for readers to appreciate a newly written text passage.

The key challenge in solving this problem is to determine the characteristic attributes of a popular highlighted text passage. An intuitive assumption is that the popularity of a highlighted passage depends on its aesthetic quality. Generally speaking, passages inclined towards expressing an author's view on a subject, which may often be philosophical in nature, with considerable application of atypical figures of speech, e.g. anaphora, alliteration, antithesis, metaphor, simile, personification etc., are more likely to be highlighted than a straight-forward story narrative passage. For example, the highlighted passage in Figure 1 is rich in anaphora (repetition of the same word or group of words in a paragraph, e.g. "times", "age", "epoch" etc.) and antithesis (juxtaposition of opposing or contrasting ideas, e.g. "best of times", "worst of times"; "wisdom", "foolishness" etc). An automated approach of aesthetic quality prediction thus has to take into account these different features of a text passage. The idea of using these features for text aesthetics prediction, in fact, forms a core part of our work.

It is particularly interesting to see that this problem of automatically predicting text aesthetics is largely different from the standard well researched problem of document text classification (Sebastiani, 2002). The reason is as follows. The problem of text categorization can effectively be solved by the application of discrete categorical features, such as character n-gram frequencies and word frequencies. In other words, the presence of characteristic words from a particular domain is a good indicator of the class of a document, e.g. the presence of the words "soccer", "goal" etc. in a document is a good indicator that the document is of the sports genre, whereas the presence of words such as "money", "bank" etc. would indicate that the genre is finance. Consequently, the generative framework of a multinomial Naive Bayes (NB) model with character n-gram and word n-grams based features works effectively for this class of problems (McCallum and Nigam, 1998).

In the case of aesthetic quality prediction, however, the mere presence of a particular word or character n-gram can hardly be a good indicator of the inherent literary quality of the text. The output classes of this classification problem, namely *aesthetic* or *not aesthetic*, do not comprise a small vocabulary of domain-specific representative terms such as in the case of the sports or finance domains. The vocabularies of the respective classes in this classification problem are largely unrestricted and mutually indistinguishable.

The rest of the paper is organized as follows. Section 2 presents related research. In Section 3, we present our proposed approach to solve the text aesthetics problem. Section 4 describes our experimental settings, following which Section 5 presents the results. In Section 6, we investigate the contribution from individual features and then the relative importance of the features when used in combination. Finally, Section 7 concludes the paper.

2 Related Work

A computational viewpoint of aesthetic quality, in general, takes into account the subjectivity of an observer and postulates that among several observations, the aesthetically most pleasing one

is the one with the shortest description, given the observer's previous knowledge (Schmidhuber, 2010). An agent driven reinforcement based learning algorithm can then be used in principle to produce creative (novel and interesting) outputs (Schmidhuber, 2010). Our work in this paper is largely different from the general reinforcement learning paradigm, because we focus on the particular problem of text aesthetics viewing the problem as a supervised classification task. Moreover, the proposition of minimum description length as an attribute of aesthetic quality (Schmidhuber, 2010) is counter-intuitive for literary works.

There has been considerable research interest in automatically predicting visual aesthetic quality of images (Dhar et al., 2011) and layout of web pages (Reinecke et al., 2013). Most empirically successful approaches to image aesthetics prediction first transform an image into a feature vector of characteristic attributes that play a pivotal role in differentiating an *interesting* image from a *non-interesting* one. Generally speaking, some of these attributes which determine whether an image is aesthetically pleasing are the presence of salient objects (indicated by a low depth of field), compositional attributes (e.g. the rule of thirds), the effect of light in natural landscapes, etc. The next step is to apply a supervised learning algorithm, e.g. support vector machine (SVM), to learn a two-class prediction model. Useful features, extracted from images for this classification task include: i) colourfulness, contrast, symmetry, vanishing point and facial features (Jiang et al., 2010); ii) face poses, between-face distances, and the consistency of expressions on multiple faces (Li et al., 2010); iii) high level describable attributes, such as compositional attributes (e.g. rule of thirds image layout), content attributes related to the presence of people, animals, sky illumination attributes etc. (Dhar et al., 2011).

Our proposed method of text aesthetics prediction is similarly based on extracting characteristic features from the text passages. However, in the case of literature, it is worth mentioning that in contrast to image aesthetics it is more difficult to describe the subtle attributes which differentiate an aesthetically pleasing text from its counterpart.

Although the authors are not aware of any reported research on text aesthetics, there has been a considerable amount of research in the somewhat closely related problem of detecting metaphors in text. Automated approaches to metaphor detection involve both supervised and unsupervised approaches, some of which include: i) supervised classification on extracted verbal target feature vectors of sentences (Gedigian et al., 2006); ii) expectation maximization (EM) based unsupervised approach to non-literal word sense detection (Birke and Sarkar, 2006); iii) unsupervised approach using hierarchical graph factorization clustering (Shutova and Sun, 2013).

In general, it is intuitive to assume that metaphorical or figurative parts of text are aesthetically pleasing and interesting, which makes the problem of text aesthetics prediction somewhat similar to that of metaphor detection. Unfortunately, this assumption is not often true, and this is particularly the case for literary works due to the availability of a large number of figures of speech at an author's disposal (metaphor just being one of them). For example, the sample Kindle highlighted passage shown in Section 1 has an obvious aesthetic appeal to a large number of readers, in spite of it being not metaphorical.

3 Our Approach to the Text Aesthetics Prediction Problem

In this section, we describe the details of our approach to text aesthetics prediction. We hypothesize that a NB classifier with word or character n-gram based features is not suitable for this particular problem due to the mutual overlap and lack of domain specific restriction in the vocabulary of the output classes (i.e. aesthetic and non-aesthetic). One thus needs to extract a set of characteristic features from the text passages which may be useful to solve the classification problem. We describe the features used in our approach in Section 3.1. In Section 3.2, we propose to use the mapping convergence (MC) algorithm for the text aesthetics problem, where the intention is to learn a classifier only from positive samples.

The truth is rarely pure and never simple. Modern life would be very tedious if it were either, and modern literature a complete impossibility!

Figure 2: Passage from *The Importance of Being Earnest* (Oscar Wilde).

3.1 Feature Vector Encoding of Text Passages

In this section, we introduce the various features used for the text aesthetics classification task. Each feature is a function which maps a passage of text $P = \{w_1 \dots w_N\}$ comprising N words into a real number.

3.1.1 Word-based Features

In Section 1, we illustrated that that an anaphora is a rheoteric device used by authors to emphasize a text passage, which in turn indicates that such a passage is likely to attract the attention of readers and hence are likely to be highlighted by them. Moreover, the closer the repetitions are, the stronger is the emphasis.

On the basis of this reasoning, we employ an average positional difference weighted count of word repetitions in a passage. To be more precise, for each word in a passage we compute the number of times a word w_i is repeated, divide this count by the difference between the repeating position (say at position j), and average the sum of counts for all repeating words over the passage length, as shown in Equation 1. In Equation 1, $\mathbb{1}(w_i = w_j)$ is the indicator function which is 1 if and only if $w_i = w_j$ and 0 otherwise.

The second word level feature which we use, is the average length of words in a passage. The reasoning behind using this feature is that authors tend to use relatively longer words (e.g. superlatives) to emphasize a passage. Equation 2 shows how this is computed.

$$W_1(P) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{\mathbb{1}(w_i = w_j)}{j - i} \tag{1}$$

$$W_2(P) = \frac{1}{N} \sum_{i=1}^{N} len(w_i) \tag{2}$$

3.1.2 Topic-based Features

An attribute which can be considered responsible for the aesthetic quality of a text passage is the diversity of topics it expresses. It is reasonable to assume that a text passage expressing a broad idea or opinion of an author, often philosophical in nature, is likely to be appealing to readers. Such general themed text passages typically cover a broad range of topics, as a result of which the constituent words of such text passages involve collocation of seemingly unrelated terms. For example, in the text passage shown in Figure 2, the word pairs (*truth*, *tedious*), and (*literature*, *impossibility*) would typically appear in different topic classes, where by a topic we mean a set of words with high co-ocurrence likelihood estimated from a collection of documents by standard topic modelling techniques such as the Latent Dirichlet allocation (LDA) (Blei et al., 2003). To encode this diversity of topics as a real valued feature function, we use Equation 3.

$$T_1(P) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{\mathbb{1}[z(w_i) \neq z(w_j)]}{(j - i)} \tag{3}$$

In Equation 3, $z(w)$ denotes the topic class of the word w obtained with the help of LDA. A mismatch in the topic class is divided by the distance between the mismatches to assign more weight to the close mismatches. As an example, the mismatch between (*literature*, *impossibility*) bears more importance than the mismatch between (*modern*, *impossibility*).

The second topic-based feature which we use pertains to predicting the abstractness of the content of a passage. It has been reported that words highly representative of topics are generally

not metaphorical. We apply a similar reasoning to hypothesize that since an interesting piece of text is more likely to be philosophical or abstract in nature in comparison to a story narrative, the constituent words are less likely to be the representatives of their topic classes. Formally speaking in terms of LDA, these words are expected to have smaller values of $\max_k \phi_k(w)$. Recall that a topic representative word in LDA exhibits a skewed distribution with a peak for one topic class (with a high value of $\max_k \phi_k(w)$), whereas a less representative word exhibits a more uniform distribution of $\phi_k(w)$ values over the topic classes (thus a low value of $\max_k \phi_k(w)$). We use Equation 4 to compute the average topic concreteness of a text passage.

$$T_2(P) = \frac{1}{N} \sum_{i=1}^{N} \max_k \phi_k(w_i) \tag{4}$$

3.1.3 Part of Speech Feature

We hypothesize that another attribute of an aesthetic passage is that it is likely to contain a rich usage of adjectives (mostly of superlative type for the sake of emphasis) and adverbs. We therefore employ the part of speech tag (POS) information of the constituent words of a text passage as one of our features. To be more specific, we use the average number of adjectives and adverbs of a text passage as the feature value. This is shown in Equation 5.

$$POS(P) = \frac{1}{N} \sum_{i=1}^{N} (\#adjectives + \#adverbs) \tag{5}$$

3.1.4 Sentiment Feature

We pointed out in Section 1 that authors often use the *antithesis* figure of speech to express contrasting concepts. Thus, another feature which we can use is the aggregated absolute difference values between the sentiment polarities of words in a text paragraph. This again is weighted by the difference in position between a positive sentiment word and its negative counterpart to assign more importance to closely occurring opposite sentiment concepts.

To obtain the sentiment values of the constituent words, we used the SentiWordNet[3]. To illustrate with an example, consider the closely occurring opposite sentiment word pairs (*best* (0.75), *worst* (-0.75)), (*wisdom* (0.375), *foolishness* (-0.375)) etc. of Figure 1 and the word pairs (*complete* (0.625), *impossibility* (-0.25)) of Figure 2, where the numbers in the parentheses show the positive or the negative sentiment value (a normalized number between 0 and 1). Equation 6 shows the real-valued function derived from the sentiment information of word pairs, where the function $s(w)$ denotes the sentiment value associated with the word w.

$$SENT(P) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{|s(w_i) - s(w_j)|}{(j-i)} \tag{6}$$

3.1.5 Inter-word Semantic Distance Feature

An alternative way to represent the topic diversity is to capture the likelihood of the event of occurrence of two words in close vicinity. The higher this likelihood is, the better is the semantic relation or coherence between the words. We make use of the DISCO[4] tool to compute the semantic relation between two words in a word pair. In DISCO, these semantic relations between the words are precomputed on the basis of co-occurrence likelihoods from a large corpus, e.g. the Wikipedia (Kolb, 2008). DISCO provides two similarity measurements (named the first order and the second order similarities) between two input words. While the first order similarity between two input words is computed based on their collocation sets, the second order similarity is computed based on their sets of distributionally similar words (Kolb, 2008). We denote the

[3] `http://sentiwordnet.isti.cnr.it/`

[4] `http://www.linguatools.de/disco/disco_en.html`

first order and the second order similarities between words w_i and w_j respectively as $ds_1(w_i, w_j)$ and $ds_2(w_i, w_j)$ respectively.

In relation to text aesthetics, we expect a small value of average first order and second order similarity values between word pairs in a highlighted piece of text in comparison to a non-highlighted one. Similar to our earlier features, we divide these similarity values by the positional difference between the words in order to put more emphasis on semantic diversity between closely occurring words. Equation 7 shows the two features extracted making use of these similarity values.

$$SD_k(P) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{|ds_k(w_i) - ds_k(w_j)|}{(j-i)},\ k = \{1, 2\} \tag{7}$$

3.2 Learning from Positive Examples: The MC Algorithm

Binary classifiers, such as SVMs, work particularly well with a sufficient number of both positive and negative class instances for training. In the case of text aesthetics prediction problem, the passages highlighted by Kindle readers serve as the positive class samples. Although it might be intuitive to use the non-highlighted passages as instances of the negative type, there can be problems associated with this approach.

Firstly, the non-highlighted passages are not essentially instances of the negative class because the non-highlighted passages are not necessarily aesthetically unpleasing. Secondly, there is an element of cognitive bias associated with the highlighting process because a reader, who can already see popular highlights while reading a page, may be biased to highlight the same passage himself, and may not in fact highlight some other passage which he himself found interesting.

Note that this observation in fact makes our problem more challenging to solve in comparison to aesthetics prediction in other domains, such as images, where information such as Flickr[5] photo ratings can be used as strong positive or negative indicators of an image interestingness or aesthetic quality, leading to effective classification results using a standard binary classification approach (Dhar et al., 2011).

Due to the presence of incompletely labeled examples, we apply the *mapping convergence* (MC) algorithm (Yu et al., 2003) for this task. The objective of the MC algorithm is to predict the positive samples from a test data, given a mixture of positive and unlabeled samples. These unlabeled samples in the MC algorithm can be treated as instances of either the positive or the negative class in order to obtain maximum classification effectiveness.

The two stages of the MC algorithm are summarized as follows.

1. The *mapping* stage identifies from the unlabeled samples the strong negative ones, i.e. the points distinctly different from the positive samples.
2. The *convergence* stage is an iterative step to learn a binary classification model, e.g. SVM, using the positive and the strong negative samples. Each iterative step of convergence classifies the remaining unlabeled samples to collect more strong negative samples. The convergence step is repeated until no more strong negative samples are found.

The objective of the convergence step of the MC algorithm is to maximize margin to make progressively better approximation of the negative data. At the end of the iteration, the class boundary eventually converges to the boundary around the positive data set in the feature space (Yu et al., 2003).

In our approach to the text aesthetics prediction task, we implement the mapping stage of the MC algorithm with the help of standard one-class classifiers, namely the one class SVM (OSVM) (Schölkopf et al., 1999) and the support vector data descriptor (SVDD) (Tax and Duin, 2004). The OSVM separates all the data points in the feature space from the origin, with the help of a separating hyperplane with maximum distance from the origin. The OSVM is thus

[5] `https://www.flickr.com/`

able to separate out regions in the input space with high probability densities (Schölkopf et al., 1999). SVDD, on the other hand, instead of a planar, takes a spherical approach to the one class problem. The algorithm obtains a spherical boundary in feature space around the data. The volume of this hypersphere is minimized to minimize the effect of incorporating outliers in the solution (Tax and Duin, 2004).

It is worth mentioning here that although the OSVM and the SVDD can be trained with positive samples only, these models are prone to over-fitting or under-fitting due to a small number of support vectors modeled from a small number of positive samples (Yu et al., 2003). In contrast, a binary SVM can model data more robustly due to the presence of the additional negative samples. Hence, OSVM and the SVDD are typically used as a weak classifier to obtain a set of initial strong negative samples in order to initiate the convergence step of the MC algorithm.

4 Experiment Settings

In this section, we describe the dataset and the tools used for our experiments.

4.1 Dataset Construction

The standard practice to evaluate the metaphor detection problem, which is somewhat similar to the text aesthetics prediction, is to make extensive use of manually annotated data typically obtained under controlled user-based studies, where the users or the participants are instructed to perform some given objectives, such as manually label metaphors in a collection of documents, e.g. (Hovy et al., 2013). The main difficulties with this approach are that: i) it takes a considerable amount of time to collect data; ii) the quality of the data depends largely on controlled experimental settings, e.g. the data quality may be susceptible to errors caused by targeted, malicious work efforts, since there is often a financial incentive to complete tasks quickly rather than effectively (Ipeirotis et al., 2010); and iii) it is very difficult to compare the effectiveness of two methods on two different datasets obtained under different controlled user study settings.

The availability of fairly large amounts of highlighted text on the Amazon website has ensured a reliable and fast way to construct the dataset for carrying out the text aesthetics experiments. The advantages are as follows. Firstly, it is not necessary to conduct crowd sourcing experiments for data collection. Secondly, since the data is not generated by controlled crowd sourcing, the quality of the data is more reliable because there is no financial incentive to complete tasks quickly. Thirdly, since the data is publicly available, it is possible to achieve a fair comparison between different problem solving approaches.

The Amazon "Popular Highlights"[6] web page presents a ranked list of the most highlighted passages, sorted in descending order by the number of highlights. However, at the time of writing this paper, Amazon has neither made the data publicly downloadable nor provided an API to access it. For conducting our experiments with this data, we therefore had to automatically crawl data from the Popular Highlights web page.

In addition to the highlighted passages (serving as the positive class samples in our dataset), we also need the non-highlighted ones (meant to serve as the unlabeled samples). The text from the non-highlighted passages, however, are not available in the Popular Highlights web page. This data was thus extracted from those books, the passages of which are popularly highlighted. In order to ensure free access to book content, we had to restrict our dataset to the 50 most popular highlighted classic English fictions.

More precisely speaking, for every highlighted passage found while crawling the Amazon Popular Highlights page, our crawler checks if the book is available on project Gutenberg[7]. If not, then we examine the next highlighted passage, otherwise we craw the full text of the book,

[6] `https://kindle.amazon.com/most_popular/highlights_all_time/`
[7] `http://www.gutenberg.org/`

in which the current highlighted passages belongs, from project Gutenberg website. The crawler continued to run until we had collected highlighted passages from 50 different literature classics.

The dataset for the prediction task is then constructed as follows. First, we add the text of all highlighted passages as instances of the positive class. Next, for each highlighted passage, we add the paragraph preceding and succeeding it into the dataset as the unlabeled samples. Note that selecting the unlabeled samples this way is better than random selection of non-highlighted passages from full text, because this way of choosing negative samples ensures a meaningful representation of reader judgments to highlight a particular passage of text from within a surrounding context.

We then partition the dataset comprised of the positive and unlabeled samples into equal sized training and test sets. In Table 1, we outline the characteristics of the dataset.

Dataset	# Books	Vocab.	# Passages		
		Size	Highlighted	Unhighlighted	Total
Train	25	9560	168	305	473
Test	25	7883	169	319	488
Total	50	13496	337	624	961

Table 1: Dataset characteristics

4.2 Implementation Details

For each passage in the dataset, we extract the features described in Section 3.1. To compute the topic modeling based features we used Mallet[8]. The number of topics (K) in LDA was set to 100. The POS tag feature was extracted with the help of the Stanford POS tagger[9]. For extracting the sentiment feature, we made use of the Java API of the SentiWordNet[10]. For the semantic word distance feature, we used the DISCO Java API[11].

For the naive Bayes experiment, we used the Stanford classifier[12]. The SVM experiments (binary SVM, one-class SVM, SVDD) were conducted with the libSVM software[13].

4.3 Evaluation Metrics

For all the experiments reported in this paper, the classification effectiveness mainly focuses on precision and recall with respect to the positive class. Consequently, precision, recall and the F-score measures, shown in Tables 2 and 3, are measured with respect to the positive class only.

Ideally, for this problem one would want to obtain a high recall, i.e. identify as many highlighted passages correctly as possible. In this situation, recall is thus more important than precision. Achieving a good precision is desirable, nonetheless, to minimize the false positives. Although we report accuracy, we emphasize that accuracy alone is not a good measure of classification effectiveness in this case, because correct identification of negative instances is not important for this problem.

5 Results

Before conducting experiments with the MC algorithm, we obtained baseline results by classifying the dataset using NB and SVMs. In the case of NB, instead of using the real valued features from the text passages (as proposed in Section 3.1), we simply used the character n-gram and word n-gram features (maximum value of n was set to 5) from the text, automatically extracted

[8] http://mallet.cs.umass.edu/
[9] http://nlp.stanford.edu/software/tagger.shtml
[10] http://sentiwordnet.isti.cnr.it/code/SentiWordNetDemoCode.java
[11] http://www.linguatools.de/disco/disco_en.html
[12] http://nlp.stanford.edu/software/classifier.shtml
[13] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/

Classifier	Kernel	Accuracy	Precision	Recall	F-score
NB	N/A	67.40	54.40	36.70	43.80
BSVM	Linear	66.19	35.71	5.92	10.15
BSVM	Gaussian	67.00	39.39	15.38	22.13
OSVM	Linear	38.32	32.46	51.48	39.82
OSVM	Gaussian	53.68	41.87	50.29	45.70
SVDD	Linear	35.04	34.77	100.00	51.60
SVDD	Gaussian	37.91	35.56	97.63	52.13

Table 2: Text aesthetics prediction results with Naive Bayes and SVM.

Classifier		Kernel		Accuracy	Precision	Recall	F-score
Mapping	Convergence	Mapping	Convergence				
OSVM	BSVM	Linear	Linear	66.18	35.71	5.92	10.15
OSVM	BSVM	Linear	Gaussian	64.96	40.26	36.69	38.39
OSVM	BSVM	Gaussian	Linear	66.80	44.44	11.83	18.69
OSVM	BSVM	Gaussian	Gaussian	64.34	36.87	39.05	37.93
SVDD	BSVM	Linear	Linear	40.98	35.76	92.90	51.64
SVDD	BSVM	Linear	Gaussian	43.44	36.17	90.53	51.69
SVDD	BSVM	Gaussian	Linear	56.76	42.90	74.64	**54.42**
SVDD	BSVM	Gaussian	Gaussian	47.34	38.60	88.17	53.69

Table 3: Text aesthetics prediction results by the MC algorithm with different settings.

by the Stanford classifier. The result of this experiment (see Table 2) shows that the recall value is very low, which in turn indicates that word vocabulary based features, typically used for text categorization, are not effective for this task.

The next classification method that we employ is standard binary class SVM (denoted as BSVM). The training phase of the BSVM used the non-highlighted passages as negative class instances. We experimented with both linear and Gaussian kernels. For all reported results which use the Gaussian kernel, the parameter γ was set to the default value of 1/(#features) as per the libSVM implementation. Although the accuracy achieved is comparable to NB, the recall achieved is worse, which shows that treating non-highlighted passages as negative class instances is not reasonable for this problem (see Section 6.2 for an illustration).

The recall value is significantly increased with the help of one-class SVM (OSVM). SVDD performs even better in terms of recall. However, SVDD significantly underfits the data because it classifies almost every test data point as an instance of the positive class, thus achieving low accuracy and precision due to the presence of too many false positives.

Our next set of experiments involves the MC algorithm for classification. Since, the mapping phase makes use of only the positive data, we employed both the one-class classifiers used in the experiments of Table 2, i.e. OSVM and SVDD, for this purpose. Mapping with OSVM results in an improvement in the accuracy at the cost of sacrificing recall, which is not desirable for this problem. However, note that the negative samples obtained with the OSVM mapping (with Gaussian kernel) improves the classification effectiveness of the BSVM (compare the fourth row of Table 3 with the second row of Table 2), which indicates that the MC algorithm does improve the classification effectiveness, confirming our hypothesis that it is reasonable not to consider every non-highlighted passage as negative samples.

The problem of SVDD underfitting (as evident from the SVDD results of Table 2) is alleviated by the MC approach. The most effective MC approach uses Gaussian/linear kernels for mapping/convergence (see the seventh row of Table 3). Accuracy is increased to around 56% with a satisfactory recall of around 74%. The use of Gaussian kernel during both the mapping and convergence steps yields a higher recall but at the cost of more false positives (lower accuracy, precision and F-score).

Feature combination vector				Evaluation Metrics			
Word	Topics	POS/Polarity	Semantic	Accuracy	Precision	Recall	F-score
1	0	0	0	36.06	34.88	97.63	51.40
0	1	0	0	37.91	35.74	99.40	52.58
0	0	1	0	36.05	35.01	98.81	51.70
0	0	0	1	42.41	37.03	94.67	53.24
1	1	1	1	56.76	42.90	74.64	**54.42**

Table 4: Individual feature contributions for identifying text aesthetics.

Feature	igain
Topic diversity (T_1)	0.3684
Sentiment ($SENT$)	0.2685
Word repetition (W_1)	0.2509
First-order semantic distance (SD_1)	0.1543
Part-of-speech (POS)	0.1448
Second-order semantic distance (SD_2)	0.1141
Word length (W_2)	0.0732
Topic abstractness (T_2)	0.0526

Table 5: Ranking features by their igain values.

6 Posthoc Analysis

In this section, we comment on the importance of the features used for classification, and also illustrate how the MC algorithm helps in increasing the separability between the classes.

6.1 Feature Importance

First, we investigate the importance of the different features by a selective choice of only one group of features at at time for the classification. The classifier we use for this experiment is MC with a Gaussian SVDD kernel for mapping and a linear SVM kernel for convergence (as per the best settings of Table 3). The results are shown in Table 4 from which it can be seen that the best accuracy is obtained with the use of the semantic distance features.

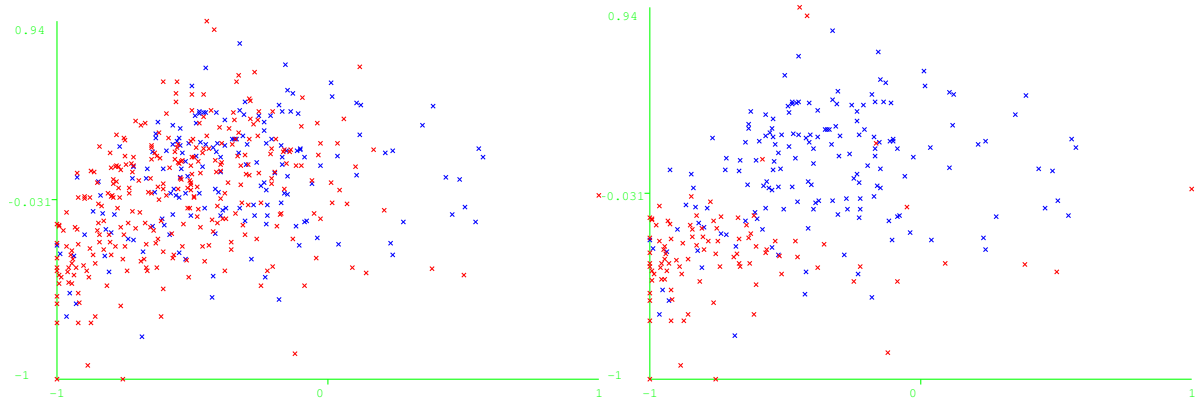
It can be observed that the accuracy values obtained with a single category of features, such as word-based (length and repetition), topic-based (generality and diversity) and so on, are considerably lower than the accuracy value obtained with a combination of all the features (the last row of Table 4. The precision values achieved with these individual feature groups are also considerably lower than the precision of 42.90% of the overall combination.

Next, we find out the relative importance of each feature in their overall combination by ranking the features with the help of a standard feature quality estimator, called *information gain* (*igain*) (Quinlan, 1986). The results are presented in Table 5. It can be seen that the topic diversity is the most discriminative feature having an igain value significantly higher than the second most important one in the list. This observation verifies our hypothesis that aesthetically appealing passages are those constituting terms from diverse topics.

The sentiment and the word repetition features, having close igain values, are second and third respectively in the list. The usefulness of the sentiment feature suggests that contrasting concepts packed in close vicinity of a sentence are likely to be aesthetically pleasing to read. The word repetition feature, on the other hand, suggests that the anaphora figure of speech is likely to be be associated with aesthetically pleasing text.

6.2 Illustration of the usefulness of the MC Algorithm

This section investigates the usefulness of the MC algorithm for the text aesthetics classification. In particular, we show that for this one class classification problem, the MC algorithm can selectively refine the set of unlabeled samples and retrain the model for better separability



(a) Before MC. (b) After MC convergence (5 iterations).

Figure 3: Visualization of the training set in the two most discriminating dimensions, i.e. topic diversity (Y-axis) and sentiment (X-axis).

between the positive and the unlabeled classes.

To illustrate our claim, we first plot the initial training set in two dimensional subspace before the application of MC, i.e. when all the unlabelled instances are treated as negative class samples; this is shown in Figure 3a. The two dimensions that we use for plotting this figure, are the two features having the highest igain values, i.e. the topic diversity (T_1) and sentiment ($SENT$) features. Figure 3a shows that the highlighted text passages (shown in blue) are not well separated from the non-highlighted ones (shown in red).

Next, in Figure 3b, we plot the training set with a reduced number of samples from the negative (non-aesthetic) class obtained after running the MC algorithm. Figure 3b clearly shows that after convergence the MC algorithm has retained only the strong negative samples for training, as is evident from a better visual separation between the classes. A binary classifier, trained on the dataset of Figure 3b, is thus likely to be more effective than that trained with Figure 3a.

7 Conclusions

This paper investigated the problem of automated text aesthetics prediction. As distinguishing features for text aesthetics identification, we applied different statistical features such as word repetitions, topic diversity, part-of-speech, word polarity etc. We collected aesthetically pleasing text passages from the Kindle "popular highlights" website for conducting our experiments. Due to the presence of only positive class samples, i.e. the highlighted passages, in this dataset, we apply the MC algorithm to iteratively train a binary classifier with the strongly negative samples.

The results of our experiments show that the MC algorithm with a Gaussian and a linear kernel applied for the mapping and convergence phases respectively, yields the best results achieving satisfactory recall, precision and F-score values of about 74%, 42% and 54% respectively. Moreover, the results also demonstrate that the topic diversity, word polarity and word repetition are the three most distinguishing features for text aesthetics identification. Furthermore, our results are comparable to those of a somewhat similar problem of figurative text detection where the best reported F-score values achieved are about 54% (Birke and Sarkar, 2006) and 64% (Shutova and Sun, 2013).

Acknowledgments

This research is supported by Science Foundation Ireland (SFI) as a part of the CNGL Centre for Global Intelligent Content at DCU (Grant No: 12/CE/I2267).

References

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language. In *EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, April 3-7, 2006, Trento, Italy*. The Association for Computer Linguistics.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, March.

Sagnik Dhar, Vicente Ordonez, and Tamara L. Berg. 2011. High level describable attributes for predicting aesthetics and interestingness. In *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pages 1657–1664.

Matt Gedigian, John Bryant, Srini Narayanan, and Branimir Ciric. 2006. Catching metaphors. In *Proceedings of the Third Workshop on Scalable Natural Language Understanding*, ScaNaLU '06, pages 41–48, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dirk Hovy, Shashank Shrivastava, Sujay Jauhar, Mrinmaya Sachan, Kartik Goyal, Huying Li, Whitney Sanders, and Eduard Hovy. 2013. Identifying metaphorical expressions with tree kernels. In *Proceedings of NAACL-HLT Meta4NLP Workshop*.

Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, HCOMP '10, pages 64–67, New York, NY, USA. ACM.

Wei Jiang, Alexander C. Loui, and Cathleen Daniels Cerosaletti. 2010. Automatic aesthetic value assessment in photographic images. In *Proceedings of the 2010 IEEE International Conference on Multimedia and Expo, ICME 2010, 19-23 July 2010, Singapore*, pages 920–925.

Peter Kolb. 2008. DISCO: A Multilingual Database of Distributionally Similar Words. In *KONVENS 2008 – Ergänzungsband: Textressourcen und lexikalisches Wissen*, pages 37–44.

Congcong Li, Alexander C. Loui, and Tsuhan Chen. 2010. Towards aesthetics: A photo quality assessment and photo selection system. In *Proceedings of the International Conference on Multimedia*, MM '10, pages 827–830, New York, NY, USA. ACM.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive bayes text classification. In *AAAI/ICML Workshop on Learning for Text Categorization*, pages 41–48.

J. R. Quinlan. 1986. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March.

Katharina Reinecke, Tom Yeh, Luke Miratrix, Rahmatri Mardiko, Yuechen Zhao, Jenny Liu, and Krzysztof Z. Gajos. 2013. Predicting users' first impressions of website aesthetics with a quantification of perceived visual complexity and colorfulness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 2049–2058, New York, NY, USA. ACM.

Jürgen Schmidhuber. 2010. Formal theory of creativity, fun, and intrinsic motivation (1990-2010). *IEEE T. Autonomous Mental Development*, 2(3):230–247.

Bernhard Schölkopf, Robert C. Williamson, Alex J. Smola, John Shawe-Taylor, and John C. Platt. 1999. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999*, pages 582–588. The MIT Press.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, March.

Ekaterina Shutova and Lin Sun. 2013. Unsupervised metaphor identification using hierarchical graph factorization clustering. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 978–988. The Association for Computational Linguistics.

David M. J. Tax and Robert P. W. Duin. 2004. Support vector data description. *Mach. Learn.*, 54(1):45–66, January.

Hwanjo Yu, ChengXiang Zhai, and Jiawei Han. 2003. Text classification from positive and unlabeled documents. In *Proceedings of the 2003 ACM CIKM International Conference on Information and Knowledge Management, New Orleans, Louisiana, USA, November 2-8, 2003*, pages 232–239. ACM.

Triple based Background Knowledge Ranking for Document Enrichment

Muyu Zhang, Bing Qin*, Ting Liu, Mao Zheng
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{myzhang,qinb,tliu,mzheng}@ir.hit.edu.cn

Abstract

Document enrichment is the task of retrieving additional knowledge from external resource over what is available through source document. This task is essential because of the phenomenon that text is generally replete with gaps and ellipses since authors assume a certain amount of background knowledge. The recovery of these gaps is intuitively useful for better understanding of document. Conventional document enrichment techniques usually rely on *Wikipedia* which has great coverage but less accuracy, or *Ontology* which has great accuracy but less coverage. In this study, we propose a document enrichment framework which automatically extracts "$argument_1, predicate, argument_2$" triple from any text corpus as background knowledge, so that to ensure the compatibility with any resource (e.g. *news text*, *ontology*, and *on-line encyclopedia*) and improve the enriching accuracy. We first incorporate source document and background knowledge together into a triple based document-level graph and then propose a global iterative ranking model to propagate relevance score and select the most relevant knowledge triple. We evaluate our model as a ranking problem and compute the MAP and $P\&N$ score to validate the ranking result. Our final result, a MAP score of 0.676 and $P\&20$ score of 0.417 outperform a strong baseline based on search engine by 0.182 in MAP and 0.04 in $P\&20$.

1 Introduction

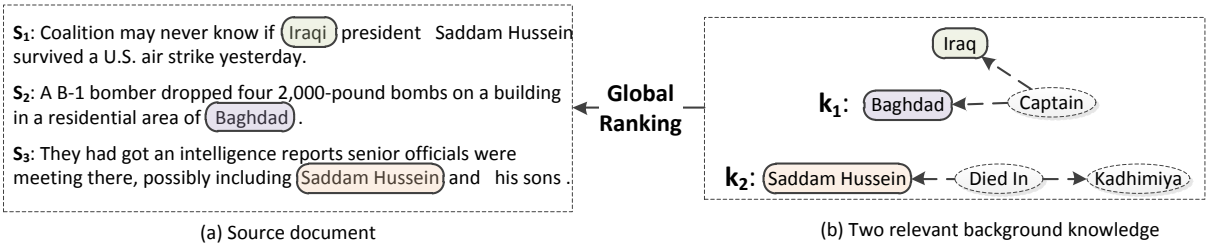
Document enrichment is the task to acquire background knowledge from external resources and recover the omitted information automatically for certain document. This task is essential because authors usually omit basic but well-known information to make the document more concise. For example, author omits "*Baghdad is the captain of Iraqi*" in the text of Figure 1 (a), which is well-known to readers. During reading process, these gaps will be automatically plugged effortlessly by the background knowledge in human brain. However, the situation is different for machine because it lacks the ability to acquire and select the proper background knowledge, which limits the performances of certain NLP applications. Document enrichment has been proved helpful in these tasks such as web search (Pantel and Fuxman, 2011), coreference resolution (Bryl et al., 2010), document cluster (Hu et al., 2009) and entity disambiguation (Bunescu and Pasca, 2006; Sen, 2012).

In the past, there are mainly two kinds of document enrichment researches according to the resource they relying on. The first line of works make use of *WikiPedia*, the largest available on-line encyclopedia as resource and link the entity (e.g. *Baghdad*) of document to its corresponding Wiki page (e.g. *Baghdad* [1] in WikiPedia), so that to enrich the document with the context of Wiki page (Bunescu and Pasca, 2006; Cucerzan, 2007; Han et al., 2011; Kataria et al., 2011; Sen, 2012; He et al., 2013). Despite the great success of these methods, there remain a great challenge that not all information in the linked Wiki page is helpful to the understanding of corresponding document. For example, the Wiki page of *Baghdad* contains lots of information about city history and culture, which are not quite relevant to the semantic of context in Figure 1 (a). So treating the whole Wiki page as the enrichment to document may cause noise

*Corresponding author.

[1] http://en.wikipedia.org/wiki/Baghdad

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 917–927, Dublin, Ireland, August 23-29 2014.



(a) Source document

(b) Two relevant background knowledge

Figure 1: An example of document enrichment with background knowledge: (a) source document talking about a U.S. air strike aiming at Saddam in Baghdad (b) two important relevant information, which is omitted in source document but acquired by our model and enriched as background knowledge .

problem. Another line of works rely on the Ontologies constructed with supervision or even manually which have great accuracy but less coverage (Motta et al., 2000; Passant, 2007; Fodeh et al., 2011; Kumar and Salim, 2012). Besides, these methods usually rely on special ontology which is rather difficult to construct and in turn limits the coverage and application of these methods.

Ideally, we would wish to integrate both coverage and accuracy, where an triple based background knowledge ranking model may help. Our framework extracts knowledge from any corpus resource including WikiPedia to ensure coverage and present knowledge as "$argument_1, predicate, argument_2$" triple to reduce noise. This model ranks background knowledge triples according to their relevance to the source document. The key idea behind the model is that document is constructed by several units of information, which can be extracted automatically. For every background knowledge b extracted automatically from a relevant corpus, the more units are relevant to b and the more important they are, the more relevant b becomes to the source document. Thus, we extract both source document information and background knowledge automatically and present them together in a document-level graph. Then we propagate the relevance score from the source document information to the background knowledge during an iterative process. After convergence, we obtain the $Top\ n$ relevant background knowledge, rather than retrieving all of them without filtering.

To evaluate our model, we use ACE [2] corpus as source documents and output the ranked list of background knowledge. Then we train three annotators to check the ranking result and annotating whether certain knowledge is relevant to corresponding source document separately. We totally annotated more than 7000 background knowledge by three annotators. We evaluate their annotation consistence by computing the $Fleiss'\ Kappa$ (Fleiss, 1971), a famous criterion in multi-annotator consistence evaluation. We achieve a $Fleiss'\ Kappa$ of value 0.8066 in best situation and 0.7076 in average, which indicates the great consistence between three annotators. The ranking result is evaluated with MAP score and $P\&N$ score (Voorhees et al., 2005). We finally achieve a MAP score of 0.676 and $P\&20$ score of 0.417 in $Top\ 20$ background knowledge, which are higher by 0.182 and 0.04 than a strong baseline based on search engine. We also evaluate the effect of the automatically extraction to source document and background knowledge, which is key to the performance of our method in real application.

2 Triple Graph based Document Representation

We believe that different parts of document are related to each other, rather than isolated. Hence, we propose a *triple graph* based document representation to incorporate source document information and background knowledge. In this presentation, "$argument_1, predicate, argument_2$" triple serves as node and the edge between nodes indicates their semantic relevance. In this part, we introduce *triple graph* and the way to extract source document information and background knowledge automatically.

2.1 Motivation for triple presentation

Compared to Wiki Page, triple based enrichment helps to reduce noise illustrated in Section 1. Compared to bag of words, triple based presentation help to reduce ambiguity of single word which is shown in

[2] http://catalog.ldc.upenn.edu/LDC2006T06

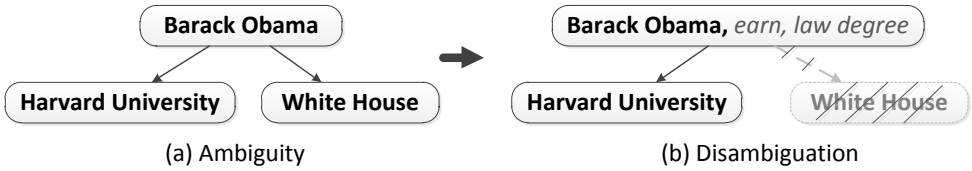


Figure 2: The motivation for the form of triple (a) relevance ambiguity of single word *Obama*, which is related to Harvard and White House (b) disambiguation with the help of other triple elements, where "*earn, law degree*" help to limit *Obama* to the graduate of Harvard.

Figure 2. Figre 2 (a) shows that the single word of *Obama* is related to multiple semantic information such as *Harvard University* as a law graduate and *White House* as the president. After introducing the information from other elements of the triple, "*earn, law degree*" help to disambiguate and limit *Obama* to the law graduate of Harvard University only in Figure 2 (b). The form of triple has been used as the presentation of knowledge in some researches such as knowledge base (Hoffart et al., 2013).

2.2 Nodes in the Graph

There are two kinds of nodes in the triple graph: source document nodes (*sd-nodes*) and background knowledge nodes (*bk-nodes*). Both of them are extracted automatically with *Open Information Extraction (Open IE)* technology which focuses on extracting assertions from massive corpora without a pre-specied vocabulary (Banko et al., 2007). Open IE systems are *unlexicalized*-formed only in terms of syntactic tokens and closed-word classes, instead of specific nouns and verbs at all costs.

There are existing Open IE systems such as TextRunner (Banko et al., 2007), WOE (Wu and Weld, 2010), and StatSnowball (Zhu et al., 2009). The output of these systems has been used to support many NLP tasks such as learning selectional preference (Ritter et al., 2010), acquiring sense knowledge (Lin et al., 2010), and recognizing entailment (Schoenmackers et al., 2010). In this work, we use the famous Open IE system ***Reverb*** (Etzioni et al., 2011), which is generated from TextRunner (Etzioni et al., 2008), to extract source document information and background knowledge automatically. We use the newest version of *ReVerb* (version 1.3) without modification, which is free download on-line [3].

Source document node (*sd-node*) Sd-nodes consists of the information extracted from source document automatically by open information extraction technology (Banko et al., 2007), especially Reverb, the famous Open IE system developed by University of Washington (Etzioni et al., 2011). The output of ReVerb is formed as "$argument_1, predicate, argument_2$", which is naturally presented as triple. In this study, we use ACE corpus as source documents and all sd-nodes are extracted by ReVerb. The setup of automatic extraction makes our method usable in many real applications. To evaluate the effect of automatic extraction, we also use the golden annotation within ACE (Doddington et al., 2004) corpus as source document information and compare the performance that with automatic extraction.

Background knowledge node (*bk-node*) Bk-nodes consist of the background knowledge extracted from external corpus resources automatically by Reverb too. We do not rely on certain existed knowledge base and extract background knowledge from external corpus resources for corresponding source document. This setup makes our methods usable in many real applications. Although we do not rely on special knowledge base, we do adapt our method for the existed knowledge base such as YAGO (Hoffart et al., 2013) and compare the performance to evaluate the effect of different knowledge sources.

2.3 Edges in the Graph

The edges between two nodes indicate their semantic relevance, which is evaluated in Section 3.1. There are two kinds of edges: (1) *sd-node* to *sd-node* (2) *sd-node* to *bk-node*, both of them are undirected. Considering all the relevance score originating from sd-nodes, we connect no edge between *bk-nodes*.

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

Edges between sd-nodes All sd-nodes are extracted from the same document, so they should be related to each other. We connect each pair of sd-nodes with an edge and set the weight of edge as their semantic relevance computed in Section 3.1. With this setup, we combine the source document as a whole where different parts affect each other through the edge.

Edges between sd-node and bk-node The basic idea of our model is to propagate relevance score from the sd-nodes to bk-nodes. Hence, we connect each pair of sd-node and bk-node with an edge and set the weight of the edge as their relevance computed in Section 3.1. These edges are all undirected, which indicates that bk-nodes also affect the relevance score of the sd-nodes during the ranking process.

3 Global Ranking Model

In this study, source document *D* is presented as the graph of sd-nodes. For every background knowledge *b*, the task of evaluating the relevance between *b* and *D* is naturally converted into evaluating the relevance between *b* and the graph of sd-nodes. So the relevance between *b* and document *D* can be computed by propagating the *relevance score* from every sd-node of *D* to *b* iteratively. After the convergence, the relevance between *b* and *D* can be evaluated by the *relevance score* of *b*. Intuitively, three factors affect their relevance:

- How many sd-nodes is *b* relevant to ?
- How relevant is *b* to these sd-nodes?
- How important are these sd-nodes ?

For the first factor, *b* should be more relevant to source document *D* if more sd-nodes are relevant to *b*. We capture this information by allowing *b* to receive relevance score from all the sd-nodes. For the second factor, *b* should be more relevant to *D* if more relevant *b* is to sd-nodes. We consider this information by evaluating the relevance between *b* and every sd-node (Section 3.1). For the last factor, important sd-nodes should have higher impact. We consider this information by evaluating the importance of sd-nodes and assigning higher initial value to importance ones (Section 3.3). We combine all factors in the global ranking process to select the top-n relevant background knowledge (Section3.2).

3.1 Relevance Evaluation between Nodes

In this section, we evaluate the semantic relevance between different nodes which is the weight of the edge between them. We introduce *Search Engine* as a resource, which has been proven effective in relevance evaluation (Gligorov et al., 2007). This method is motivated by the phenomenon that the number of results returned by search engine for query $p \cap q$indicates the relevance between p and q.

However, considering the different popularization of queries, this number alone can not accurately express their semantic relevance. For example, query $car \cap automobile$ gets $294,300,000$ results, whereas query $car \cap apple$ gets $683,000,000$, which is 2 times higher than the previous one. Obviously, $automobile$ is more relevant to car rather than $Apple$. The reason of this phenomenon is that $apple$ is far more popular than $automobile$, which increase its possibility of co-occurrence with car. So we consider the number of results for $p \cap q$ together with p and q with $WebJaccard\,Coefficient$ (Bollegala et al., 2007) to evaluate the relevance between p and q according to Formula 1, where $H(p)$, $H(q)$, and $H(p \cap q)$ indicate the number of results for query p, p, and $p \cap q$.

$$WebJaccard(p,q) = \begin{cases} 0 & \text{if } H(p \cap q) \leq C \\ \frac{H(p \cap q)}{H(p)+H(q)-H(p \cap q)} & \text{otherwise.} \end{cases} \quad (1)$$

To convert one "$argument_1, predicate, argument_2$" triple into query, we use $argument_1 \cap argument_2$ as the query for one triple. We have tried $argument_1 \cap predicate \cap argument_2$ which

is usually very sparse. Besides, the combination of two arguments usually maintain better semantic completeness of triple compared to other combinations according to our analysis. So this setup aims to balance completeness and sparseness. Accordingly, two triples are combined as $argument_1 \cap argument_2 \cap argument_1^{'} \cap argument_2^{'}$. Considering the scale and noise in the Web data, it is possible for two words to appear together accidentally. To reduce the adverse effects attributed to random co-occurrences, we set 0 to the $WebJaccard\ Coefficient$ of query $p \cap q$, if the number of result is less than a threshold C.

3.2 Iterative Relevance Propagation

Here we propose the relevance propagation based iterative process to evaluate the relevance between certain background knowledge and source document. Note that standard label propagation mainly focuses on classification task (Wang and Zhang, 2008). However, we focus on a ranking problem where the best ranking result is computed during an iterative process in this study. So we make two modifications to suit the ranking problem better: not reseting the relevance score and introducing the propagation between source document information during iteration.

Propagation possibility The edge between $node_i$ and $node_j$ is weighted by $r(i,j)$ to measure their relevance. However, $r(i,j)$ cannot completely present the propagation possibility because one node can be equally relevant to all of its neighbors. Thus, we define $p(i,j)$ based on $r(i,j)$ in formula 2 to indicate the propagation possibility between $node_i$ and $node_j$.

$$p(i,j) = \frac{r(i,j) \times \delta(i,j)}{\sum_{k \in N} r(k,j) \times \delta(k,j)} \tag{2}$$

N is the set of all nodes, $\delta(i,j)$ denotes whether an edge exists between $node_i$ and $node_l$ in the triple-graph or not, which indicates whether they may propagate to each other or not. E is the set of edges.

$$\delta(i,j) = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Iterative propagation There are $n \times n$ pairs of nodes, the $p(i,j)$ of them is stored in a matrix P. we use $\vec{W} = (w_1, w_2, \cdots, w_n)$ to denote the relevance score of all nodes, in which w_i indicates the relevance between $node_i$ and source document D. Here the $node_i$ can indicate both sd-nodes and bk-nodes because they are processed during one fellow step. So that we keep updating both sd-nodes and bk-nodes and do not distinguish them explicitly. The only difference between them is that we initialize the w_i of sd-nodes as its importance to D (Section 3.1) while bk-nodes as 0 at the beginning. We use matrix P together with $\delta(i,j)$ to compute the $\vec{W}$ during a iterative process, where $\vec{W}$ is updated to $\vec{W'}$ during the end of every iteration. The matrix $\vec{W'}$ is updated according to the following Formula 4:

$$\begin{aligned} \vec{W'} &= \vec{W} \times P \\ &= \vec{W} \times \begin{bmatrix} p(1,1) & p(1,2) & \cdots & p(1,n) \\ p(2,1) & p(2,2) & \cdots & p(2,n) \\ \cdots & \cdots & \cdots & \cdots \\ p(n,1) & p(n,2) & \cdots & p(n,n) \end{bmatrix} \end{aligned} \tag{4}$$

each w_i in $\vec{W}$ is updated to $w_i^{'}$ according to the formula 5, where w_i is propagated from all the other $w_j (j \neq i)$ according to their propagation possibility $p(j,i)$. We also introduce the propagation from bk-nodes to sd-nodes, where bk-nodes serve as intermediate to help mining latent semantics.

$$\begin{aligned} w_i^{'} &= w_1 \cdot p(1,i) + w_2 \cdot p(2,i) + \cdots + w_n \cdot p(n,i) \\ &= \sum_{k \in N} w_k \cdot p(k,i) \\ &= \sum_{k \in N} w_k \cdot \left(\frac{r(i,j) \times \delta(i,j)}{\sum_{k \in N} r(k,j) \times \delta(k,j)} \right) \end{aligned} \tag{5}$$

3.3 Importance Evaluation for sd-nodes

The main idea of our model is to propagate relevance score from sd-nodes to bk-nodes (Section 3.2). So the initialization of sd-node is important, which indicates the importance of different source document information. This section solves this problem by evaluating the importance of sd-nodes to source document. We use v_j to denote the initialization of sd-nodes, which indicates the importance of $node_j$ ($node_j \in$ set of sd-nodes) to source document. In this section, we propose a modified relevance propagation method to evaluate v_j for sd-notes. We first construct a triple-graph consisting of sd-nodes only. Then we initialize the relevance score of sd-nodes according to a simple approach based on text frequency (Kohlschütter et al., 2010). We use similar relevance propagation process without resetting the relevance score at the beginning of every iteration, until a global stable state is achieved. Finally, we normalize all the relevance scores to get $\vec{V}$, which indicates the importance of sd-nodes to the source document. We return $\vec{V}$ to the global ranking model (Section 3.2) as part of the input. The initial importance of bk-nodes is set as 0 at the beginning, which denotes that all bk-nodes are ir-relevant to source document before the starting of global ranking process.

4 Experiment

We treat our task as a ranking problem, which takes a document as input and output the ranked list of background knowledge. We evaluate our method as a ranking problem similarly to information retrieval task and focus on the performances of models with different setups.

4.1 Data Preparation

The experiment data consists of two parts: source document information and corresponding background knowledge. To select source documents, we use the *ACE corpus* (Doddington et al., 2004) for 2005 evaluation [4] which consists of 599 articles from multiple sources. We use ReVerb to extract these documents into multi-triples. For background knowledge, we first retrieve relevant web pages with simply term matching method and then extract these pages with ReVerb into a set of triples serving as background knowledge. To ensure the quality, we filter them according to the confidence given by ReVerb.

Besides automatic extraction, we also adapt our system to the golden annotation of ACE as source document information and standard YAGO knowledge base [5] as background knowledge (Hoffart et al., 2013). We compare its performance with that in fully automatic system and evaluate the effect of automatic extraction. For better comparison with YAGO, we retrieve relevant pages from WikiPedia although our automatic extraction method is applicable to any corpus resources.

For every outputted list, three trained annotators check the result and decide which background knowledge is relevant to source document. They work separately and check the same list, so that we can evaluate their annotation consistence. They totally annotated more than 7000 background knowledge and achieved a $Fleiss'\ Kappa$ value of 0.8066 in best situation and 0.7076 in average between three annotators, which is a good consistence between multi-annotator (Fleiss, 1971). When collision happened, we choose the label selected by more annotators.

4.2 Baseline system

Although we treat our task as a ranking problem, it is difficult to apply corresponding methods in traditional ranking tasks such as information retrieval (IR) (Manning et al., 2008) and entity linking (EL) (Han et al., 2011; Kataria et al., 2011; Sen, 2012) directly in our task. First, both *IR* and *EL* make use of the link structure between web or Wiki pages. However, our task takes single document as input and no link exists between documents which makes it difficult to apply *IR* and *EL* methods such as page rank (Page et al., 1999) and collective method (Han et al., 2011; Sen, 2012) in this task directly. Second, *EL* usually evaluate the text similarity between certain document and target page in WikiPedia. However, our task focuses on the ranking of "$argument_1, predicate, argument_2$" triple, which contains little text information. Lack of text information also limits the application of corresponding methods in our task.

[4] http://catalog.ldc.upenn.edu/LDC2006T06

[5] http://www.mpi-inf.mpg.de/yago-naga/yago

Setup	MAP	P&20
Baseline	0.494	0.377
AutoSD + AutoBK + NoInitial	0.504	0.378
AutoSD + AutoBK + WithInitial	0.531	0.406
GoldSD + AutoBK + NoInitial	0.564	**0.417**
GoldSD + AutoBK + WithInitial	0.553	0.406
GoldSD + YAGO + NoInitial	0.676	0.328
GoldSD + YAGO + WithInitial	**0.676**	0.328

Table 1: The result of our model in different setups: *GoldSD* indicates using annotation of ACE corpus as source document information; *YAGO* indicates using YAGO knowledge base as background knowledge; *AutoSD* and *AutoBK* means aotomatic extraction to source document and background knowledge; *NoInitial* and *WithInitial* means whether using different initial importance to source document information.

For better comparison, we introduce search engine as resource which is proved effective in relevance evaluation (Gligorov et al., 2007) and propose a search engine based strong baseline. As illustrated before, the relevance R_i between background knowledge b_i and source document D has been converted into the relevance between b_i and the triples of D. Hence, we compute R_i by accumulating all r_{ij}, the relevance scores between b_i and every sd-node s_j with the same method in Section 3.1 ($R_i = \sum_{s_j \in S} r_{ij}$, S is the set of sd-nodes). Then we rank all background knowledge according to the value of R_i and output the ranked list as final result. We extract source document and background knowledge automatically in the baseline system, which makes it applicable in different setups.

4.3 Experiment setup

We evaluate our model in different setups. First, we extract both source document information and background knowledge automatically. Second, we use golden annotation of ACE as source document information but extract background knowledge automatically. Third, we use golden annotation of ACE and introduce standard YAGO as background knowledge. For all of them three, we evaluate the different performances with and without initial importance of sd-nodes(Section 3.3). We evaluate the performance with two famous criteria in ranking problem: MAP (Voorhees et al., 2005) requires more accuracy and focuses on the knowledge in higher position; $P\&N$ which require more coverage and pays more attention to the number of relevant ones in $Top\ N$ knowledge. Note that we do not evaluate the $Recall$ performance because there can be millions of background knowledge to be ranked for every document. It is impossible to check all of them. So we focus on the $Top\ N$ candidates and evaluate the performance with MAP and $P\&N$. In this study, we evaluate the $Top\ 20$ background knowledge triples which are most easily to be viewed by users.

4.4 Experiment Result

The performance of our model is shown in Table 1. Our search engine based baseline system achieve a rather good performance: a *MAP* value of 0.494 and 0.377 in *P&20*. At the same time, our model outperforms the baseline system in almost every setup and evaluation criterion. The best performance of *MAP* is achieved by *GoldSD+YAGO* (0.676), while the best performance of *P&20* is achieved by *GoldSD+AutoBK* (0.417). To analyze the result further, we find that the initial importance, automatic extraction to source document, and to background knowledge have different effect on the final performance.

4.4.1 Effect of automatic extraction to source document

We use ACE corpus as source documents, which contain golden annotation to document information. So we can evaluate the effect of automatic extraction to source document by comparing the performance with and without golden annotation. The performance without golden annotation is shown in *AutoSD+AutoBK* of Table 1, while the other one shown in *GoldSD+AutoBK*. We can find that the performance of *GoldSD+AutoBK* is better than that of *AutoSD+AutoBK* in both *MAP* and *P&20*, which indicates that golden annotation do help to improve the ranking result.

We further analyze the result and find an interesting phenomenon: these two systems performs greatly different with the setup of *NoInitial*, but equally with the setup of *WithInitial*, which indicates that the performance of *AutoSD+AutoBK* has been improved by evaluating the importance of source document information (Section 3.3). So we can naturally infer that, with a better importance evaluating method in *AutoSD+AutoBK*, we may achieve similar performance compared to that in golden annotation. Note that, *AutoSD+AutoBK* is compatible with any corpus which is more useful in real applications.

4.4.2 Effect of automatic extraction to background knowledge

We evaluate the effect of automatic extraction to background knowledge by comparing the performances between *GoldSD+AutoBK* and *GoldSD+YAGO*. In *GoldSD+AutoBK*, the background knowledge is extracted automatically with ReVerb, which has greater coverage but less accuracy. In contrast, the *GoldSD+YAGO* make use of YAGO as background knowledge, which is less coverage but better accuracy. This difference are reflected on the system performance, where *GoldSD+YAGO* achieves much better result in *MAP*, but much worse in *P&20*. This is partly because that *MAP* focus on the background knowledge in higher position which requires more accuracy, while *P&20* pays more attention to the number of relevant background knowledge which require more coverage.

In general, automatic extraction system has better coverage but less accuracy compared to YAGO based system. However, automatic extraction to background knowledge may help in real applications by improving coverage greatly. Besides, the loss of accuracy is partly due to the technology of information extraction which may be improved in the future. In addition, we can also combine these two ways to acquire background knowledge to balance coverage and accuracy in the future.

4.4.3 Effect of initial importance to source document information

Initial importance to source document information (Section 3.3) is important to the performance of our models as shown in Table 1. The model *AutoSD+AutoBK+WithInitial* outperforms the *AutoSD+AutoBK+NoInitial* compared to other setups, which indicates the help of initial importance to the ranking result. Especially, initial importance to source document information helps most in the setup of *AutoSD+AutoBK*, which is most useful in real applications. So we can naturally infer that, by proposing better importance evaluating method, we may further improve the performance of *AutoSD+AutoBK+WithInitial*, which will great helpful in the future application of this method.

5 Related Work

Document enrichment focuses on introducing external knowledge into source document. There are mainly two kinds of works in this topic according to the resource they relying on. The first line of works make use of *WikiPedia* and enrich source document by linking the entity to its corresponding Wiki page (Bunescu and Pasca, 2006; Cucerzan, 2007). In early stage, most researches rely on the similarity between the context of the mention and the definition of candidate entities by proposing different measuring criteria such as dot product, cosine similarity, KL divergence, Jaccard distance and more complicated ones (Bunescu and Pasca, 2006; Cucerzan, 2007; Zheng et al., 2010; Hoffart et al., 2011; Zhang et al., 2011). However, these methods mainly rely on text similarity but neglect the internal structure between mentions. So another kind of works explore the structure information with collective disambiguation (Kulkarni et al., 2009; Kataria et al., 2011; Sen, 2012; He et al., 2013). These methods make use of structure information within context and resolve different mentions based on the coherence among decisions. Despite the success, the entity linking methods rely on WikiPedia which has great coverage but less accuracy.

Another line of works try to improve the accuracy of enrichment by introducing ontologies (Motta et al., 2000; Passant, 2007; Fodeh et al., 2011; Kumar and Salim, 2012) and structured knowledge such as WordNet (Nastase et al., 2010) and Mesh (Wang and Lim, 2008). In these studies, resources usually provides word or phrase semantic information such as synonym (Sun et al., 2011) and antonym (Sansonnet and Bouchet, 2010). However, these methods rely on special ontologies constructed with supervision or even manually, which is difficult to expand and in turn limits the application of them.

6 Conclusion and Future Work

This study presents a triple based background knowledge ranking model to acquire most relevant background knowledge to certain source document. We first develop a triple graph based document presentation to combine source document together with the background knowledge. Then we propose a global iterative ranking model to acquire $Top\ n$ relevant knowledge, which provide additional information beyond the source document. Note that, both source document information and background knowledge are extracted automatically which is useful in real application. The experiments show that our model achieves better results over a strong baseline, which indicates the effectiveness of our framework.

Another interesting phenomenon is that *YAGO* based enrichment model achieved better ranking accuracy, but less coverage compared to automatic extraction model. To combine these two sources of background knowledge may help to overcome both coverage and accuracy problem. So exploiting proper way to incorporate knowledge base and automatic extraction is an important topic in our future work.

Finally, we believe that this background knowledge based document enriching technology may help in those semantic based NLP applications such as coherence evaluation, coreference resolution and question answering. In our future work, we will explore how to make use of these background knowledge in real applications, hopefully to improve the performance significantly in the future.

Acknowledgements

We thank Muyun Yang and Jianhui Ji for their great help. This work was supported by National Natural Science Foundation of China(NSFC) via grant 61133012, the National 863 Leading Technology Research Project via grant 2012AA011102 and the National Natural Science Foundation of China Surface Project via grant 61273321.

References

Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJCAI*, volume 7, pages 2670–2676.

Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2007. Measuring semantic similarity between words using web search engines. *www*, 7:757–766.

Volha Bryl, Claudio Giuliano, Luciano Serafini, and Kateryna Tymoshenko. 2010. Using background knowledge to support coreference resolution. In *ECAI*, volume 10, pages 759–764.

Razvan C Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL*, volume 6, pages 9–16.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*, volume 7, pages 708–716.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *LREC*. Citeseer.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. 2008. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open information extraction: The second generation. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume One*, pages 3–10. AAAI Press.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Samah Fodeh, Bill Punch, and Pang-Ning Tan. 2011. On ontology-driven document clustering using core semantic features. *Knowledge and information systems*, 28(2):395–421.

Risto Gligorov, Warner ten Kate, Zharko Aleksovski, and Frank van Harmelen. 2007. Using google distance to weight approximate ontology matches. In *Proceedings of the 16th international conference on World Wide Web*, pages 767–776. ACM.

Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013. Learning entity representation for entity disambiguation. *Proc. ACL2013*.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. Yago2: a spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61.

Xiaohua Hu, Xiaodan Zhang, Caimei Lu, Eun K Park, and Xiaohua Zhou. 2009. Exploiting wikipedia as external knowledge for document clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 389–396. ACM.

Saurabh S Kataria, Krishnan S Kumar, Rajeev R Rastogi, Prithviraj Sen, and Srinivasan H Sengamedu. 2011. Entity disambiguation with hierarchical topic models. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1037–1045. ACM.

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 441–450. ACM.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM.

Yogan Jaya Kumar and Naomie Salim. 2012. Automatic multi document summarization approaches. *Journal of Computer Science*, 8(1).

Thomas Lin, Oren Etzioni, et al. 2010. Identifying functional relations in web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1266–1276. Association for Computational Linguistics.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge.

Enrico Motta, Simon Buckingham Shum, and John Domingue. 2000. Ontology-driven document enrichment: principles, tools and applications. *International Journal of Human-Computer Studies*, 52(6):1071–1109.

Vivi Nastase, Michael Strube, Benjamin Börschinger, Cäcilia Zirn, and Anas Elghafari. 2010. Wikinet: A very large scale multi-lingual concept network. In *LREC*.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web.

Patrick Pantel and Ariel Fuxman. 2011. Jigs and lures: Associating web queries with structured entities. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 83–92. Association for Computational Linguistics.

Alexandre Passant. 2007. Using ontologies to strengthen folksonomies and enrich information retrieval in weblogs. In *Proceedings of International Conference on Weblogs and Social Media*.

Alan Ritter, Oren Etzioni, et al. 2010. A latent dirichlet allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 424–434. Association for Computational Linguistics.

Jean-Paul Sansonnet and François Bouchet. 2010. Extraction of agent psychological behaviors from glosses of wordnet personality adjectives. In *Proc. of the 8th European Workshop on Multi-Agent Systems (EUMAS10)*.

Stefan Schoenmackers, Oren Etzioni, Daniel S Weld, and Jesse Davis. 2010. Learning first-order horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1088–1098. Association for Computational Linguistics.

Prithviraj Sen. 2012. Collective context-aware topic models for entity disambiguation. In *Proceedings of the 21st international conference on World Wide Web*, pages 729–738. ACM.

Koun-Tem Sun, Yueh-Min Huang, and Ming-Chi Liu. 2011. A wordnet-based near-synonyms and similar-looking word learning system. *Educational Technology & Society*, 14(1):121–134.

Ellen M Voorhees, Donna K Harman, et al. 2005. *TREC: Experiment and evaluation in information retrieval*, volume 63. MIT press Cambridge.

Xudong Wang and Azman O Lim. 2008. Ieee 802.11 s wireless mesh networks: Framework and challenges. *Ad Hoc Networks*, 6(6):970–984.

Fei Wang and Changshui Zhang. 2008. Label propagation through linear neighborhoods. *Knowledge and Data Engineering, IEEE Transactions on*, 20(1):55–67.

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

Wei Zhang, Yan Chuan Sim, Jian Su, and Chew Lim Tan. 2011. Entity linking with effective acronym expansion, instance selection and topic modeling. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 1909–1914. AAAI Press.

Zhicheng Zheng, Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010. Learning to link entities with knowledge base. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 483–491. Association for Computational Linguistics.

Jun Zhu, Zaiqing Nie, Xiaojiang Liu, Bo Zhang, and Ji-Rong Wen. 2009. Statsnowball: a statistical approach to extracting entity relationships. In *Proceedings of the 18th international conference on World wide web*, pages 101–110. ACM.

Towards an open-domain conversational system fully based on natural language processing

Ryuichiro Higashinaka[1], **Kenji Imamura**[1], **Toyomi Meguro**[2], **Chiaki Miyazaki**[1]
Nozomi Kobayashi[1], **Hiroaki Sugiyama**[2], **Toru Hirano**[1]
Toshiro Makino[1], **Yoshihiro Matsuo**[1]
[1]NTT Media Intelligence Laboratories
[2]NTT Communication Science Laboratories
{higashinaka.ryuichiro, imamura.kenji, meguro.toyomi, miyazaki.chiaki,
kobayashi.nozomi, sugiyama.hiroaki, hirano.tohru,
makino.toshiro, matsuo.yoshihiro}@lab.ntt.co.jp

Abstract

This paper proposes an architecture for an open-domain conversational system and evaluates an implemented system. The proposed architecture is fully composed of modules based on natural language processing techniques. Experimental results using human subjects show that our architecture achieves significantly better naturalness than a retrieval-based baseline and that its naturalness is close to that of a rule-based system using 149K hand-crafted rules.

1 Introduction

Although task-oriented dialogue systems have been extensively researched over the decades (Walker et al., 2001; Williams et al., 2013), it is only recently that non-task-oriented dialogue, open-domain conversation, or chat has been attracting attention for its social and entertainment aspects (Bickmore and Picard, 2005; Ritter et al., 2011; Bessho et al., 2012). Creating an open-domain conversational system is a challenging problem. In task-oriented dialogue systems, it is possible to prepare knowledge for a domain and create understanding and generation modules for that domain (Nakano et al., 2000). However, for open-domain conversation, such preparation cannot be performed. Since it is difficult to handle users' open-domain utterances, to create workable systems, conventional approaches have used hand-crafted rules (Wallace, 2004). Although elaborate rules may work well, the problem with the rule-based approach is the high cost and the dependence on individual skills of developers, which hinders systematic development. Another problem with the rule-based approach is its low coverage; that is, the inability to handle unexpected utterances.

The recent increase of web data has propelled the development of approaches that use data retrieved from the web for open-domain conversation (Shibata et al., 2009; Ritter et al., 2011). The merit of such retrieval-based approaches is that, owing to the diversity of the web, systems can retrieve at least some responses for user input, which solves the coverage problem. However, this comes at the cost of utterance quality. Since the web, especially Twitter, is inherently noisy, it is, in many cases, difficult to sift out appropriate sentences from retrieval results.

In this paper, we propose an architecture for an open-domain conversational system. The proposed architecture is fully composed of modules based on natural language processing (NLP) techniques. Our stance is not just to hand-craft or to search the web for utterances, but to create a system that can fully understand and generate utterances. We want to show that it is possible to build an open-domain conversational system by combining NLP modules, which will open the way to a systematic development and improvement. We describe our open-domain conversational system based on our architecture and present results of an evaluation of its performance by human subjects. We compare our system with rule-based and retrieval-based systems, and show that our architecture is a promising direction. In this work, we regard the term open-domain conversation to be interchangeable with non-task-oriented dialogue, casual conversation (Eggins and Slade, 2005), chat, or social dialogue (Bickmore and Cassell, 2000). We use the term to denote that user input is not restricted in any way as in open-domain question answering

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 928–939, Dublin, Ireland, August 23-29 2014.

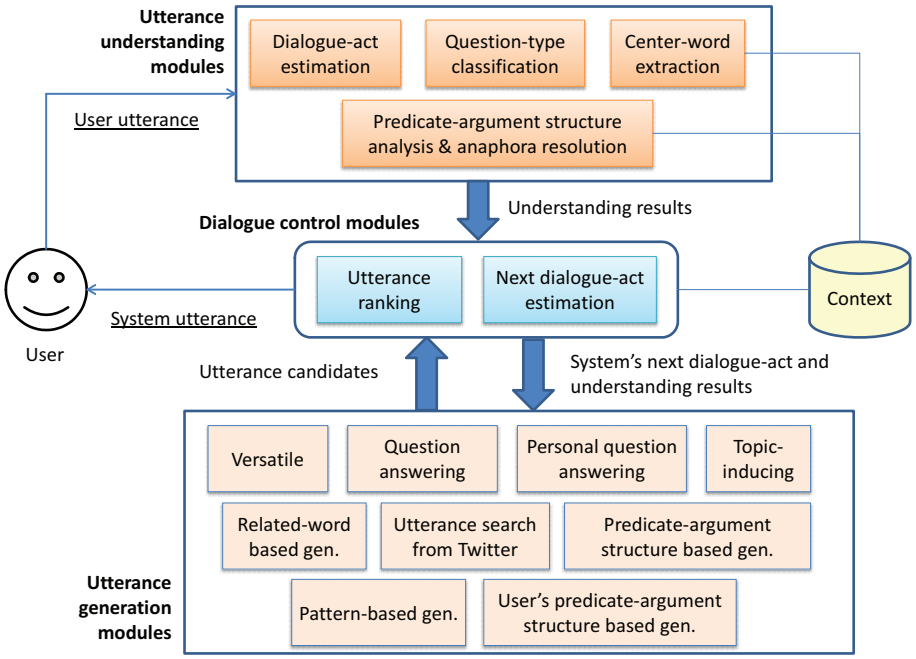


Figure 1: System architecture.

(QA) (Voorhees and Tice, 2000) and open information extraction (Etzioni et al., 2008). The application here in mind is one that can chat with users like chatbots. It should also be noted that we deal with Japanese text chat in this paper, although we believe the architecture to be largely language-independent and extendable with other modalities.

In Section 2, we describe the architecture and its underlying modules. In Section 3, we describe the rule-based and retrieval-based systems that we use for comparison. In Section 4, we describe the experiment we performed to evaluate our system. Section 5 summarizes the paper.

2 Architecture and System Description

Figure 1 shows the architecture we propose for an open-domain conversational system. The architecture has three main components: utterance understanding, dialogue control, and utterance generation. Following the literature on discourse theory (Grosz and Sidner, 1986), we regard *intention* (intentional structure), *topic* (attention state), and *content* (linguistic structure) as three important elements in conversation, and seek to create a system that can understand and generate on the basis of them in a general way. The dialogue control component works by ranking utterance candidates using a general coherence criterion (Hovy, 1991). Note that the overall architecture is roughly the same as conventional dialogue systems; however, the internal architecture is different so as to allow open-domain conversation. To give a rough idea of how the system works, Figure 2 shows an example dialogue between our system and a user (one of the subjects in our experiment). As this example shows, the system can handle various user utterances. Below, we describe how this is achieved.

2.1 Utterance Understanding Modules

We identify dialogue-act, question-type, center-word, and predicate-argument structure (PAS). Dialogue-act and question-type correspond to intention, center-word to topic, and PAS to content. We use PASs because they can represent an arbitrary sentence. For languages other than Japanese, instead of PASs, semantic role labeling (SRL) can be used (Palmer et al., 2010). Below, we describe each module.

Dialogue-act estimation: As a dialogue-act tag set, we use the one proposed by Meguro et al. (2013). Although their tag set is designed for annotating listening-oriented dialogue (LoD), since speakers in LoD are allowed to speak freely, the tag set can cover diverse utterances, making it suitable for

open-domain conversation. There are 33 dialogue-acts in the tag set. See (Meguro et al., 2013) for details. We used 1259 LoDs annotated with dialogue-acts and trained a classifier using a support vector machine (SVM). The features used are word N-grams, semantic categories (obtained from a Japanese thesaurus Goi-Taikei (Ikehara et al., 1997)), and character N-grams. Here, unless otherwise noted, we use JTAG (Fuchi and Takagi, 1998) for morphological analysis in this work. When we use the LoD data for training and testing, by a ten-fold cross validation, the estimation accuracy is 45%, which is reasonable when considering that the inter-annotator agreement rate is 59%. For reference, the majority baseline, which estimates the dialogue-acts of all utterances to be information-provision, has 12% accuracy.

Question-type classification: We use the question taxonomy by Nagata et al. (2006) because it was derived by analyzing questions from the general public and therefore covers diverse questions. The taxonomy has 23 question types under five main categories: name, quantity, explanation, yes-no, and other. Since some types could be too specific, by merging similar ones, we shrinked the 23 types into 13: name-other, name-person, quantity-other, quantity-date, quantity-period, quantity-money, yes-no, explanation-reason, explanation-definition, explanation-method, explanation-reputation, explanation-association, and other. Using an in-house data set of about 48K questions annotated with the 13 types, we trained a logistic-regression-based classifier that achieves a classification accuracy of 92.5% by a five-fold cross validation. The majority baseline that always classifies to name-other has 39.5% accuracy.

Center-word extraction: We define a center-word as a noun phrase (NP) that denotes the topic of a conversation. We hypothesize that an utterance has at most one NP suitable for a center-word. To extract an NP from an utterance, we use conditional random fields (Lafferty et al., 2001); NPs are extracted directly from a sequence of words without creating a parse tree. For the training and testing, we prepared 10K sentences with center-word annotation. Here, the sentences were those randomly sampled from the open-domain conversation corpus (See Section 2.2). The feature template uses words, part-of-speech (POS) tags, and semantic categories of current and neighboring words. The extraction accuracy is 83.4% by a five-fold cross validation. This module has access to the context. When there are already center-words (represented by a stack) in the context, the center-word of the current utterance is placed at the top of the stack.

PAS analysis and anaphora resolution: In PAS analysis, predicates and their arguments are detected. A predicate can be a verb, adjective, or copular verb, and the arguments are NPs associated with cases in case grammar. As cases, we use standard cases `ga` (nominative), `wo` (accusative), `ni` (dative) as well as `de` (locative/instrumental), `to` (with), `kara` (source), `made` (goal). We use the PAS analyzer described in (Imamura et al., 2009; Imamura et al., 2014a; Imamura et al., 2014b). The analyzer works statistically by ranking NPs in the context using supervised learning with an obligatory case information dictionary and a large-scale word dependency language model. For the ranking, NPs in the current utterance and the center-words in the context are considered. Also considered are the special NPs 'I', 'YOU', 'exog (exophora general)', and 'null'. Here, 'exog' means that there should be some argument for a case but it is missing in the context, and 'null' means that no argument is required for a case. When arguments are filled with NPs from the context, this means that zero pronoun resolution (a type of anaphora resolution) has been performed. The analyzer is reported to have an accuracy of 57-62% for `ga`, `wo`, and `ni` cases for dialogue data (Imamura et al., 2014b). When a zero pronoun is resolved, the NP used for the resolution is placed at the top of the center-word stack because such an elided NP is grounded among the conversational participants and is likely to be the focus of conversation (Walker et al., 1994).

2.2 Dialogue Control Modules

We have two modules for dialogue control. The next dialogue-act estimation module roughly decides the next user utterance on the intention level (dialogue-act level). The utterance ranking module ranks utterance candidates (See Section 2.3 for how utterance candidates are created) by a coherence criterion.

	Utterance (English translation by the authors)	DA	Gen. Module
SYS1	Doumo desu (Hi)	Greeting	Initial prompt
USR1	Doumo. Genki desuka? (Hi. How are you?)	Greeting	
SYS2	Saikin no oishii mono ni tsuite kikasete kudasai (Tell me about delicious food that you've had recently)	Q-Plan	Topic-Inducing
USR2	Karei ni hamatterunda! (I'm now into curry!)	S-Disc Fact	
SYS3	Aishou ga ii desuka (Does it go together well with you?)	Confirmation	PAS
USR3	Toku ni tai karei wa aishou ga yoimitai (Thai curry especially goes together well with me)	S-Disc Pref. Pos.	
SYS4	Guzai ga gorogoro shitete, natsukashii karei ga sukidesu (I like nostalgic curry with large chunks of ingredients)	S-Disc Pref. Pos.	Twitter
USR4	Yokosuka no karei nanka, soudayone (Curry in Yokosuka is like that, isn't it?)	Information	
SYS5	Watashi mo daisuki desu! (I like it very much!)	S-Disc Pref. Pos.	Versatile
USR5	Yokosuka, ikimasuka? (Do you go to Yokosuka?)	Q-Fact	
SYS6	Yokosuka sunderunde, nanka ureshii desu (I live in Yokosuka, so I'm flattered)	S-Disc Fact	Twitter

Figure 2: Example dialogue with our proposed system [Prop (tri); See Section 4.1]. DA denotes recognized dialogue-acts for user utterances and those chosen by the system for system utterances. Gen. Module indicates the generation module used. Underlined words denote newly recognized center-words.

Next dialogue-act estimation: Using the same approach as Sugiyama et al. (2013), this module takes into account three previous dialogue-acts in the context to estimate the system's next dialogue-act. We trained an SVM-based dialogue-act estimator using 1259 LoDs. Using 1,000 dialogues as training data and 259 as test data, the trained estimator can predict the next dialogue-act with the accuracy of 28% (NB. majority baseline has 15% accuracy). Although the accuracy is low, since the task is subjective and there is no definite answer, we consider the estimator to have sufficient ability to choose a reasonable next dialogue-act.

Utterance ranking: We adopt coherence as a general criterion for ranking utterances because it is a well-recognized measure of discourse and can be applied to arbitrary sentences (Lapata, 2003; Barzilay and Lapata, 2008). We hypothesize that an utterance that is the most cohesive to the current context should be chosen for the output of the system. To create the ranker, we first collected a data set of 3,680 open-domain conversations (hereafter, open-domain conversation corpus; 134K utterances) between humans, and from 3,496 of them (184 were held out for development), created dialogue snippets (excerpts) by taking N consecutive utterances. We use these dialogue snippets as references (positive examples). We also create counter-references (pseudo negative examples) by swapping the last utterance of each snippet with a randomly selected one from the dialogue from which the snippet was taken. This is similar to how Barzilay and Lapata (2008) created their training data for their coherence models. We then train a ranker by ranking SVM (Joachims, 2002) in the same manner as (Higashinaka and Isozaki, 2008). The ranker is trained so that references are ranked higher than counter-references. Following Lapata (2003), who used pairs of words for sentence ordering, we use, as features, the pairs of words, POS-tags, and semantic categories between the last utterance and each of the previous utterances. For example, when the last utterance U_l has k words and one of its previous utterance U_p has m words, we create $k \times m$ features by combining them. This feature generation is done also for POS-tags and semantic categories and is iterated over all previous utterances. We trained two rankers using 2 and 3 for N. When N is 2, we have 124,213 snippets. When N is 3, we have 120,717. By using four-fifths of the data for training and using the remaining one-fifth for testing, the rankers achieve 66.7% and 66.4% accuracies for N=2 and N=3, respectively. Here, the random baseline's accuracy is 50%. We only use 2 and 3 for N here since a larger N could lead to the explosion of features. By default, we use the ranker trained with N=3. The trained ranker ranks utterance candidates (generated by the modules in Section 2.3) and outputs the top one as a system utterance.

2.3 Utterance Generation Modules

We prepared nine modules for generation. The versatile, QA, and personal QA modules generate on the basis of dialogue-acts and question-types (intention). The topic-inducing, related-word, Twitter, and

PAS modules generate on the basis of center-words (topic). The pattern and user PAS modules generate by using the surface string and PASs of user utterances (content). Note that, for all modules, the system's next dialogue-act is taken into account; that is, wherever necessary, the aforementioned dialogue-act estimation module is applied to generated utterances so that utterances whose estimated dialogue-acts match the system's next dialogue-act are returned.

Versatile: This module receives the system's next dialogue-act and returns utterances randomly chosen from the list of utterances for that dialogue-act. To create lists for dialogue-acts, we first extracted frequent utterances for each dialogue-act in the LoD corpus. Then, we selected context-independent utterances for the dialogue-act. We call such utterances "versatile utterances" because they can be used in various situations. For example, we have "I like it", "It is good", and "That's great" for S-Disc Pref. Pos. (a dialogue-act that discloses one's positive preference).

QA: When the user dialogue-act is a question and the question-type requires a named entity as an answer (i.e., when the question-type starts with a 'name' or 'quantity'), we call an off-the-shelf QA API that is publicly available[1]. The API returns top-N answers (NEs) for a natural language query (Uchida et al., 2013; Higashinaka et al., 2013). We refer to this API with the user input sentence as a query and obtain the top-five answers. This module returns these answers as utterance candidates.

Personal QA: When the user dialogue-act is a question, this module is called for answering personal questions. Answering such questions is important in chat (Batacharia et al., 1999) or even in task-oriented dialogue (Takeuchi et al., 2007). We use the same method as (Sugiyama et al., 2014b; Sugiyama et al., 2014a) and create a person database (PDB) of question-answer pairs for a persona. In the PDB, the questions are given category labels (e.g., favorite sport, whether the persona likes dogs, etc.) as well as question-types based on our taxonomy. Given a question, the answer is obtained by searching the PDB by the category label and the question-type for the question. To obtain the category label, a separately-trained logistic-regression-based classifier is used. We prepared a PDB for a persona 'Aiko' (a 29 year-old Japanese woman). The PDB contains 4,428 question-answer pairs. This module searches Aiko's PDB and returns obtained answers as utterance candidates.

Topic-inducing: When there is no center-word, this module returns utterances that introduce topics (e.g., "Let's talk about favorite foods!"). The utterances are chosen randomly from a list of utterances that we extracted from the dialogue-initiating utterances in the LoDs.

Related-word: The input to this module is the top center-word (C) and the next dialogue-act. For given C, we first get its related-words. Although we cannot describe details for lack of space, as related-words, we have attributes, question words, associative words, and category words. Such words are mined from blogs, Twitter, and Wikipedia by using lexico-syntactic patterns (Hearst, 1992). By combining related-words with a small number of templates, utterances are created. For example, we have a template "C wa ADJ desune (C is ADJ)" where ADJ is an adjectival attribute of C. The created utterances are returned as utterance candidates. This approach is similar to that used by (Higuchi et al., 2008) and (Sugiyama et al., 2013) in that words obtained from large text data are combined with templates for generation.

Twitter: We use the same approach as (Higashinaka et al., 2014), who created a database of Twitter sentences by word-level and syntactic-level filtering. The database is searched by a query expanded with its related-words so that tweets relevant to the query can be accurately retrieved. It has been reported that only 6% of the retrieved results are judged as inappropriate by subjective evaluation. Using the same database and method as (Higashinaka et al., 2014), this module returns the top-ten retrieved sentences from the database using the top center-word as a query. The database contains about 7M sentences.

[1] https://dev.smt.docomo.ne.jp/?p=docs.api.page&api_docs_id=6

PAS: We created a database of PASs by processing more than three years' of blogs. For fear of noise, we only harvested PASs that have just a predicate and an argument for `ga` (nominative) with its topic (an NP) explicitly marked by a topic marker `wa`. From the blogs, we obtained 146K PASs for 50K topics. Given the top center-word and the next dialogue-act, this module looks for PASs whose topic matches the top center-word. Then, it converts the PASs into sentences so that they can convey the intention of the system's next dialogue-act. This conversion is automatic: we first convert the PASs into declarative sentences using a simple rule. Then, their sentence-end expressions (NB. In Japanese, modalities are mostly expressed by sentence-end expressions) are swapped with those matching the target dialogue-act. The sentence-end expressions used here are those automatically mined from dialogue-act annotated dialogue data. This module returns the converted sentences.

Pattern: In everyday conversation, there are typical exchanges of utterances like adjacency pairs (Schegloff and Sacks, 1973). To obtain such exchanges, we mined Twitter. We first collected about 919M tweets. Then, by extracting tweets connected with an in-reply-to relationship, we created a Twitter conversation corpus (20M conversations containing 90M tweets). By taking two consecutive tweets in the corpus and retaining only the frequent ones by a cut-off threshold of ten occurrences, we obtained 22K utterance pairs. The input to this module is the user utterance string, and the module outputs utterances from matched utterance pairs.

User PAS: This module uses the PASs of the user utterance and the next dialogue-act. It performs the same operation as the PAS-based generation and returns the converted sentences. The merit of this module is that the system can use the user's content in its utterance, which has been found to be useful in casual conversation for showing understanding (Ivey et al., 2013) and entraining with users (Nenkova et al., 2008).

3 Rule-based and Retrieval-based Systems

For comparison, we prepared a rule-based system and a retrieval-based one. Since there is no off-the-shelf rule-based system in Japanese, we created one on our own. Because we wanted to compare our system with a state-of-the-art rule-based system, we put a great deal of effort in its development. Remember that creating rules is still the standard way of creating an open-domain conversational system. Last year's Loebner Prize (a chatbot contest) winner, Mitsuku, was based on rules written in artificial intelligence markup language (AIML) (Wallace, 2004). Beating a reasonably created rule-based system is a hard task. As for the retrieval-based system, we use a large Twitter database to replicate the current best retrieval-based system, which we regard as our baseline.

3.1 Rule-based System

We had a seasoned engineer, who specializes in text analysis, create rules in AIML for two and a half months. We used only one person to ensure consistency in the rules. He first created some initial rules by referring to the AIML rules of A.L.I.C.E. (Wallace, 2004). The rules here include common exchanges of utterances (e.g., compliments of the season) as well as those to evade weak topics, changes of topics, and innocuous utterances. Then, he used the open-domain conversation corpus (with 10% held out for evaluation as we describe later) to extract two consecutive utterances and coded them as rules, making the first utterance the matching pattern and the second utterance the output sentence (called a template in AIML). This process involved replacing certain words with asterisks (wildcard) to widen the coverage of patterns and modifying templates if necessary. To augment the rules, he also used the topic-based utterance corpus that we created. The corpus contains 25K utterances conceived by 20 individuals for 250 topic words. The instruction given to the individuals was to come up with an interesting utterance that you think you would like to hear for a given topic word. Here, the topic words are frequent NPs in the open-domain conversation corpus. The pairs of a topic word and its utterance were coded as rules. To enable the system to answer personal questions, the Aiko PDB was employed. Since the PDB comprises a set of question-answer pairs, the PDB was straightforwardly incorporated into AIML with minor modifications. The rules were loaded by ProgramD, an AIML interpreter (http://aitools.org/Program_D), and were tested

	Utterance (English translation by the authors)
SYS1	Doumo desu. (Hi)
USR1	Konnichiwa (Hello)
SYS2	Konnnichiwa. Ryokou wa suki desuka? (Hello. Do you like traveling?)
USR2	Hai, sukidesu. (Yes, I do)
SYS3	Kokunai ryokou ga ooi desuka, kaigai ryokou ga ooi desuka? (Do you travel in Japan or go overseas?)
USR3	Moppara kokunai desu nee. (Mainly in Japan)
SYS4	Watashi wa moppara kokunai ryokou desu. (I travel mainly in Japan)
USR4	Onaji desune (It's the same)
SYS5	Soudesu yone (It is indeed)
USR5	Hai (Yes)
SYS6	Shuugaku ryokou wa doko deshitaka? (Where did you go for the school trip?)
USR6	Nagano ken deshita. (We went to Nagano)
SYS7	Nagano no kikou, natsu wa hontoni iidesu yone. (The climate in Nagano, it is really great in summer)
USR7	A, eeto, shuugaku ryokou wa nagano no yukiyama deshita. (Well, actually, the school trip was in the snowy mountains in Nagano)

Figure 3: Example dialogue with the rule-based system.

	Utterance (English translation by the authors)
SYS1	Doumoo (Hi)
USR1	Konnnichiwa (Hello)
SYS2	Chuugaku jidai no uiuishii midiamuhea ga natsukashii (It brings back the memories about the medium hair in junior high)
USR2	Nani wo totsuzen, doushitan desuka! (Where did that come from? Are you OK?)
SYS3	Mou, kimochi iikara konomama nemashouyo suyaa (Well, since I feel very relaxed, let's sleep now)
USR3	Yaruki nai desunee (Are you even trying?)
SYS4	Baito ganbare! Yoru wa hanabi ja! (Good luck on your part-time job! We have fireworks at night!)
USR4	Hanabi ni ikun desuka? (Do you go to fireworks?)
SYS5	Ikou to omottoru. Iya, iku. (I'm planning to...No, I'm definitely going)
USR5	Dare to ikun desuka? (With who?)
SYS6	Kimetenai (I haven't decided yet)
USR6	Ima kara kimeruno? (Will you decide from now?)

Figure 4: Example dialogue with the retrieval-based system.

by an external human checker. The checker randomly extracted 100 utterances from the remaining 10% of the open-domain conversation corpus and put each utterance into ProgramD and subjectively evaluated the adequacy of each output. If more than 90% of the output was judged adequate, the rule creation process ended; otherwise, the rules had to be improved for another check. After several iterations, the 90% criterion was met, resulting in 149,300 rules. As this process illustrates, the rules were meticulously crafted. Our rule-based system uses ProgramD loaded with these rules. Figure 3 shows an example dialogue with the rule-based system (collected in our experiment; see Section 4.1).

3.2 Retrieval-based System

We used the Twitter conversation corpus (See Section 2.3) to create a database for retrieval. We extracted two consecutive utterances as input-output pairs and indexed them using the text search engine Lucene (http://lucene.apache.org/core/). For a given utterance as a query, the top-ten utterance pairs are retrieved on the basis of the similarity between the query and the input-part of the indexed pairs. Here, the similarity is the cosine similarity of TF-IDF weighted word vectors. Then, one of the retrieved pairs is randomly selected to produce the system's next utterance. Here, we adopt random selection so that the same utterance won't be uttered for the same input. Since the amount of indexed tweets is large (90M), we consider this to be a reasonable baseline. This system is our replication of IR-Status in (Ritter et al., 2011). Figure 4 shows an example dialogue with the retrieval-based system.

4 Experiment

We evaluated our proposed system in an experiment using human judges. We compared it with the rule-based and retrieval-based systems.

Questionnaire	(a) Rule	(b) Retrieval	(c) Prop (noTW)	(d) Prop (noPAS)	(e) Prop (bi)	(f) Prop (tri)
Q1 Naturalness	**3.88**^{bbe}	***2.68***	3.60^{bb}	3.48^{bb}	3.33^{bb}	3.44^{bb}
Q2 Generation	**4.40**^{bbe}	***2.80***	4.03^{bb}	3.98^{bb}	3.80^{bb}	3.92^{bb}
Q3 Understanding	**3.73**^{bb}	***2.61***	3.46^{bb}	3.33^{bb}	3.16^{b}	3.25^{bb}
Q4 Informativeness	**3.00**^{bb}	***2.24***	2.70	2.65	2.62	2.80^{b}
Q5 Diversity	**3.58**	3.44	***3.08***	3.17	3.27	3.38
Q6 Continuity	**3.87**^{bbf}	***2.63***	3.44^{bb}	3.38^{bb}	3.41^{bb}	3.23^{bb}
Q7 Willingness	**3.60**^{bb}	***2.64***	3.25^{b}	3.14^{b}	3.12^{b}	3.12^{b}
Q8 Satisfaction	**3.62**^{bb}	***2.72***	3.24^{b}	3.21^{b}	3.13	3.13

Table 1: Subjective evaluation results: ratings averaged over all dialogues for each system. Superscripts a–f next to the numbers indicate that the number is statistically better than systems (a)–(f), respectively. Double-letters (e.g., bb) mean $p < 0.01$; otherwise $p < 0.05$. For the statistical test, we used a Steel-Dwass multiple comparison test (Dwass, 1960). The largest and smallest numbers in a row are indicated by bold and bold italic font, respectively.

4.1 Experimental Procedure

We recruited 30 human subjects (14 males and 16 females, ages from 18 to 55). They were paid for their participation. Each participant took part in 24 dialogue sessions, talking four times to each of six different systems. The systems used were (a) the rule-based system, (b) the retrieval-based system, and four different configurations of our proposed system: (c) Prop (noTW), in which utterance search from Twitter is disabled; (d) Prop (noPAS), in which PAS-based generation is disabled; (e) Prop (bi), where N=2 is used for utterance ranking (See Section 2.2); and (f) Prop (tri), in which no module is disabled. All systems start a conversation with a greeting prompt. Each dialogue session lasted for two minutes. Two-minute interaction could be short, but we wanted to test the systems with different topics that can change dialogue-by-dialogue. The participants were instructed to enjoy the conversation with the systems. No dialogue topic was specified. No prior knowledge was provided about the systems, including the number of systems they were to talk to. The order of the systems was randomized. Since the rule-based and retrieval-based systems require less computation, four seconds of delay was inserted before their utterances. After each dialogue, each participant filled out a questionnaire comprising eight items (See the column Questionnaire in Table 1) asking for his/her subjective evaluation of the dialogue on a seven-point Likert scale, where 1 is the worst and 7 the best. We asked the participants not to take into account the delay of system responses for their evaluation. After all 24 sessions, each participant filled in a free-form opinion sheet to end the participation.

4.2 Results and Analyses

Table 1 shows the results of subjective evaluations. As can be seen from the table, the rule-based system performed the best and the retrieval-based system performed the worst. The retrieval-based system was the worst for all questionnaire items except Q5 (Diversity of system utterances); at least the large Twitter database produced diverse utterances. Our proposed systems placed between the rule-based and retrieval-based systems. The averaged scores and the results of statistical tests indicate that our systems are significantly better than the retrieval-based baseline and that our systems' performance is close to that of the rule-based system. The difference between the rule-based and our proposed systems is not statistically significant (except for a small number of cases). When we focus on Q1 (Naturalness of dialogue), Prop (noTW) attained a score of 3.6, which is close to that of the rule-based system (3.88). This indicates that our system has the ability to perform reasonably natural conversation and that it is possible to create a system of rule-based-level naturalness with our architecture. As for other questionnaire items, the difference between our systems and the rule-based system is a little wider in mean scores. Although further examination is needed, this is probably because user satisfaction is related to more sensitive issues such as politeness, linguistic style, consistency, and users' preferences.

When we look at the difference in the four configurations, we see that Prop (noTW) is consistently

System		# Uniq utt	# Uniq word	# Utt	# Word	# Word/Utt	Perplexity
(a) Rule	USR	915	956	1049	5838	5.565	59.81
	SYS	***353***	803	1169	9565	**8.182**	***23.46***
	ALL	1263	1333	2218	**15403**	6.945	60.47
(b) Retrieval	USR	937	995	1067	5007	***4.693***	61.22
	SYS	1016	2043	1186	7744	6.530	80.30
	ALL	**1936**	**2449**	**2253**	12751	5.660	**100.48**
(c) Prop (noTW)	USR	750	889	879	4875	5.546	58.76
	SYS	613	***698***	999	5820	5.826	34.17
	ALL	1345	1187	1878	10695	5.695	57.76
(d) Prop (noPAS)	USR	744	852	***865***	***4823***	5.576	54.82
	SYS	551	807	985	6394	6.491	45.50
	ALL	1279	1242	1850	11217	6.063	67.89

Table 2: The number of unique utterances, unique words, utterances, words, words per utterance, and perplexity for systems (a)–(d). The results for systems (e) and (f) are omitted because they are between those for (c) and (d). USR, SYS, and ALL indicate rows for user, system, and all utterances, respectively. For perplexity calculation, half the data were used to train a trigram language model to be tested with the other half. Bold and bold italic font indicates max and min in each column.

Module	(c) Prop (noTW)	(d) Prop (noPAS)	(e) Prop (bi)	(f) Prop (tri)
Versatile	0.281 (247)	0.274 (237)	0.198 (174)	0.205 (173)
QA	***0.008*** (7)	***0.003*** (3)	***0.008*** (7)	***0.005*** (4)
Personal QA	0.047 (41)	0.054 (47)	0.033 (29)	0.046 (39)
Topic-Inducing	0.143 (126)	0.142 (123)	0.129 (113)	0.157 (132)
Related-word	0.060 (53)	0.090 (78)	0.043 (38)	0.027 (23)
Twitter	N/A	**0.358** (310)	0.187 (164)	0.253 (213)
PAS	**0.383** (337)	N/A	**0.280** (246)	**0.265** (223)
Pattern	0.032 (28)	0.031 (27)	0.081 (71)	0.021 (18)
User PAS	0.046 (40)	0.046 (40)	0.042 (37)	0.021 (18)

Table 3: Selected ratios (raw counts in parentheses) for the generation modules. Bold and bold italic font indicate max and min in each column.

better than the others except for Q4 and Q5. Since the main difference is whether Twitter sentences are used, this is probably the cause. The reason could be the inconsistency of linguistic styles in Twitter or the noise that could not be suppressed by the filtering. Since Twitter sentences surely augment diversity, we would like to consider ways to make better use of them, for example, by normalizing the linguistic style and applying stricter filters. There is a slight tendency for Prop (tri) to be preferred to Prop (bi), which is reasonable because it uses more context for deciding the next utterance. In the future, we would like to pursue methods that can exploit longer context, such as entity grids (Barzilay and Lapata, 2008) and co-reference structures (Swanson and Gordon, 2012).

We performed a brief analysis of the collected dialogues. Table 2 shows, for each system, the number of unique utterances, unique words, utterances, words, words per utterance, and perplexity. It can be seen that the utterances of the rule-based system are very rigid: the perplexity is very low (23.46) and there are only 353 unique utterances, which is about half of that of the other systems. It is interesting that, despite this fact, the rule-based system was perceived to produce the most diverse utterances by questionnaire. Since the rule-based system produced much longer utterances (8.182), this probably had a positive effect for the perceived diversity. In terms of natural interaction, it is not desirable for one participant to contribute more than the other. In this respect, our proposed systems seem appropriate because the users and the systems exchange a similar number of words per utterance.

Table 3 shows the selected ratios for the generation modules. It can be seen that all modules contributed to conversation. The most frequent ones were Twitter and PAS-based generation, followed by the versatile and topic-inducing modules. Although QA and personal QA were not used as frequently for output, when we examined the logs, we found that there were many cases where these modules could not obtain any answer from the QA API or the PDB. Since answering questions is a basic function in conversation, this needs to be improved. Similarly, we also want to evaluate the contribution of each module quantitatively, for example, by associating the behavior of each module with user subjective evaluations in a framework similar to PARADISE (Walker et al., 2000). Enabling this kind of analysis is a clear benefit of having an architecture such as the one we proposed.

5 Summary

This paper proposed an architecture for an open-domain conversational system and evaluated an implemented system. The results indicate that our architecture enables better dialogue than a retrieval-based baseline using a large Twitter database. Although our system could not reach the level of a carefully crafted rule-based system and still has a number of limitations, our architecture can achieve naturalness close to that of the rule-based system. The contributions of this paper are that we introduced a viable architecture for an open-domain conversational system and experimentally verified its effectiveness. Rather than creating rules on the basis of developers' intuition, our architecture will enable module-by-module development, which will lead to rapid improvement in open-domain conversational systems in the future.

References

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

B Batacharia, D Levy, R Catizone, A Krotov, and Y Wilks. 1999. CONVERSE: a conversational companion. In *Machine conversations*, pages 205–215. Springer.

Fumihiro Bessho, Tatsuya Harada, and Yasuo Kuniyoshi. 2012. Dialog system using real-time crowdsourcing and Twitter large-scale corpus. In *Proc. SIGDIAL*, pages 227–231.

Timothy Bickmore and Justine Cassell. 2000. How about this weather? social dialogue with embodied conversational agents. In *Proc. AAAI Fall Symposium on Socially Intelligent Agents*.

Timothy W. Bickmore and Rosalind W. Picard. 2005. Establishing and maintaining long-term human-computer relationships. *ACM Transactions on Computer-Human Interaction*, 12(2):293–327.

Meyer Dwass. 1960. Some k-sample rank-order tests. *Contributions to probability and statistics*, pages 198–202.

Suzanne Eggins and Diana Slade. 2005. *Analysing Casual Conversation*. Equinox Publishing Ltd.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. 2008. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74.

Takeshi Fuchi and Shinichiro Takagi. 1998. Japanese morphological analyzer using word co-occurrence: JTAG. In *Proc. COLING*, volume 1, pages 409–413.

Barbara J Grosz and Candace L Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational linguistics*, 12(3):175–204.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. COLING*, volume 2, pages 539–545.

Ryuichiro Higashinaka and Hideki Isozaki. 2008. Automatically acquiring causal expression patterns from relation-annotated corpora to improve question answering for why-questions. *ACM Transactions on Asian Language Information Processing (TALIP)*, 7(2):6.

Ryuichiro Higashinaka, Kugatsu Sadamitsu, Kuniko Saito, and Nozomi Kobayashi. 2013. Question answering technology for pinpointing answers to a wide range of questions. *NTT Technical Review*, 11(7).

Ryuichiro Higashinaka, Nozomi Kobayashi, Toru Hirano, Chiaki Miyazaki, Toyomi Meguro, Toshiro Makino, and Yoshihiro Matsuo. 2014. Syntactic filtering and content-based retrieval of Twitter sentences for the generation of system utterances in dialogue systems. In *Proc. IWSDS*, pages 113–123.

Shinsuke Higuchi, Rafal Rzepka, and Kenji Araki. 2008. A casual conversation system using modality and word associations retrieved from the web. In *Proc. EMNLP*, pages 382–390.

Eduard H Hovy. 1991. *Approaches to the planning of coherent text*. Springer.

Satoru Ikehara, Masahiro Miyazaki, Satoshi Shirai, Akio Yokoo, Hiromi Nakaiwa, Kentaro Ogura, Yoshifumi Ooyama, and Yoshihiko Hayashi. 1997. Goi-Taikei—A Japanese lexicon.

Kenji Imamura, Kuniko Saito, and Tomoko Izumi. 2009. Discriminative approach to predicate-argument structure analysis with zero-anaphora resolution. In *Proc. ACL-IJCNLP (Short Papers)*, pages 85–88.

Kenji Imamura, Ryuichiro Higashinaka, and Tomoko Izumi. 2014a. Adaptaion of predicate-argument structure analysis with zero-anaphora resolution to dialogues. In *Proc. Annual Meeting of the Association for Natural Language Processing*, pages 709–712. (In Japanese).

Kenji Imamura, Ryuichiro Higashinaka, and Tomoko Izumi. 2014b. Predicate-argument structure analysis with zero-anaphora resolution for dialogue systems. In *Proc. COLING*.

Allen Ivey, Mary Ivey, and Carlos Zalaquett. 2013. *Intentional interviewing and counseling: Facilitating client development in a multicultural society*. Cengage Learning.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proc. KDD*, pages 133–142.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, pages 282–289.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *Proc. ACL*, volume 1, pages 545–552.

Toyomi Meguro, Yasuhiro Minami, Ryuichiro Higashinaka, and Kohji Dohsaka. 2013. Learning to control listening-oriented dialogue using partially observable markov decision processes. *ACM Transactions on Speech and Language Processing (TSLP)*, 10(4):15.

Masaaki Nagata, Kuniko Saito, and Yoshihiro Matsuo. 2006. Japanese natural language search system: Web Answers. *Proc. Annual Meeting of the Association for Natural Language Processing*, pages 320–323. (In Japanese).

Mikio Nakano, Noboru Miyazaki, Norihito Yasuda, Akira Sugiyama, Jun-ichi Hirasawa, Kohji Dohsaka, and Kiyoaki Aikawa. 2000. WIT: A toolkit for building robust and real-time spoken dialogue systems. In *Proc. SIGDIAL*, pages 150–159.

Ani Nenkova, Agustín Gravano, and Julia Hirschberg. 2008. High frequency word entrainment in spoken dialogue. In *Proc. ACL-HLT (Short Papers)*, pages 169–172.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103.

Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-driven response generation in social media. In *Proc. EMNLP*, pages 583–593.

Emanuel A Schegloff and Harvey Sacks. 1973. Opening up closings. *Semiotica*, 8(4):289–327.

Masahiro Shibata, Tomomi Nishiguchi, and Yoichi Tomiura. 2009. Dialog system for open-ended conversation using web documents. *Informatica (Slovenia)*, 33(3):277–284.

Hiroaki Sugiyama, Toyomi Meguro, Ryuichiro Higashinaka, and Yasuhiro Minami. 2013. Open-domain utterance generation for conversational dialogue systems using web-scale dependency structures. In *Proc. SIGDIAL*, pages 334–338.

Hiroaki Sugiyama, Toyomi Meguro, Ryuichiro Higashinaka, and Yasuhiro Minami. 2014a. Large-scale collection and analysis of personal question-answer pairs for conversational agents. In *Proc. IVA*. (to appear).

Hiroaki Sugiyama, Toyomi Meguro, Ryuichiro Higashinaka, and Yasuhiro Minami. 2014b. Response generation for questions about dialogue system's personality. In *JSAI Technical Report (SIG-SLUD-B303)*, pages 33–38. (In Japanese).

Reid Swanson and Andrew S Gordon. 2012. Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(3):16.

Shota Takeuchi, Tobias Cincarek, Hiromichi Kawanami, Hiroshi Saruwatari, and Kiyohiro Shikano. 2007. Construction and optimization of a question and answer database for a real-environment speech-oriented guidance system. In *Proc. COCOSDA*.

Wataru Uchida, Chiaki Morita, and Takeshi Yoshimura. 2013. Knowledge Q&A: Direct answers to natural questions. *NTT DOCOMO Technical Journal*, 14(4):4–9.

Ellen M Voorhees and DM Tice. 2000. Overview of the TREC-9 question answering track. In *Proc. TREC*.

Marilyn Walker, Sharon Cote, and Masayo Iida. 1994. Japanese discourse and the process of centering. *Computational linguistics*, 20(2):193–232.

Marilyn Walker, Candace Kamm, and Diane Litman. 2000. Towards developing general models of usability with PARADISE. *Natural Language Engineering*, 6(3&4):363–377.

Marilyn Walker, Rebecca Passonneau, and Julie E. Boland. 2001. Quantitative and qualitative evaluation of DARPA Communicator spoken dialogue systems. In *Proc. ACL*, pages 515–522.

Richard S. Wallace. 2004. *The Anatomy of A.L.I.C.E.* A.L.I.C.E. Artificial Intelligence Foundation, Inc.

Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. 2013. The dialog state tracking challenge. In *Proc. SIGDIAL*, pages 404–413.

The Impact of Deep Hierarchical Discourse Structures in the Evaluation of Text Coherence

Vanessa Wei Feng[1], **Ziheng Lin**[2], **and Graeme Hirst**[1]
[1] Department of Computer Science
University of Toronto
{weifeng, gh}@cs.toronto.edu
[2] Singapore Press Holdings
linziheng@gmail.com

Abstract

Previous work by Lin et al. (2011) demonstrated the effectiveness of using discourse relations for evaluating text coherence. However, their work was based on discourse relations annotated in accordance with the Penn Discourse Treebank (PDTB) (Prasad et al., 2008), which encodes only very shallow discourse structures; therefore, they cannot capture long-distance discourse dependencies. In this paper, we study the impact of deep discourse structures for the task of coherence evaluation, using two approaches: (1) We compare a model with features derived from discourse relations in the style of Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), which annotate the full hierarchical discourse structure, against our re-implementation of Lin et al.'s model; (2) We compare a model encoded using only shallow RST-style discourse relations, against the one encoded using the complete set of RST-style discourse relations. With an evaluation on two tasks, we show that deep discourse structures are truly useful for better differentiation of text coherence, and in general, RST-style encoding is more powerful than PDTB-style encoding in these settings.

1 Introduction

In a well-written text, utterances are not simply presented in an arbitrary order; rather, they are presented in a logical and coherent form, so that the readers can easily interpret the meaning that the writer wishes to present. Therefore, coherence is one of the most essential aspects of text quality. Given its importance, the automatic evaluation of text coherence is one of the crucial components of many NLP applications.

A particularly popular model for the evaluation of text coherence is the entity-based local coherence model of Barzilay and Lapata (B&L) (2005; 2008), which extracts mentions of entities in the text, and models local coherence by the transitions, from one sentence to the next, in the grammatical role of each mention. Since the initial publication of this model, a number of extensions have been proposed, the majority of which are focused on enriching the original feature set. However, these enriched feature sets are usually application-specific, i.e., it requires a certain expertise and intuition to conceive good features.

In contrast, we seek insights of better feature encoding from a more general problem: discourse parsing (to be introduced in Section 2). Discourse parsing aims to identify the discourse relations held among various discourse units in the text. Therefore, one can expect that discourse parsing provides useful information to the evaluation of text coherence, because, essentially, the existence and the distribution of discourse relations are the basis of the coherence in a text.

In fact, there is already evidence showing that discourse relations can help better capture text coherence. Lin et al. (2011) use a PDTB-style discourse parser (to be introduced in Section 2.1) to identify discourse relations in the text, and they represent a text by entities and their associated discourse roles in each sentence. In their experiments, using discourse roles alone, their model performs very similar or even better than B&L's model. Combining their discourse role features with B&L's entity-based transition features further improves the performance.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 940–949, Dublin, Ireland, August 23-29 2014.

S_1: The dollar finished lower yesterday, after tracking another rollercoaster session on Wall Street.
S_2: [Concern about the volatile U.S. stock market had faded in recent sessions]$C_{2.1}$, [and traders appeared content to let the dollar languish in a narrow range until tomorrow, when the preliminary report on third-quarter U.S. gross national product is released.]$C_{2.2}$
S_3: But seesaw gyrations in the Dow Jones Industrial Average yesterday put Wall Street back in the spotlight and inspired market participants to bid the U.S. unit lower.

Three discourse relations are presented in the text above:

1. Implicit *EntRel* between S_1 as Arg1, and S_2 as Arg2.
2. Explicit *Conjunction* within S_2: $C_{2.1}$ as Arg1, $C_{2.2}$ as Arg2, with *and* as the connective.
3. Explicit *Contrast* between S_2 as Arg1 and S_3 as Arg2, with *but* as the connective.

Figure 1: An example text fragment composed of three sentences, and its PDTB-style discourse relations.

However, PDTB-style discourse relations encode only very shallow discourse structures, i.e., the relations are mostly local, e.g., within a single sentence or between two adjacent sentences. Therefore, in general, features derived from PDTB-style discourse relations cannot capture long discourse dependency, and thus the resulting model is still limited to being a local model. Nonetheless, long-distance discourse dependency could be quite useful for capturing text coherence from a global point of view.

Therefore, in this paper, we study the effect of deep hierarchical discourse structure in the evaluation of text coherence, by adopting two approaches to perform a direct comparison between models that incorporate deep hierarchical discourse structures and models with shallow structures. To evaluate our models, we conduct experiments on two datasets, each of which resembles a real sub-task in the evaluation of text coherence: **sentence ordering** and **essay scoring**. On both tasks, the model derived from deep discourse structures is shown to be more powerful than the model derived from shallow discourse structures. Moreover, for sentence ordering, combining our model with entity-based transition features achieves the best performance. However, for essay scoring, the combination is detrimental.

2 Discourse parsing

Discourse parsing is the problem of identifying the discourse structure within a text, by recognizing the specific type of its discourse relations, such as *Contrast*, *Explanation*, and *Causal* relations. Although discourse parsing is still relatively less well-studied, a number of theories have been proposed to capture different rhetorical characteristics or to serve different applications.

Currently, the two main directions in the study of discourse parsing are PDTB-style and RST-style parsing. These two directions are based on distinct theoretical frameworks, and each can be potentially useful for particular kinds of downstream applications. As will be discussed shortly, the major difference between PDTB- and RST-style discourse parsing is the notion of deep hierarchical discourse structure, which, according to our hypothesis, can be very useful for recognizing text coherence.

2.1 PDTB-style Discourse Parsing

The Penn Discourse Treebank (PDTB), developed by Prasad et al. (2008), is currently the largest discourse-annotated corpus, consisting of 2159 Wall Street Journal articles. The annotation in PDTB adopts the predicate-argument view of discourse relations, where a discourse connective (e.g., *because*) is treated as a predicate that takes two text spans as its arguments. The argument that the discourse connective structurally attaches to is called Arg2, and the other argument is called Arg1. In PDTB, relations are further categorized into *explicit* and *implicit* relations: a relation is explicit if there is an explicit discourse connective presented in the text; otherwise, it is implicit. PDTB relations focus more on *locality* and *adjacency*: explicit relations seldom connect text units beyond local context; for implicit relations,

S_1: [The dollar finished lower yesterday,]e_1 [after tracking another rollercoaster session on Wall Street.]e_2
S_2: [Concern about the volatile U.S. stock market had faded in recent sessions,]e_3 [and traders appeared content to let the dollar languish in a narrow range until tomorrow,]e_4 [when the preliminary report on third-quarter U.S. gross national product is released.]e_5
S_3: [But seesaw gyrations in the Dow Jones Industrial Average yesterday put Wall Street back in the spotlight]e_6 [and inspired market participants to bid the U.S. unit lower.]e_7

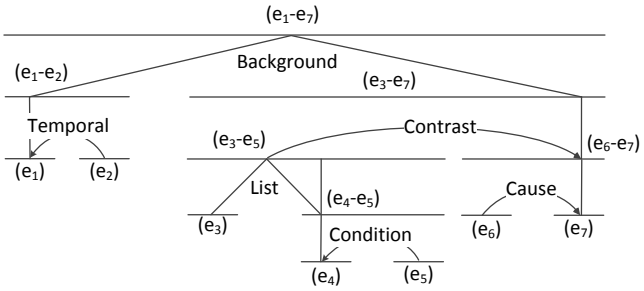


Figure 2: An example text fragment composed of seven EDUs, and its RST discourse tree representation.

only adjacent sentences within paragraphs are examined for the existence of implicit relations.

The PDTB-style discourse parsing is thus the type of framework in accordance with the PDTB, which extracts the discourse relations in a text, by identifying the presence of discourse connectives, the associated discourse arguments, and the specific types of the relations. An example text fragment is shown in Figure 1, consisting of three sentences, S_1, S_2, and S_3. A sentence may further contain clauses, e.g., $C_{2.1}$ and $C_{2.2}$ in S_2. The three PDTB-style discourse relations in this text are explained below the text.

2.2 RST-style Discourse Parsing

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

Consider Figure 2, in which the same example as in Figure 1 is chunked into seven EDUs (e_1-e_7), segmented by square brackets. Its discourse tree representation is shown below in the figure, following the notational convention of RST. The two EDUs e_1 and e_2 are related by a mononuclear relation *Temporal*, where e_1 is the more salient span; e_4 and e_5 are related by *Condition*, with e_4 as the nucleus; and e_6 and e_7 are related by *Cause*, with e_7 as the nucleus. Then, the spans (e_3-e_5) and (e_6-e_7) are related by *Contrast* to form a higher-level discourse structure, and so on. Finally, a *Background* relation merges the span (e_1-e_2) and (e_3-e_7) on the top level of the tree.

As can be seen, thanks to the tree-structured representation of RST, compared to PDTB-style representation, we have a full hierarchy of discourse relations in the text: discourse relations exist not only in a local context, but also on higher text levels, such as between S_1 and the concatenation of S_2 and S_3.

3 Entity-based Local Coherence Model

The entity-based local coherence model was initially developed by Barzilay and Lapata (B&L) (2005; 2008). The fundamental assumption of this model is that a document makes repeated reference to elements of a set of entities that are central to its topic.

For a document d, an entity grid is constructed, in which the columns represent the entities referred

S_1: [**The dollar**]$_{\mathbf{S}}$ finished lower [**yesterday**]$_{\mathbf{X}}$, after tracking [**another rollercoaster session**]$_{\mathbf{O}}$ on [**Wall Street**]$_{\mathbf{X}}$.
S_2: [**Concern**]$_{\mathbf{S}}$ about [**the volatile U.S. stock market**]$_{\mathbf{X}}$ had faded in [**recent sessions**]$_{\mathbf{X}}$, and [**traders**]$_{\mathbf{S}}$ appeared content to let [**the dollar**]$_{\mathbf{S}}$ languish in [**a narrow range**]$_{\mathbf{X}}$ until [**tomorrow**]$_{\mathbf{X}}$, when [**the preliminary report**]$_{\mathbf{S}}$ on [**third-quarter U.S. gross national product**]$_{\mathbf{X}}$ is released.
S_3: But [**seesaw gyrations**]$_{\mathbf{S}}$ in [**the Dow Jones Industrial Average**]$_{\mathbf{X}}$ [**yesterday**]$_{\mathbf{X}}$ put [**Wall Street**]$_{\mathbf{O}}$ back in [**the spotlight**]$_{\mathbf{X}}$ and inspired [**market participants**]$_{\mathbf{O}}$ to bid [**the U.S. unit**]$_{\mathbf{S}}$ lower.

	dollar	**yesterday**	**session**	**Wall Street**	**concern**	**market**	**sessions**	**traders**	**range**	**tomorrow**	**report**	**GNP**	**gyrations**	**DJIA**	**spotlight**	**participants**
S_1	S	X	O	X	-	-	-	-	-	-	-	-	-	-	-	-
S_2	S	-	-	-	S	X	S	X	X	X	S	X	-	-	-	-
S_3	S	X	-	O	-	-	-	-	-	-	-	-	S	X	X	O

Table 1: The entity grid for the example text with three sentences and eighteen entities. Grid cells correspond to grammatical roles: subjects (S), objects (O), or neither (X).

to in d, and rows represent the sentences. Each cell corresponds to the grammatical role of an entity in the corresponding sentence: subject (S), object (O), neither (X), or nothing ($-$), and an entity is defined as a class of coreferent noun phrases. If the entity serves in multiple roles in a single sentence, then we resolve its grammatical role following the priority order: $\mathrm{S} \succ \mathrm{O} \succ \mathrm{X} \succ -$. Consider the text in our previous examples; its entity grid is shown in Table 1, and the entities are highlighted in boldface in the text above[1]. A local transition is defined as a sequence $\{\mathrm{S},\mathrm{O},\mathrm{X},-\}^n$, representing the occurrence and grammatical roles of an entity in n adjacent sentences. Such transition sequences can be extracted from the entity grid as continuous subsequences in each column. For example, the entity *dollar* in Table 1 has a bigram transition $\{\mathrm{S},\mathrm{S}\}$ from sentence 1 to 2. The entity grid is then encoded as a feature vector $\Phi(d) = (p_1(d), p_2(d), \ldots, p_m(d))$, where $p_t(d)$ is the normalized frequency of the transition t in the entity grid, and m is the number of transitions with length no more than a predefined length k. $p_t(d)$ is computed as the number of occurrences of t in the entity grid of document d, divided by the total number of transitions of the same length. Moreover, entities are differentiated by their salience — an entity is deemed to be salient if it occurs at least l times in the text, and non-salient otherwise — and transitions are computed separately for salient and non-salient entities.

3.1 Extension: Lin et al.'s Discourse Role Matrix

As mentioned previously, most extensions to B&L's entity-based local coherence model focus on enriching the feature set, including the work of Filippova and Strube (2007), Cheung and Penn (2010), Elsner and Charniak (2011), and Lin et al. (2011). To the best of our knowledge, the only exception is Feng and Hirst (2012a)'s extension from the perspective of improving the learning procedure.

Among various extensions to B&L's entity-based local coherence model, the one most related to ours is Lin et al. (2011)'s work on encoding a text as a set of entities with their associated discourse roles. Lin et al. observed that coherent texts preferentially follow certain relation patterns. However, simply using such patterns to measure the coherence of a text can result in feature sparseness. To solve this problem, they expand the relation sequence into a discourse role matrix, as shown in Table 2. Columns correspond to the entities in the text and rows represent the contiguous sentences. Each cell $\langle E_i, S_j \rangle$ corresponds to the set of discourse roles that the entity E_i serves as in sentence S_j. For example, the entity *yesterday* from S_3 takes part in Arg2 of the last relation, so the cell $\langle yesterday, S_3 \rangle$ contains the role *Contrast*.Arg2.

[1]Text elements are considered to be a single entity with multiple mentions if they refer to the same object or concept in the world, even if they have different textual realizations; e.g., *dollar* in S_1 and *U.S. unit* in S_3 refer to the same entity.

	dollar	**yesterday**	**session**	**Wall Street**	**concern**	**market**
S_1	*EntRel*.Arg1	*EntRel*.Arg1	*EntRel*.Arg1	*EntRel*.Arg1	*nil*	*nil*
S_2	*EntRel*.Arg2 *Conj*.Arg2 *Contrast*.Arg1	*nil*	*nil*	*nil*	*EntRel*.Arg2 *Conj*.Arg1 *Contrast*.Arg1	*EntRel*.Arg2 *Conj*.Arg1 *Contrast*.Arg1
S_3	*Contrast*.Arg2	*Contrast*.Arg2	*nil*	*Contrast*.Arg2	*nil*	*nil*

Table 2: A fragment of Lin et al.'s PDTB-style discourse role matrix for the example text with the first six entities across three sentences.

An entry may be empty (with a symbol *nil*, as in $\langle yesterday, S_2 \rangle$) or contain multiple discourse roles (as in $\langle dollar, S_2 \rangle$). Next, the frequencies of the discourse role transitions of lengths 2 and 3, e.g., *EntRel*.Arg1 $\rightarrow$ *Conjunction*.Arg2 and *EntRel*.Arg1 $\rightarrow$ *nil* $\rightarrow$ *Contrast*.Arg2, are calculated with respect to the matrix. For example, the frequency of *EntRel*.Arg1 $\rightarrow$ *Conjunction*.Arg2 is $1/24 = 0.042$ in Table 2.

4 Methodology

As discussed in Section 1, the main objective of our work is to study the impact of deep hierarchical discourse structures in the evaluation of text coherence. In order to conduct a direct comparison between a model with features derived from deep hierarchical discourse relations and a model with features derived from shallow discourse relations only, we adopt two separate approaches: (1) We implement a model with features derived from RST-style discourse relations, and compare it against a model with features derived from PDTB-style relations. (2) In the framework of RST-style discourse parsing, we deprive the model of any information from higher-level discourse relations and compare its performance against the model that uses the complete set of discourse relations. Moreover, as a baseline, we also re-implemented B&L's entity-based local coherence model, and we will study the effect of incorporating one of our discourse feature sets into this baseline model. Therefore, we have four ways to encode discourse relation features, namely, entity-based, PDTB-style, full RST-style, and shallow RST-style.

4.1 Entity-based Feature Encoding

In entity-based feature encoding, our goal is to formulate a text into an entity grid, such as the one shown in Table 1, from which we extract entity-based local transitions. In our re-implementation of B&L, we use the same parameter settings as B&L's original model, i.e., the optimal transition length $k = 3$ and the salience threshold $l = 2$. However, when extracting entities in each sentence, e.g., *dollar*, *yesterday*, etc., we do not perform coreference resolution; rather, for better coverage, we follow the suggestion of Elsner and Charniak (2011) and extract all nouns (including non-head nouns) as entities. We use the Stanford dependency parser (de Marneffe et al., 2006) to extract nouns and their grammatical roles. This strategy of entity extraction also applies to the other three feature encoding methods to be described below.

4.2 PDTB-style Feature Encoding

To encode PDTB-style discourse relations into the model, we parse the texts using an end-to-end PDTB-style discourse parser[2] developed by Lin et al. (2014). The F_1 score of this parser is around 85% for recognizing explicit relations and around 40% for recognizing implicit relations. A text is thus represented by a discourse role matrix in the same way as shown in Table 2. Most parameters in our PDTB-style feature encoding follow those of Lin et al. (2011): each entity is associated with the fully-fledged discourse roles, i.e., with type and argument information included; the maximum length of discourse role transitions is 3; and transitions are generated separately for salient and non-salient entities with a threshold set at 2. However, compared to Lin et al.'s model, there are two differences in our re-implementation, and evaluated on a held-out development set, these modifications are shown to be effective in improving the performance.

[2] `http://wing.comp.nus.edu.sg/~linzihen/parser/`

	dollar	yesterday	session	Wall Street	concern	market
S_1	*Background*.N *Temporal*.N	*Background*.N *Temporal*.N	*Temporal*.S	*Temporal*.S	*nil*	*nil*
S_2	*List*.N *Condition*.N *Contrast*.S	*nil*	*nil*	*nil*	*List*.N *Contrast*.S	*List*.N *Contrast*.S
S_3	*Contrast*.N *Background*.N *Cause*.N	*Cause*.S	*nil*	*Cause*.S	*nil*	*nil*

Table 3: A fragment of the full RST-style discourse role matrix for the example text with the first six entities across three sentences.

First, we differentiate between intra- and multi-sentential discourse relations, which is motivated by a finding in the field of RST-style discourse parsing — distributions of various discourse relation types are quite distinct between intra-sentential and multi-sentential instances (Feng and Hirst, 2012b; Joty et al., 2012) — and we assume that a similar phenomenon exists for PDTB-style discourse relations. Therefore, we assign two sets of discourse roles to each entity: intra-sentential and multi-sentential roles, which are the roles that the entity plays in the corresponding intra- and multi-sentential relations.

Second, instead of Level-1 PDTB discourse relations (6 in total), we use Level-2 relations (18 in total) in feature encoding, so that richer information can be captured in the model, resulting in $18 \times 2 = 36$ different discourse roles with argument attached. We then generate four separate set of features for the combination of intra-/multi-sentential discourse relation roles, and salient/non-salient entities, among which transitions consisting of only *nil* symbols are excluded. Therefore, the total number of features in PDTB-style encoding is $4 \times (36^2 + 36^3 - 2) \approx 192$K.

4.3 Full RST-style Feature Encoding

For RST-style feature encoding, we parse the texts using an end-to-end RST-style discourse parser developed by Feng and Hirst (2014), which produces a discourse tree representation for each text, such as the one shown in Figure 2. For relation labeling, the overall accuracy of this discourse parser is 58%, evaluated on the RST-DT.

We encode the RST-style discourse relations in a similar fashion to PDTB-style encoding. However, since the definition of discourse roles depends on the particular discourse framework, here, we adapt Lin et al.'s PDTB-style encoding by replacing the PDTB-style discourse relations with RST-style discourse relations, and the argument information (Arg1 or Arg2) by the nuclearity information (nucleus or the satellite) in an RST-style discourse relation. More importantly, in order to reflect the hierarchical structure in an RST-style discourse parse tree, when extracting the set of discourse relations that an entity participates in, we find all those discourse relations that the entity appears in the main EDUs of each relation[3] and represent the role of the entity in each of these discourse relations. In this way, we can encode long-distance discourse relations for the most relevant entities. For example, considering the RST-style discourse tree representation in Figure 2, we encode the *Background* relation for the entities *dollar* and *yesterday* in S_1, as well as the entity *dollar* in S_3, but not for the remaining entities in the text, even though the *Background* relation covers the whole text. The corresponding full RST-style discourse role matrix for the example text is shown in Table 3.

As in PDTB-style feature encoding, we differentiate between intra- and multi-sentential discourse relations; we use 17 coarse-grained classes of RST-style relations in feature encoding; the optimal transi-

[3]The main EDUs of a discourse relation are the EDUs obtained by traversing the discourse subtree in which the relation of interest constitutes the root node, following the nucleus branches down to the leaves. For instance, for the RST discourse tree in Figure 2, the main EDUs of the *Background* relation on the top level are $\{e_1, e_7\}$, and the main EDUs of the *List* relation among (e_3-e_5) are $\{e_3, e_4\}$.

tion length k is 3; and the salience threshold l is 2. The total number of features in RST-style encoding is therefore $4 \times (34^2 + 34^3 - 2) \approx 162$K, which is roughly the same as that in PDTB-style feature encoding.

4.4 Shallow RST-style Feature Encoding

Shallow RST-style encoding is almost identical to full RST-style encoding, as introduced in Section 4.3, except that, when we derive discourse roles, we consider shallow discourse relations only. To be consistent with the majority of PDTB-style discourse relations, we define shallow discourse relations as those relations which hold between text spans of the same sentence, or between two adjacent sentences. For example, in Figure 2, the *Background* relation between (e_1-e_2) and (e_3-e_7) is not a shallow discourse relation (it holds between a single sentence and the concatenation of two sentences), and thus will be excluded from shallow RST-style feature encoding.

5 Experiments

To evaluate our proposed model with deep discourse structures encoded, we conduct two series of experiments on two different datasets, each of which simulates a sub-task in the evaluation of text coherence, i.e., **sentence ordering** and **essay scoring**. Since text coherence is a matter of degree rather than a binary classification, in both evaluation tasks we formulate the problem as a pairwise preference ranking problem. Specifically, given a set of texts with different degrees of coherence, we train a ranker which learns to prefer a more coherent text over a less coherent counterpart. Accuracy is therefore measured as the fraction of correct pairwise rankings as recognized by the ranker. In our experiments, we use the SVM^{light} package[4] (Joachims, 1999) with the ranking configuration, and all parameters are set to their default values.

5.1 Sentence Ordering

The task of sentence ordering, which has been extensively studied in previous work, attempts to simulate the situation where, given a predefined set of information-bearing items, we need to determine the best order in which the items should be presented. As argued by Barzilay and Lapata (2005), sentence ordering is an essential step in many content-generation components, such as multi-document summarization.

In this task, we use a dataset consisting of a subset of the Wall Street Journal (WSJ) corpus, in which the minimum length of a text is 20 sentences, and the average length is 41 sentences. For each text, we create 20 random permutations by shuffling the original order of the sentences. In total, we have 735 source documents and $735 \times 20 = 14,700$ permutations. Because the RST-style discourse parser we use is trained on a fraction of the WSJ corpus, we remove the training texts from our dataset, to guarantee that the discourse parser will not perform exceptionally well on some particular texts. However, since the PDTB-style discourse parser we use is trained on almost the entire WSJ corpus, we cannot do the same for the PDTB-style parser.

In this experiment, our learning instances are pairwise ranking preferences between a source text and one of its permutations, where the source text is always considered more coherent than its permutations. Therefore, we have $735 \times 20 = 14,700$ total pairwise rankings, and we conduct 5-fold cross-validation on five disjoint subsets. In each fold, one-fifth of the rankings are used for testing, and the rest for training.

5.2 Essay Scoring

The second task is essay scoring, and we use a subset of International Corpus of Learner English (ICLE) (Granger et al., 2009). The dataset consists of 1,003 essays about 34 distinct topics, written by university undergraduates speaking 14 native languages who are learners of English as a Foreign Language. Each essay has been annotated with an organization score from 1 to 4 at half-point increments by Persing et al. (2010). We use these organization scores to *approximate* the degrees of coherence in the essays. The average length of the essays is 32 sentences, and the average organization score is 3.05, with a standard deviation of 0.59.

[4] `http://svmlight.joachis.org/`

Model		sentence ordering	essay scoring
No discourse structure	Entity	95.1	66.4
Shallow discourse structures	PDTB	97.2	82.2
	PDTB&Entity	97.3	83.3
	Shallow RST	98.5	87.2
	Shallow RST&Entity	98.8	87.2
Deep discourse structures	Full RST	99.1	**88.3**
	Full RST&Entity	**99.3**	87.7

Table 4: Accuracy (%) of various models on the two evaluation tasks: sentence ordering and essay scoring. For sentence ordering, accuracy difference is significant with $p < .01$ for all pairs of models except between PDTB and PDTB&Entity. For essay scoring, accuracy difference is significant with $p < .01$ for all pairs of models except between shallow RST and shallow RST&Entity. Significance is determined with the Wilcoxon signed-rank test.

In this experiment, our learning instances are pairwise ranking preferences between a pair of essays on the same topic written by students speaking the same native language, excluding pairs with the same organization score. In total, we have 22,362 pairwise rankings. Similarly, we conduct 5-fold cross-validations on these rankings.

In fact, the two datasets used in the two evaluation tasks reflect different characteristics by themselves. The WSJ dataset, although somewhat artificial due to the permuting procedure, is representative of texts with well-formed syntax. By contrast, the ICLE dataset, although not artificial, contains occasional syntactic errors, because the texts are written by non-native English speakers. Therefore, using these two distinct datasets allows us to evaluate our models in tasks where different challenges may be expected.

6 Results

In this section, we demonstrate the performance of our models with discourse roles encoded in one of the three ways: PDTB-style, full RST-style or shallow RST-style, and compare against their combination with our re-implemented B&L's entity-based local transition features. The evaluation is conducted on the two tasks, sentence ordering and essay scoring, and the accuracy is reported as the fraction of correct pairwise rankings averaged over 5-fold cross-validation.

The performance of various models is shown in Table 4. The first section of the table shows the results of our re-implementation of B&L's entity-based local coherence model, representing the effect with **no** discourse structure encoded. The second section shows the results of four models with **shallow** discourse structures encoded, including the two basic models, PDTB-style and shallow RST-style feature encoding, and their combination with the entity-based feature encoding. The last section shows the results of our models with **deep** discourse structures encoded, including the RST-style feature encoding and its combination with the entity-base feature encoding. With respect to the performance, we observe a number of consistent patterns across both evaluation tasks.

First, with **no** discourse structure encoded, the entity-based model (the first row) performs the worst among all models, suggesting that discourse structures are truly important and can capture coherence in a more sophisticated way than pure grammatical roles. Moreover, the performance gap is particularly large for essay scoring, which is probably due to the fact that, as argued by Persing et al. (2010), the organization score, which we use to approximate the degrees of coherence, is not equivalent to text coherence. Organization relates more to the logical development in the texts, while coherence is about lexical and semantic continuity; but discourse relations can capture the logical relations at least to some extent.

Secondly, with **deep** discourse structures encoded, the RST-style model in the third section significantly outperforms ($p < .01$) the models with shallow discourse structures, i.e., the PDTB-style and

shallow RST-style models in the middle section, confirming our intuition that deep discourse structures are more powerful than shallow structures. This is also the case when entity-based features are included.

Finally, considering the models in the middle section of the table, we can gain more insight into the difference between PDTB-style and RST-style encoding. As can be seen, even without information from the more powerful deep hierarchical discourse structures, shallow RST-style encoding still significantly outperforms PDTB-style encoding on both tasks ($p < .01$). This is primarily due to the fact that the discourse relations discovered by RST-style parsing have wider coverage of the text[5], and thus induce richer information about the text. Therefore, because of its ability to annotate deep discourse structures and its better coverage of discourse relations, RST-style discourse parsing is generally more powerful than PDTB-style parsing, as far as coherence evaluation is concerned.

However, with respect to combining full RST-style features with entity features, we have contradictory results on the two tasks: for sentence ordering, the combination is significantly better than each single model, while for essay scoring, the combination is worse than using RST-style features alone. This is probably related to the previously discussed issue of using entity-based features for essay scoring, due to the subtle difference between coherence and organization.

7 Conclusion and Future Work

In this paper, we have studied the impact of deep discourse structures in the evaluation of text coherence by two approaches. In the first approach, we implemented a model with discourse role features derived from RST-style discourse parsing, which represents deep discourse structures, and compared it against our re-implemented Lin et al. (2011)'s model derived from PDTB-style parsing, with no deep discourse structures annotated. In the second approach, we compared our complete RST-style model against a model with shallow RST-style encoding. Evaluated on the two tasks, sentence ordering and essay scoring, deep discourse structures are shown to be effective for better differentiation of text coherence. Moreover, we showed that, even without deep discourse structures, shallow RST-style encoding is more powerful than PDTB-style encoding, because it has better coverage of discourse relations in texts. Finally, combining discourse relations with entity-based features is shown to have an inconsistent effect on the two evaluation tasks, which is probably due to the different nature of the two tasks.

In our future work, we wish to explore the effect of automatic discourse parsers in our methodology. As discussed previously, the PDTB- and RST-style discourse parsers used in our experiments are far from perfect. Therefore, it is possible that using automatically extracted discourse relations creates some bias to the training procedure; it is also possible that what our model actually learns is the distribution over those discourse relations which automatic discourse parsers are mostly confident with, and thus errors (if any) made on other relations do not matter. One potential way to verify these two possibilities is to study the effect of each particular type of discourse relation to the resulting model, and we leave it for future exploration.

Acknowledgements

We thank the reviewers for their valuable advice and comments. This work was financially supported by the Natural Sciences and Engineering Research Council of Canada and by the University of Toronto.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 42rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 141–148.

Jackie Chi Kit Cheung and Gerald Penn. 2010. Entity-based local coherence modelling using topological fields. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 186–195.

[5]The entire text is covered by the annotation produced by RST-style discourse parsing, while this is generally not true for PDTB-style discourse parsing.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*.

Micha Elsner and Eugene Charniak. 2011. Extending the entity grid with entity-specific features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pages 125–129.

Vanessa Wei Feng and Graeme Hirst. 2012a. Extending the entity-based coherence model with multiple ranks. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 315–324, Avignon, France.

Vanessa Wei Feng and Graeme Hirst. 2012b. Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 60–68, Jeju, Korea.

Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of The 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, Baltimore, USA, June.

Katja Filippova and Michael Strube. 2007. Extending the entity-grid coherence model to semantically related entities. In *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 2007)*, pages 139–142.

Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. 2009. *International Corpus of Learner English (Version 2)*. Presses universitaires de Louvain.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA.

Shafiq Joty, Giuseppe Carenini, and Raymond T. Ng. 2012. A novel discriminative framework for sentence-level discourse analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL 2012, pages 904–915.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011)*, Portland, Oregon, USA, June.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, 2:151–184.

William Mann and Sandra Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Isaac Persing, Alan Davis, and Vincent Ng. 2010. Modeling organization in student essays. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 229–239, Cambridge, MA, October. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse Treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*.

Lexical Chaining for Measuring Discourse Coherence Quality in Test-taker Essays

Swapna Somasundaran
Educational Testing Service
660 Rosedale Road
Princeton, NJ 08540
ssomasundaran@ets.org

Jill Burstein
Educational Testing Service
660 Rosedale Road
Princeton, NJ 08540
jburstein@ets.org

Martin Chodorow
Hunter College, CUNY
695 Park Avenue
New York, NY 10065
martin.chodorow@hunter.cuny.edu

Abstract

This paper presents an investigation of lexical chaining (Morris and Hirst, 1991) for measuring discourse coherence quality in test-taker essays. We hypothesize that attributes of lexical chains, as well as interactions between lexical chains and explicit discourse elements, can be harnessed for representing coherence. Our experiments reveal that performance achieved by our new lexical chain features is better than that of previous discourse features used for this task, and that the best system performance is achieved when combining lexical chaining features with complementary discourse features, such as those provided by a discourse parser based on rhetorical structure theory, and features that reflect errors in grammar, word usage, and mechanics.

1 Introduction

Coherence, the reader's ability to construct meaning from a document, is greatly influenced by the presence and organization of cohesive elements in the text (Halliday and Hasan, 1976; Moe, 1979). The lexical chain (Morris and Hirst, 1991) is one such element. It consists of a sequence of related words that contribute to the continuity of meaning based on word repetition, synonymy and similarity. In this paper we explore how lexical chains can be employed to measure coherence in essays. Specifically, our goal is to investigate how attributes of lexical chains can encode discourse coherence quality, such as adherence to the essay topic, elaboration, usage of varied vocabulary, and sound organization of thoughts and ideas. To do this, we build lexical chains and extract linguistically-motivated features from them. The number of chains and their properties, such as length, density and link strength, can potentially reveal discourse qualities related to focus and elaboration. In addition, features that capture the interactions between chains and explicit discourse cues, such as transition words, can show if the cohesive elements in text have been organized in a coherent fashion.

The main contributions of this paper are as follows: We use lexical chaining features to train a discourse coherence classifier on annotated essays from six different essay-writing tasks which differ in essay genre and/or test-taker population. We then perform experiments to measure the effect of the features when they are used alone and when they are combined with state-of-the-art features to classify the coherence quality of essays. Our results indicate that lexical chaining features yield better results than discourse features previously explored for this task and that the best performing feature combinations contain lexical chaining features. We also show that lexical chaining features can improve system performance across multiple genres of writing and populations. Our efforts result in the creation of a higher performing state-of-the-art feature set for measuring coherence in test-taker writing.

The rest of the paper is organized as follows: In Section 2, we describe our intuitions about lexical chains and how they can be used for measuring discourse coherence quality in essays. Section 3 describes our data, and Section 4 describes our experiments in predicting discourse coherence quality. We discuss related work in Section 5 and conclude in Section 6.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 950–961, Dublin, Ireland, August 23-29 2014.

2 Lexical Chains and Discourse Coherence Quality

According to Morris and Hirst (1991), *lexical cohesion* is the result of chains of related words that contribute to the continuity of lexical meaning. These sequences are characterized by the relations between the words, as well as by their distance and density within a given span. Lexical chains do not stop at sentence boundaries – they can connect a pair of adjacent words or range over an entire text. Morris and Hirst also observe that lexical chains tend to delineate portions of text that have a strong unity of meaning. In this paper, we use this underlying principle of cohesion to detect the *quality of coherence* in a discourse. Specifically, we employ lexical chains to quantify and represent expectations for coherent discourse in test-taker essays. Presumably, violations of these expectations would indicate lack of (or poor) coherence. We believe lexical chains have the potential to reveal the following characteristics about discourse coherence in essays:

Text unity: Textual continuity is vital for the reader's ability to construct meaning from the text (Halliday and Hasan, 1976). Coherent essays generally maintain focus over the main theme, so lexical chains constructed over such essays will have chains representing the central topic running through most of the length of the essay. These types of chains would presumably represent the main claim or position in persuasive texts, the main object or person in descriptive texts, and the main story-line in narrative texts. On the other hand, incoherent texts that jump from one topic to another, or do not adhere to a central idea, will exhibit no chains or chains with very few member words.

Elaboration and Detailing: A function of elaboration in discourse is to overcome misunderstanding or lack of understanding, and to enrich the understanding of the reader by expressing the same thought from a different perspective (Hobbs, 1979). Good writers usually initiate topics, ideas or claims and provide clear elaborations or reasons. That is, a sequence of many related words and phrases will be evoked to explain an idea or provide an account of the writer's reasoning. This development and detailing will be exhibited by lexical chains with a good number of member words.

Variety: While cohesiveness is vital for coherence, too much repetition of the same word can, in fact, harm the discourse quality (Witte and Faigley, 1981). Using a variety of words to express an idea or elaborate on a topic is generally a characteristic of skillful writing. Lexical chains corresponding to such writing will have a variety of similar words within the same chain.

Organization: In addition to cohesion (as represented by lexical chains in our case), one other factor must be present for text to have coherence – organization (Moe, 1979; Perfetti and Lesgold, 1977). Thus, it is important to organize ideas using clear discourse transitions. Transitions from one topic to another, or from a topic to its subtopic, should be clearly cued in order to assist the reader's understanding of the discourse. Consequently, in coherent writing, we would expect lexical chain patterns to synchronize with discourse cues. For example, we would expect some chains to start after a new (sub) topic initiation cue, such as "Secondly" or "Finally", and at least some chains (corresponding to the previous topic) to end immediately before the cue. Similarly, we would expect at least some chains to cross over discourse cues indicating elaboration or reason (e.g. "because") due to topic continuity.

2.1 Features for Measuring Discourse Coherence

In order to measure discourse coherence quality, we create features based on attributes of lexical chains extracted from essays. These features are then used to train a machine learning model, using essays manually labeled for overall discourse coherence quality.

2.1.1 Lexical Chain Construction

Lexical chains in a text are composed of words and terms that are related. Based on Hirst and St-Onge (1995), these relations can be exact repetitions, called *extra-strong relations*, close synonyms, called *strong relations*, or words with weaker semantic relations, called *medium-strong relations*. We implement the lexical chaining program described in Hirst and St. Onge (1995), where if a word or phrase is potentially chainable, it is considered a candidate *node* for existing chains. First, an extra-strong relation is sought throughout all existing chains, and if one is found, the word is added to it. If not, strong relations are sought, but for these, the search scope is limited to the words in any chain that is

no farther away than the previous six sentences in the text; the search ends as soon as a strong relation is found. Finally, if no relationship has yet been found, medium-strong relations are sought with the search scope limited to words in chains that are no farther away than the previous three sentences. If the node cannot be added to any existing chains, it forms its own single-node chain.

In this work, nouns are the focus of the lexical chains. Nouns, adjective-noun and noun-noun structures are identified as potential chain participants. Lin's thesaurus (Lin, 1998) is used to measure similarity between words and phrases. Candidate pairs receiving similarity scores greater than 0.8 are considered to have an extra-strong relationship (word repetition receives a similarity score of 1), pairs with similarity greater than 0.172 are considered to have a strong relation, and pairs with similarity scores greater than 0.099 are considered to have a medium-strong relation. These thresholds were chosen after qualitative inspection of a separate development data set of essays, and are also based on a previous finding (Burstein et al., 2012) that 0.172 is the mean similarity value across different parts of speech in the Lin thesaurus.

We created two feature sets to capture the intuitions described above. The first set, *LEX-1*, encodes the characteristics of text unity, elaboration and variety, while the second, *LEX-2*, encodes organization.

2.1.2 LEX-1 feature set

In order to capture text unity and detailing, we create features such as: *total number of chains in the essay*, *average chain size*, *number (and percentage) of large chains* (chains having more than four nodes are considered to be large chains[1]). As discussed previously, essays that show ample chaining might do so because they adhere to themes and their development, while the presence of large, dense chains might be an indicator that a topic is being discussed in detail. To represent variety, we employ features such as *number (and percentage) of chains that have a variety of words* (chains containing more than one word/phrase type are considered to have variety), as well as *number (and percentage) of large chains with a variety of words*. To encode the characteristics of cohesive relationships, we look at the nature of the links. Examples of these features are: *number and percentage of each link type*, *number (and percentage) of links of each type in large chains as well as in small chains*. Corresponding to each feature that uses counts (e.g. total number of chains) we also created normalized versions of the numbers to account for the essay length. LEX-1 has a total of 38 features.

2.1.3 LEX-2 feature set

LEX-2 features capture the interactions between discourse transitions, indicated by explicit discourse cues, and lexical chaining patterns. For this, we use a discourse cue tagger described in Burstein et al. (1998) that was specifically developed for tagging discourse cues in the essay genre. Using patterns and syntactic rules, the tagger automatically identifies words and phrases used as discourse cues, and assigns them a discourse tag. Each tag has a primary component, indicating whether an argument (or topic) is being initialized (*arg-init*) or developed (*arg-dev*), and a secondary component indicating the specific type of discourse initialization (e.g. *CLAIM*, *SUMMARY*), or development (e.g. *CLAIM*, *CONTRAST*). Examples of the discourse tags and their cues are: *arg-init:SUMMARY* (e.g. *all in all*, *in conclusion*, *in summary*, *overall*), arg-init:TRANSITION (e.g. *let us*), arg-init:PARALLEL (e.g. *firstly*, *similarly*, *finally*), arg-dev:CONTRAST (e.g. *nonetheless*, *however*, *on the contrary*, *rather than*, *even if*), arg-dev:EVIDENCE (e.g. *because of*, *since*), arg-dev:INFERENCE (e.g. *as a result*, *consequently*, *therefore*), arg-dev:DETAIL (e.g. *as well as*, *in this case*, *in addition*, *such as*), arg-dev:REFORMULATION (e.g. *in other words*, *that is*).

For each discourse cue tagged in the text, we replace the cue with its tag and measure the number of chains that (1) start after it, (2) end before it, and (3) continue over it (chains having nodes before and after the tag). We create such features for the tags in the original form (e.g. *arg-dev:DETAIL*), as well as for the primary component alone (e.g. *arg-dev*) and the secondary component alone (e.g. *DETAIL*). This alleviates the data sparseness that we see with certain tags, and results in a total of 138 tags for the LEX-2 feature set.

[1]This number was chosen after inspecting chains in a separate development data set.

3 Data

We use essays from different essay-writing tasks, representing different genres, writing proficiency and populations. Specifically, our essays consist of the following six subsets:

1. *PE-G-N*: **P**ersuasive/**E**xpository essays written by **g**raduate school applicants who are a mix of **n**ative and non-native speakers. (e.g. "As people rely more and more on technology to solve problems, the ability of humans to think for themselves will surely deteriorate. Discuss the extent to which you agree or disagree ... ") [n= 145 essays]

2. *AC-G-N*: **A**rgumentation **c**ritique essays written by **g**raduate school applicants who are a mix of **n**ative and non-native speakers. ("Examine the stated and/or unstated assumptions of the argument. Be sure to explain how the argument depends on the assumptions and what the implications are if the assumptions prove unwarranted ...") [n= 138 essays]

3. *PE-UG-NN*: **P**ersuasive/**E**xpository essays written by **u**ndergraduate and **g**raduate school applicants, who are **n**on-**n**ative speakers. [n= 146 essays]

4. *CS-UG-NN*: **C**ontrastive **s**ummary essays written by **u**ndergraduate and **g**raduate school applicants who are **n**on-**n**ative speakers. Here, the prompt focuses on a specific type of summarization, where ideas from an audio lecture are to be contrasted with ideas from a written passage. [n= 147 essays]

5. *S-G-N*: **S**ubject matter essays written by **g**raduates in a professional licensure exam who are a mix of **n**ative and non-native speakers. [n= 150 essays]

6. *M-K12-N*: A **M**ix of expository, persuasive, descriptive and narrative essays written by **K**-**12** school students who are a mix of **n**ative and non-native speakers. [n= 150 essays]

Of the total of 876 essays, 40 essays were used for system development, and the rest were used for cross-validation experiments. Each essay in the data set was manually annotated for overall discourse coherence quality by annotators not involved in this research. The discourse coherence score was assigned using a 4-point scale (with score point 4 for excellent discourse coherence). Twenty percent of the essays were double annotated and the rest were annotated by one of the annotators. Inter-annotator agreement over the doubly annotated essays, calculated using quadratic weighted kappa (QWK), was 0.61 (substantial agreement). The data distribution for each score point was: 1% for score 1, 9% for score 2, 27% for score 3, 63% for score 4.

4 Experiments

4.1 Baseline Features

A review by Burstein et al. (2013a) describes the several systems that measure discourse coherence quality across various text genres including test-taker essays. Features used to evaluate the discourse coherence quality systems in this study include those previously discussed in Burstein et al. (described below). In addition to comparing our features with previously explored features, our goal is to see if the state-of-the-art feature set can be extended with the use of lexical chaining features.

Entity-grid transition probabilities (entity). Entity-grid transition probabilities (Barzilay and Lapata, 2008) are intended to address unity, progression and coherence by tracking nouns and pronouns in text. An entity grid is constructed in which all entities (nouns and pronouns) are represented by their syntactic roles in a sentence (i.e., Subject, Object, Other). Entity grid transitions track how the same word appears in a syntactic role across adjacent sentences.

Type/Token Ratios for Entities (type/token). These are modified entity-grid transition probabilities. While the entity grid only captures, for example "Subject-Subject" transitions, type/token ratios capture the proportion of unique words that make such transitions. Higher ratios indicate that more concepts are being introduced in a given syntactic role, and lower ratios indicate fewer concepts.

RST-derived features (RST). Rhetorical relations (Mann and Thompson, 1988) derived from an RST parser (Marcu, 2000) are used to evaluate if and how certain rhetorical relations, combinations of rhetorical relations, or rhetorical relation tree structures contribute to discourse coherence quality. These include: (a) relative frequencies of n-gram rhetorical relations in the context of the RST parse tree structure (*unigrams*, or occurrences of a single relation (e.g., ThemeShift); *bigrams*, (e.g., "ThemeShift -> Elaboration"); and *trigrams*, (e.g., "ThemeShift -> Elaboration -> Circumstance")); (b) relative proportions of leaf-parent relation types in the tree structure; and (c) counts of root node relation types in the trees.

Maximum LSA Value for Distant Sentence Pairs (maxLSA). This feature set is the maximum Latent Semantic Analysis (LSA) similarity score found between pairs of sentences that are separated by at least 5 intervening sentences in the essay. It captures reintroduction of topics later in an essay, consistent with a backward inference strategy (Van den Broek et al., 1993; Van den Broek, 2012). LSA has also been employed to measure semantic relatedness between texts for discourse coherence (Foltz et al., 1998).

Grammatical error features (gramErr). These features address errors in grammar that could interfere with a reader's ability to construct meaning and have been used in previous studies (e.g. (Attali and Burstein, 2006; Burstein et al., 2013b)). Specifically, they are based on more than 30 kinds of errors in grammar, such as subject-verb agreement errors, in word usage, such as missing article errors, and in spelling. We use e-rater®, an essay scoring engine developed by Educational Testing Service (ETS), to detect the grammar errors. Aggregate counts of these errors are used as features for predicting discourse coherence.

Program Features (program). This is a single feature for identifying the data type listed in Section 3. Genre and population play an important role with respect to discourse coherence – essays written by more advanced writers, such as those at the graduate level, are typically more coherent than essays written by populations where English is a second language, or by K-12 school students. Note that the program feature is not linguistically motivated – it does not capture the writing construct or a writing skill. However, it is a strong feature as it can reliably bias the system to change its expectations about the discourse quality based on population and task.

4.2 Principal Components Analysis

To reduce the number of lexical chain features, a Principal Components Analysis (PCA) was calculated on an independent set of 6000 essays randomly sampled from the six task types. For 38 LEX-1 features, a 4-component solution accounted for about 0.70 of the feature variance. An 8-component solution explained about 0.30 of the feature variance for the 138 LEX-2 features. (While the variance was lower for this PCA solution, the components were fairly clean.) The component scores were then computed for the 876 essays in our annotated data set. The 4-component scores were used as LEX-1 features, and the 8-component scores were used as LEX-2 features. PCA was used for lexical chaining features in order to reduce the number of features used to build the models rather than using a much larger number of correlated features. PCA was not applied to features from previous work, as we wanted to reproduce their performance.

4.3 Results

A 10-fold cross-validation was run with an unscaled, gradient boosting regressor[2] tuned using quadratic weighted kappa[3]. Specifically, we used the standard Gradient Boosting Regressor in the scikit-learn toolkit[4] (Pedregosa et al., 2011). The learner was trained to assign 4-point coherence quality scores using different combinations of the feature sets described in sections 2.1 and 4.1. In addition to each of the individual features in Section 4.1, we tested two baseline feature combinations: *Baseline-1*, a system using all discourse-based features from Section 4.1, and *Baseline-2*, a system using all features described in Section 4.1.

[2]We experimented with SVMs and Random Forest learners too, but the best results were obtained with the regressor.

[3]The software for the regressor can be found at https://github.com/EducationalTestingService/skll/

[4]http://scikit-learn.org/stable/modules/generated/sklearn.ensemble

Performance was calculated using Quadratic Weighted Kappa (QWK) (Cohen, 1968), which measures the agreement between the system score and the human-annotated coherence score. QWK corrects for chance agreement, and it penalizes large disagreements more than small disagreements. The formula for QWK is as follows:

$$\kappa = 1 - \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} w_{ij}o_{ij}}{\sum_{i=1}^{k}\sum_{j=1}^{k} w_{ij}e_{ij}}$$

where k is the total number of categories (4 in our case), o_{ij} is the observed value in cell i, j of the confusion matrix between system predictions and human scores, e_{ij} is the expected value for cell i, j, and w_{ij} is the weight given to the discrepancy between $category_i$ and $category_j$. The expected value e_{ij} is calculated as:

$$e_{ij} = \frac{\sum_{j=1}^{k} o_{ij} \sum_{i=1}^{k} o_{ij}}{\sum_{i=1}^{k}\sum_{j=1}^{k} o_{ij}}$$

For quadratic weighted kappa, w_{ij} is calculated as:

$$w_{ij} = 1 - \frac{(i-j)^2}{(k-1)^2}$$

where i and j are categories, and k is the total number of categories. We use QWK as it is the standard evaluation metric used in automated essay scoring, and it also helps us to compare our results with previous work.

Table 1 reports the results for our proposed features and for each individual feature set investigated in previous work. Here, feature sets explicitly targeting discourse phenomena are grouped under *Discourse-based Features*. The features grouped under *Non-Discourse Features* also capture coherence quality; however they are based on grammatical errors or data type information. The best performing system in each group is shown in **bold**. We see that the full set of lexical chaining features (LEX-1 + LEX-2) is the best performing discourse-based feature set. It performs better than each of the other discourse-based features used alone, and also better than Baseline-1, which uses a combination of *all* discourse-based features from previous work. Notice that the performance of each discourse-based system is below the performance of both *gramErr* and *program*, indicating that they can play an important role in predicting text coherence.

While grammar (*gramErr*) and data type (*program*) are powerful features, it is also important to incorporate capabilities for detecting and evaluating discourse-specific phenomena to ensure construct relevance, as the grading guidelines for essays specify the need for proper organization of ideas (e.g."sustains a well-focused, well-organized analysis, connecting ideas logically"). Lack of construct relevance has been a major criticism of automated scoring methods (Deane, 2013; Shermis and Burstein, 2013). Additionally, discourse-relevant features will allow for interpretable, useful, explicit feedback to students regarding discourse coherence and its breakdown.

In Table 1 we also see that no individual discourse-based system outperforms Baseline-2, comprising all features from the state-of-the-art (Section 4.1). In fact, the human-system agreement obtained by Baseline-2 surpasses the human-human agreement (QWK of 0.61) reported in Section 3. This phenomenon is not uncommon in essay scoring. For example, Bridgeman et al. (2012) performed detailed analyses and found that across all test populations, human-automated system score correlations surpassed human-human score correlations.

Because the *gramErr* and *program* features contain information that is complementary to discourse-based features, we combined the discourse features, first with gramErr features, and then with gramErr+program features. Table 2 reports the results from these experiments. The best performing system for each column is in **bold**, and all features with QWK higher than Baseline-2 are in *italics*. Here,

Feature set	QWK
Discourse-based features	
LEX-1+ LEX-2	**0.316**
LEX-1	0.176
LEX-2	0.246
entity	0.249
type/token	0.178
RST	0.295
maxLSA	0.171
Baseline-1	0.302
Non-Discourse Features	
gramErr	**0.592**
program	0.387
Baseline-2	0.631

Table 1: Performance of individual feature sets.

Feature set	+gramErr	+gramErr +program
LEX-1+ LEX-2	0.608	*0.646*
LEX-1	0.611	*0.650*
LEX-2	0.577	***0.654***
entity	**0.621**	0.609
type/token	0.600	0.623
RST	0.612	*0.649*
maxLSA	0.592	*0.650*
gramErr+program	*0.644*	

Table 2: Performance (QWK), of individual discourse-based features when *gramErr* is added (column 2) and *gramErr* and *program* are added (column 3)

we see that, when combined with *gramErr+program*, the full set of lexical chaining features (LEX-1+LEX-2), as well as LEX-1 and LEX-2 individually, perform above Baseline-2. Surprisingly, we find that when some individual discourse features from previous work are combined with *gramErr+program*, they achieve better performance than Baseline-2 indicating that using the full combination of discourse features may not result in the best system. In the last row, we see that the combination of gramErr and program features alone (*gramErr+program*) is more competitive that Baseline-2, underscoring their usefulness for detecting coherence quality.

Finally, we performed full ablation studies to see which feature set combination produces the best system for identifying discourse coherence quality. Different combinations of the 8 feature sets resulted in 255 different systems, which we ranked based on their performance. Table 3 lists some of the systems, with their respective ranks and QWK values.

First, we observe that the best performing system contains the full set of lexical chaining features and achieves a QWK of 0.673. In fact, all of the top-5 performing systems contain either *LEX-1* or *LEX-2*. The best performance produced by a system not containing any lexical chaining features ranks eighth (*gramErr+ maxLSA+ program+ RST*). Notice that *gramErr+program* is at rank 31, Baseline-2 is at rank 61, and Baseline-1 is at rank 235. Interestingly, RST features are also seen in all of the top-5 systems, indicating that RST features and lexical chaining features capture complementary information about discourse quality. Surprisingly, maxLSA features, which have the same underlying principle of cohesion in text as lexical chains, are in some of the top-performing feature combinations (at ranks 4 and 5), indicating that, in addition to how ideas and themes are presented throughout the essay, the re-introduction of topics is also important.

We tested the statistical significance of the performance differences between our best system (*gramErr + LEX-2+ LEX-1+ maxLSA+ program+ RST*, at rank 1 in Table 3) and three other systems (Baseline-1, Baseline-2 and *gramErr+program*) by drawing 10,000 bootstrap samples (Berg-Kirkpatrick et al., 2012) from our manually scored essays. For each sample, QWKs were calcuated between the human scores and the predictions of our best system, and between the human scores and each of the other three systems' predictions. For each sample, the differences in QWKs were recorded, and the distributions of differences were used for significance testing. Results show that our best performing system is significantly better than Baseline-1 ($p < 0.001$) and Baseline-2 ($p < 0.01$), and it marginally outperformed the system with *gramErr+program* features ($p < 0.06$).

These results show that lexical chaining information is a reliable indicator of discourse quality, and that it can be combined synergistically with other complementary features to extend the state-of-the-art for measuring discourse coherence quality.

Feature set	QWK	Rank
gramErr + LEX-2+ LEX-1+ maxLSA+ program+ RST	**0.673**	1
gramErr+ LEX-1+ program+ RST	0.661	2
gramErr+ LEX-2+ program+ RST	0.661	3
gramErr+ LEX-2+ maxLSA+ program+ RST	0.660	4
gramErr+ LEX-1+ maxLSA+ program+ RST	0.659	5
gramErr+ maxLSA+ program+ RST	0.656	8
gramErr+ program	0.644	31
Baseline-2: entity+ gramErr+ RST+ maxLSA+ program+ type/token	0.631	61
Baseline-1: entity+ RST+ maxLSA+ type/token	0.302	235

Table 3: Performance (QWK), and ranks of systems using different feature combinations

4.4 Analysis by Data Type

In the previous section we saw that features based on lexical chaining are able to successfully encode and predict the quality of discourse coherence. We now examine if this impact is uniform across all essay genres and populations of writers. Table 4 shows the performance of *gramErr +program* (in column 2), the best performing features and their respective performance (**Best system**, columns 3 and 4), and the best feature set when lexical chaining features are removed, with their respective performance (**Best Minus LEX-1 and LEX-2**, columns 5 and 6). Here we use *gramErr +program* as an additional baseline, as it was found to be more competitive than both Baseline-1 and Baseline-2.

Program	gramErr +prog	Best system		Best Minus LEX-1 and LEX-2	
		Features	QWK	Features	QWK
CS-UG-NN	0.418	gramErr+ maxLSA+ RST	**0.523**	gramErr+ maxLSA+ RST	**0.523**
PE-UG-NN	0.406	gramErr + LEX-2 + maxLSA + RST	**0.468**	gramErr	0.406
PE-G-N	0.614	gramErr + LEX-1 + maxLSA	**0.676**	gramErr + maxLSA	0.650
AC-G-N	0.744	gramErr + LEX-2 + maxLSA	**0.839**	gramErr + maxLSA + type/token	0.766
S-G-N	0.414	entity + gramErr+ LEX-1+ RST+ type/token	**0.532**	gramErr+ RST+ type/token	0.487
M-K12-N	0.635	gramErr + LEX-2 + maxLSA	**0.656**	gramErr + maxLSA + RST + type/token	0.649

Table 4: Performance of feature sets by data type. Best performance is shown in **bold**.

In general, for all data types, addition of discourse features produces improvement over just using a combination of *gramErr* and *program* features. Also, the addition of lexical chaining features produces performance improvement for most data types. Specifically, there is substantial improvement in performance for persuasive writing (PE-UG-NN and PE-G-N), expository subject writing (S-G-N) and writing involving critical argumentation (AC-G-N). M-K12-N, which is composed of a mix of genres and writing proficiency, shows a minor improvement. Interestingly, for contrastive summarization (CS-UG-NN), the best system for predicting discourse coherence does not employ any lexical chaining features. For this type of writing, the best feature set using lexical chaining features achieved a QWK of 0.465, which improves over *gramErr+program* but is lower than the best performing feature set. This is perhaps because the discourse phenomena targeted by our lexical chaining features (topical detailing, variety and organization) are already provided for the writer in the source document and the audio lecture, i.e., the materials that are to be referred to in writing this type of essay. Thus, other features play a more prominent role, such as the RST features that capture local discourse organization which is needed, for example, when drawing a contrast between two sources of conflicting information.

5 Related Work

5.1 Discourse coherence quality

A number of models for measuring the quality of discourse coherence have been based on Centering Theory (Grosz et al., 1995). For example, Barzilay and Lapata (2008) construct entity grids based on syntactic subjects and objects. Their algorithm keeps track of the distribution of entity transitions between adjacent sentences and computes a value for all transition types based on their proportion of occurrence in a text. The algorithm has been evaluated with three tasks using well-formed newspaper corpora: text ordering, summary coherence evaluation, and readability assessment. Along similar lines, Rus and Niraula (2012) find centered paragraphs based on prominent syntactic roles. Similarly, Miltsakaki and Kukich (2000) use manually marked centering information and find that higher numbers of Rough Shifts within paragraphs are indicative of a lack of coherence. Using well-formed texts, Pitler and Nenkova (2008) show that a text coherence detection system yields the best performance when it includes features using the Barzilay and Lapata (2008) entity grids, syntactic features, discourse relations from the Penn Discourse Treebank (Prasad et al., 2008), and vocabulary and length features. Wang, Harrington, and White (2012) combine the approaches from Barzilay and Lapata (2008), and Miltsakaki and Kukich (2000) to detect coherence breakdown points. The biggest difference between our approach and the approaches based on Centering Theory is that we do not use syntactically prominent items or try to establish a center. Instead, multiple concurrent thematic chains can "flow" through the paragraph, and their length, density, and interaction with discourse markers are used to model coherence.

In other related work, Lin et al. (2011) use discourse relations from Discourse Lexicalized Tree Adjoining Grammar (D-LTAG) and compile sub-sequences of discourse role transitions to see how the discourse role of a term varies through the progression of the text. Our work, in contrast, traces how chains or thematic threads are organized with respect to the discourse. Our approach also differs from models that measure local coherence between adjacent sentences (Foltz et al., 1998), in that lexical chains can run though the length of the entire text, and hence the features derived from them are able to capture aggregate thematic properties of the entire text such as number, distribution and elaboration of topics.

Discourse coherence models have been previously employed for the task of information-ordering in well-formed texts (e.g., (Soricut and Marcu, 2006; Elsner et al., 2007; Elsner and Charniak, 2008)). In our tasks, discourse coherence quality is influenced by many factors including, but not limited to, ordering of information, such as text unity, detailing and organization.

Higgins et al. (2004) implemented a genre-dependent system to predict discourse coherence quality in essays. Their approach, however, was reliant on organizational structures particular to expository and persuasive essays, such as thesis statement and conclusion.

5.2 Lexical Chaining and Cohesion

Lexical chaining has been used in a number of applications such as news segmentation (Stokes, 2003), question-answering (Moldovan and Novischi, 2002), summarization (Barzilay and Elhadad, 1997), detection and correction of malapropisms (Hirst and St-Onge, 1995), topic detection (Hatch et al., 2000), topic tracking (Carthy and Sherwood-Smith, 2002), and keyword extraction (Ercan and Cicekli, 2007).

In a closely related study, Feng et al. (2009) use lexical chains to measure readability. Lexical chain features are employed to indicate the number of entities/concepts that a reader must keep in mind while reading a document, and two of their features (number of chains in the document and average length of chains) overlap with our LEX-1 features. Our work also differs from systems using cohesion to measure writing quality (e.g., (Witte and Faigley, 1981; Flor et al., 2013)), in that we focus on predicting the quality of discourse coherence.

6 Conclusion

In this paper, we investigated the use of lexical chaining for measuring discourse coherence quality. Based on intuitions about what makes a text coherent, we extracted two sets of features from lexical chains, one encoding how topical themes and cohesive elements are addressed in the text, and another

encoding how the topical themes interact with explicit discourse organizational cues. We performed detailed experiments which showed that lexical chaining features are useful for predicting discourse coherence quality. Specifically, when compared to other previously explored discourse-based features, we found that our lexical chaining features are best performers when used alone. We then experimented with various feature combinations and showed that top performing systems contain lexical chaining features. Our analyses also indicated that lexical chaining features can improve performance on various genres of writing by different populations of writers. Our future work on measuring discourse coherence quality involves extending chains by using verb information and by exploring finer distinctions within the chains themselves (e.g., topical and sub-topical chains).

References

Yigal Attali and Jill Burstein. 2006. Automated essay scoring with e-rater v. 2.0. *Journal of Technology, Learning, and Assessment*, 4:3.

Regina Barzilay and Michael Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL workshop on intelligent scalable text summarization*, volume 17, pages 10–17.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An empirical investigation of statistical significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005. Association for Computational Linguistics.

Brent Bridgeman, Catherine Trapani, and Yigal Attali. 2012. Comparison of human and machine scoring of essays: Differences by gender, ethnicity, and country. *Applied Measurement in Education*, 25(1):27–40.

Jill Burstein, Karen Kukich, Susanne Wolff, Ji Lu, and Martin Chodorow. 1998. Enriching automated essay scoring using discourse marking. In *Workshop on Discourse Relations and Discourse Marking*. ERIC Clearinghouse.

Jill Burstein, Jane Shore, John Sabatini, Brad Moulder, Steven Holtzman, and Ted Pedersen. 2012. The language musesm system: Linguistically focused instructional authoring. Technical report, Educational Testing Services (ETS).

Jill Burstein, Joel Tetreault, and Martin Chodorow. 2013a. Holistic discourse coherence annotation for noisy essay writing. *Dialogue and Discourse*, 4(2):34–52.

Jill Burstein, Joel Tetreault, and Nitin Madnani, 2013b. *Handbook of Automated Essay Evaluation: Current Applications and New Directions*, chapter The E-rater Automated Essay Scoring System. Routledge.

Joseph Carthy and Michael Sherwood-Smith. 2002. Lexical chains for topic tracking. In *2002 IEEE International Conference on Systems, Man and Cybernetics*, volume 7. IEEE.

Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70(4):213.

Paul Deane. 2013. On the relation between automated essay scoring and modern views of the writing construct. *Assessing Writing*, 18(1):7 – 24. Automated Assessment of Writing.

Micha Elsner and Eugene Charniak. 2008. Coreference-inspired coherence modeling. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 41–44. Association for Computational Linguistics.

Micha Elsner, Joseph Austerweil, and Eugene Charniak. 2007. A unified local and global model for discourse coherence. In *Proceedings of NAACL/HLT*.

Gonenc Ercan and Ilyas Cicekli. 2007. Using lexical chains for keyword extraction. *Information Processing & Management*, 43(6):1705–1714.

Lijun Feng, Noémie Elhadad, and Matt Huenerfauth. 2009. Cognitively motivated features for readability assessment. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 229–237. Association for Computational Linguistics.

Michael Flor, Beata Beigman Klebanov, and Kathleen M. Sheehan. 2013. Lexical tightness and text complexity. In *Proceedings of the Workshop on Natural Language Processing for Improving Textual Accessibility*, pages 29–38, Atlanta, Georgia, June. Association for Computational Linguistics.

Peter W Foltz, Walter Kintsch, and Thomas K Landauer. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse processes*, 25(2-3):285–307.

Barbara J Grosz, Scott Weinstein, and Aravind K Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Michael AK Halliday and Ruqaiya Hasan. 1976. *Cohesion in english*. English Language Series. Longman Group Ltd.

Paula Hatch, Nicola Stokes, and Joe Carthy. 2000. Topic detection, a new application for lexical chaining. In *the proceedings of BCS-IRSG*, pages 94–103.

Derrick Higgins, Jill Burstein, Daniel Marcu, and Claudia Gentile. 2004. Evaluating multiple aspects of coherence in student essays. In *HLT-NAACL*, pages 185–192.

Graeme Hirst and David St-Onge. 1995. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, 305:305–332.

Jerry R Hobbs. 1979. Coherence and coreference. *Cognitive Science*, 3(1):67–90.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 997–1006. Association for Computational Linguistics.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 2000. *The theory and practice of discourse parsing and summarization*. MIT Press.

Eleni Miltsakaki and Karen Kukich. 2000. Automated evaluation of coherence in student essays. In *Proceedings of LREC 2000*.

Alden J Moe. 1979. Cohesion, coherence, and the comprehension of text. *Journal of Reading*, 23(1):16–20.

Dan Moldovan and Adrian Novischi. 2002. Lexical chains for question answering. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):21–48.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Charles A Perfetti and Alan M Lesgold. 1977. *Discourse Comprehension and Sources of Individual Differences.* ERIC.

Emily Pitler and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 186–195. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The Penn Discourse TreeBank 2.0. In *LREC*.

Vasile Rus and Nobal Niraula. 2012. Automated detection of local coherence in short argumentative essays based on centering theory. In *Computational Linguistics and Intelligent Text Processing*, pages 450–461. Springer.

Mark D Shermis and Jill Burstein. 2013. *Handbook of automated essay evaluation: Current applications and new directions*. Routledge.

Radu Soricut and Daniel Marcu. 2006. Discourse generation using utility-trained coherence models. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 803–810. Association for Computational Linguistics.

Nicola Stokes. 2003. Spoken and written news story segmentation using lexical chains. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Proceedings of the HLT-NAACL 2003 student research workshop-Volume 3*, pages 49–54. Association for Computational Linguistics.

Paul Van den Broek, Charles R Fletcher, and Kirsten Risden. 1993. *Investigations of inferential processes in reading: A theoretical and methodological integration*. Taylor & Francis.

Paul Van den Broek. 2012. Individual and developmental differences in reading comprehension: Assessing cognitive processes and outcomes. *Measuring up: Advances in how we assess reading ability*, page 39.

Y Wang, M Harrington, and P White. 2012. Detecting breakdowns in local coherence in the writing of Chinese English learners. *Journal of Computer Assisted Learning*, 28(4):396–410.

Stephen P Witte and Lester Faigley. 1981. Coherence, cohesion, and writing quality. *College composition and communication*, pages 189–204.

Improving Cloze Test Performance of Language Learners Using Web N-Grams

Martin Potthast Matthias Hagen Anna Beyer Benno Stein
Bauhaus-Universität Weimar, Germany
<first name>.<last name>@uni-weimar.de

Abstract

We study the effectiveness of search engines for common usage, a new category of search engines that exploit n-gram frequencies on the web to measure the commonness of a formulation, and that allow their users to submit wildcard queries about formulation uncertainties often encountered in the process of writing. These search engines help to resolve questions on common prepositions following verbs, common synonyms in given contexts, and word order difficulties, to name only a few. Until now, however, it has never been shown that search engines for common usage have a positive impact on writing performance.

Our contribution is a large-scale user study with 121 participants using the Netspeak search engine to shed light on this issue for the first time. Via carefully designed cloze tests we show that second language learners who have access to a search engine for common usage significantly and effectively improve their test performance as opposed to not using them.

1 Introduction

When writing texts in a second language, uncertainties on specific formulations regularly come up. Even experienced second language writers may sometimes be in doubt about the preposition following a verb or what word order to choose. In this paper, we study search engines for common usage (usage search engines, for short) that aim at assisting second language writers to cope with their uncertainties. These search engines allow for phrasal queries that include wildcards at positions where a user is not sure what to write. The search results typically consist of a list of phrases matching the query's expression—the wildcards filled with formulations. The returned phrases are ranked by their commonness of being used in everyday writing, where a phrase's commonness is estimated by its occurrence frequency in a collection of web n-grams. The occurrence frequencies are usually not hidden from the user but displayed alongside each phrase, either implicitly or explicitly. This way, the users of usage search engines have a way of judging whether a phrase is commonly used by others. Figure 1 (left) shows an example search result.

Target audience of usage search engines is language learners who have mastered basic vocabulary and grammar but whose language proficiency in terms of their feeling for language usage is still worse than that of a native speaker. Until recently, there has been hardly any technological support for them, so they could only resort to studying abstract style guides, consuming foreign language media, and language study travels in order to improve their usage skills. Today, three public usage search engines are available. The first one, called Netspeak (Stein et al., 2010), is developed at our group since 2008. It was followed by PhraseUp and Linggle (Boisson et al., 2013), which have been released in 2011 and 2013.[1] Moreover, there is Google's N-Gram Viewer prototype (Michel et al., 2011), which has a different purpose and target audience but visualizes n-gram usage over time.

All of these search engines provide a way to quantify the commonness of a phrase and thus have the potential to become important tools for second language learners. That is, if they work as advertised. Until now, it has not at all been clear whether writers can actually benefit from the information distilled from analyzing n-gram occurrence frequencies, or whether they are easily misled, for example, by noisy

[1] Netspeak is freely available at www.netspeak.org, PhraseUp at www.phraseup.com, and Linggle at www.linggle.com.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 962–973, Dublin, Ireland, August 23-29 2014.

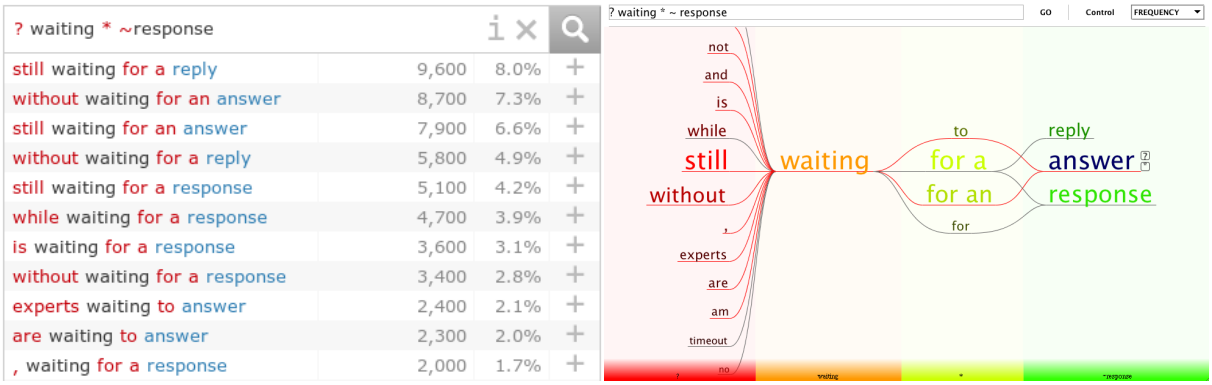


Figure 1: Netspeak's two alternative interfaces: search results can either be displayed as textual ranked list of phrases alongside frequencies (left), or as WordGraph visualization (right) (Riehmann et al., 2011), where the frequencies determine various aspects of the visualization. The WordGraph is particularly suited to handling multiple wildcards per query. The participants of our user study used primarily the textual interface, since they did not require more than one or two wildcards for solving the cloze tests.

data. Our contribution is to shed light on this issue for the first time and to conduct a large-scale user study with 121 language learners aged 14–18, measuring their performance when using our Netspeak search engine to solve cloze tests. The study ascertains the positive impact of Netspeak and by extension, usage search engines in general; moreover, it shows the low barrier to entry of Netspeak's user interface.

The paper is organized as follows: after a detailed discussion of related work in Section 2, Netspeak's retrieval engine is formally described in Section 3 as background for the design of our user study and as an example of how such search engines work internally. Section 4 reports on our user study and provides a statistical analysis of our findings. The paper closes with a conclusion and an outlook into future work.

2 Related Work

Carrying out research and development on usage search engines is an interdisciplinary effort that requires expertise from information retrieval, information visualization and interface design, as well as domain knowledge from computer linguistics. Therefore, we divide our review of related work into four parts: (1) existing search engines and web services, (2) retrieval engines and wildcard search from the perspective of information retrieval, (3) search result visualization, and, (4) writing support systems dedicated to second language writers.

2.1 Public Search Engines and Web Services

There are currently three public search engines and one public prototype that fall into the category of search engines for common usage, namely Netspeak (Stein et al., 2010), PhraseUp, Linggle (Boisson et al., 2013), and the Google N-Gram Viewer (Michel et al., 2011). All of them index large n-gram corpora, and their search interfaces are primarily dedicated to returning results that allow their users to judge the commonness of a phrase compared to alternative phrases. We distinguish the former three search engines from the latter mainly by its target audience. While the former target average web users, the latter targets professional linguists and humanities researchers. To the best of our knowledge, our paper is the first to investigate the effectiveness of such search engines for the use case of assisting writers, thereby underpinning these efforts.

Moreover, a number of other linguistic search engines are available, such as WebCorp Live (Kehoe and Renouf, 2002), WebAsCorpus (Fletcher, 2007), and the Linguist's Search Engine (Resnik and Elkiss, 2005). These search engines cannot be readily used for usage search as defined above, since they work more like concordancers in that they only retrieve usage examples and present them in context, disregarding usage commonness. Again, their target audience is professional linguists rather than laymen users, let alone second language learners. While they may still be applied in the context of language learning, the search interfaces of these search engines are not sufficiently tailored to this domain.

Another category of related web services that are readily available to second language learners include style and grammar checkers, such as Grammarly, PaperRater, SlickWrite, AfterTheDeadline (Mudge, 2010), the Hemingway App, GrammarBase, etc. From what can be said by analyzing their features, all of these services are based on a collection of basic style and grammar rules that can be checked automatically with some degree of confidence in their recommendations. However, none of the services we found make any recommendations with regard to usage commonness, i.e., they do not identify uncommon formulations or make recommendations for more common ones.

2.2 Information Retrieval Models and Indexes for Wildcard Search

The retrieval models employed by usage search engines are hardly ever discussed in the literature cited above. One of the few exceptions is Netspeak (Stein et al., 2010), where the retrieval model has been a contribution in itself since it is tailored specifically to its application domain. For the lack of discussion of the finer details of how the above search engines work, it can be assumed that they do not employ a specifically tailored retrieval approach. Nevertheless, when reviewing the information retrieval literature for retrieval models that support linguistic queries or wildcard queries, a number of sources can be found.

Cafarella et al. (2005, 2007) study indexing methods that are particularly suited to support queries comprising parts-of-speech as wildcards. They introduce so-called neighborhood indexes whose disk accesses required to answer a query are on the order of the number of non-wildcard terms in a query. Rafiei and Li (2009) develop a wildcard search engine that supports linguistically rich wildcards in order to support information extraction from the web, which employs a preprocessor for queries, and a postprocessor for search results on top of a traditional web search engine. The approach does not create a tailored index but translates the wildcard queries into flat queries that can be answered by traditional search engines. Sekine (2008) explores the trie data structure as an alternative to inverted indexes when indexing large-scale n-gram corpora. The approach is limited to short n-grams ($n < 10$) to be feasible, which can be a strong point in terms of retrieval speed. Netspeak's retrieval engine is also intentionally restricted to small values of n, but uses minimal perfect hash functions instead of tries to maximize retrieval performance.

While all of the aforementioned approaches support shallow linguistic wildcards, or only basic wildcards, Tsang and Chawla (2011) propose a method to support regular expressions. Doing so involves various trade-offs between retrieval performance and index size. Further, a search engine like this may be only useful to experts, but not second language learners. Again, all of the aforementioned contributions target either professional linguists or they are meant to facilitate automatic usage, instead of supporting average writers.

2.3 Visualization of Usage Search Results

An important part of every search engine is its user interface. Since usage search engines are still in their infancy, their user interfaces have not been studied in-depth, so far. As a first attempt to close this gap, we developed and analyzed two alternative user interfaces for Netspeak in a previous work, one textual interface and one using a tailored visualization that was specifically developed for usage search engines, the so-called WordGraph (Riehmann et al., 2011). Figure 1 shows them side-by-side. The textual interface displays search results in the form of a tabular list, where each row lists an n-gram matching the wildcard query alongside its absolute and relative occurrence frequency. If a query comprises more than one wildcard, situations arise where this linear ranking of n-grams is insufficient to grasp the true distribution of formulations that may be used instead of the wildcards. The WordGraph therefore visualizes the search results as a horizontal graph, so that the i-th word of an n-gram is displayed as a node on the i-th level of the graph. Paths from left to right through the graph correspond to n-grams found in the result set returned by Netspeak. A user study that investigated the fitness of the WordGraph to serve as a user interface for specific search tasks found that study participants prefer the WordGraph over the textual user interface when the number of wildcards increases (Riehmann et al., 2012). The user study we report on in this paper is based solely on the textual user interface, since most of our cloze tests can be solved by using one wildcard.

2.4 Writing Support for Second Language Learners

"*For writers of English as a Second Language (ESL), useful editorial assistance geared to their needs is surprisingly hard to come by*," and "[...] *there has been remarkably little progress in this area over the last decade*," observe Brockett et al. (2006) about the state of the art. This is despite the fact that English is the second language of most people who speak English today.[2] A recent overview of technology to detect grammatical errors of language learners is given by Leacock et al. (2010), whereas computer feedback for second language learners is mostly studied within pedagogical research under the label of computer-aided language learning (CALL). There, classroom systems are being deployed on a small scale to measure their effects on student learning performance. The development of usage search engines in general, our Netspeak engine in particular, and the user study contributed in this paper may be considered first steps toward the development of new, better technologies that specifically target the needs of second language learners and writers.

3 Netspeak: A Search Engine for Common Usage Based on Web N-Grams

As a background for our user study and as an example of how usage search engines work internally, this section briefly describes Netspeak and its retrieval engine.[3] The main building block of Netspeak is a query processor tailored to the following task: given a wildcard query q and a set D of n-grams, retrieve those n-grams $D_q \subseteq D$ that match the pattern defined by q. To solve this task, we have developed an index-based wildcard query processor addressing the three steps indexing, retrieval, and filtering, as illustrated in Figure 2 (middle).

3.1 Query Language

Netspeak utilizes a query language defined by the EBNF grammar shown in Figure 2 (left). A query is a sequence of literal words and wildcard operators, wherein the literal words must occur in the expression sought after, while the wildcard operators allow to specify uncertainties. Currently five operators are supported:

- the question mark (`?`), which matches exactly one word;
- the asterisk (`*`), which matches any sequence of words;
- the tilde sign in front of a word (∼`<word>`), which matches any of the word's synonyms;
- the multiset operator (`{<words>}`), which matches any ordering of the enumerated words; and,
- the optionset operator (`[<words>]`), which matches any one word from a list of options.

The textual interface displays the search results for the given query as a ranked list of phrases, ordered by decreasing absolute and relative occurrence frequencies. This way, the user can find confidence in choosing a particular phrase by judging both its absolute and relative frequencies. For example, a phrase may have a low relative frequency but a high absolute frequency, or vice versa, which in both cases indicates that the phrase is not the worst of all choices. Furthermore, the textual web interface offers example sentences for each phrase, which are retrieved on demand when clicking on a plus sign next to a phrase. This allows users who are still in doubt to get an idea of the larger context of a phrase.

3.2 Retrieval Engine

The indexing step is done offline. Let V denote the set of all words found in the n-grams D, and let $D^\wedge$ denote the set of integer references to the storage positions of the n-grams in D on hard disk. During indexing, an inverted index $\mu : V \rightarrow \mathcal{P}(D^\wedge)$ is built that maps each word $w \in V$ to a sorted list $\mu(w) \subseteq D^\wedge$, where $\mu(w)$ is comprised of exactly all references to the n-grams in D that contain w.

[2] `http://en.wikipedia.org/wiki/English_language#Geographical_distribution`

[3] Extended versions of this section can be found in previous publications on Netspeak's WordGraph visualization (Riehmann et al., 2011; Riehmann et al., 2012).

EBNF grammar of Netspeak's query language

query	=	{ word | wildcard }$_1^5$
word	=	([apostrophe] (letter { alpha })) | ","
letter	=	"a" | ... | "z" | "A" | ... | "Z"
alpha	=	letter | "0" | ... | "9"
apostrophe	=	"'"
wildcard	=	"?" | "*" | synonyms | multiset | optionset
synonyms	=	"~" word
multiset	=	"{" word { word } "}"
optionset	=	"[" word { word } "]"

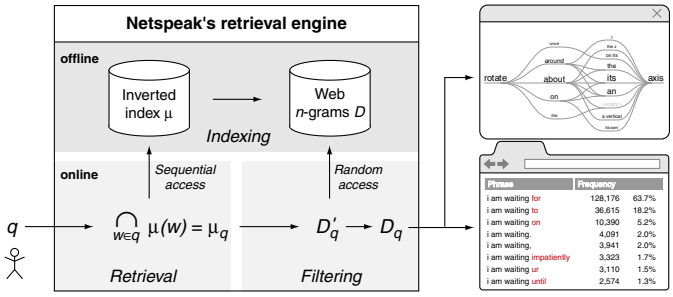


Figure 2: Netspeak at a glance (Riehmann et al., 2012): the left table shows Netspeak's query language as an EBNF grammar, the middle figure overviews its retrieval engine, and the right figure shows an example of search results as shown to its users. Given a query q, the intersection of relevant postlists yields a tentative postlist μ_q, which then is filtered and presented as a ranked list. The index μ exploits essential characteristics that are known a-priori about possible queries and the n-gram set D.

The list $\mu(w)$ is referred to as posting list or postlist. Since D is invariant, μ can be implemented as an external hash table with $O(1)$-access to $\mu(w)$. For μ being space-optimal, a minimal perfect hash function based on the CHD algorithm is employed (Belazzougui et al., 2009).

The two online steps, retrieval and filtering, are taken successively when answering a query q. Within the retrieval step, a tentative postlist $\mu_q = \bigcap_{w \in q} \mu(w)$ is constructed; μ_q is the complete set of references to n-grams in D that contain all words in q. The computation of μ_q is done in increasing order of postlist length, whereas each $\mu(w)$ is read sequentially from hard disk. Within the filtering step, a pattern matcher is compiled from q, and D_q is constructed as the set of those n-grams referenced in μ_q that are accepted by the pattern matcher. Constructing D_q requires random hard disk access. Basically, this approach corresponds to how web search engines retrieve documents for a given keyword query before ranking them. In what follows, we briefly outline how the search in D is significantly narrowed down.

With an inverted index that also stores specific n-gram information along with the keywords, the filtering of μ_q can be avoided. In this regard, we distinguish the queries that can be formulated with Netspeak's query language into two classes: fixed-length queries and variable-length queries. A fixed-length query contains only wildcard operators that represent an a-priori known number of words, while a variable-length query contains at least one wildcard operator that expands to a variable number of words. For example, the query `fine ? me` is a fixed-length query since only 3-grams in D match this pattern, while the query `fine * me` is a variable-length query since n-grams of length $2, 3, 4, \ldots$ match. Obviously, fixed-length queries can be answered with less filtering effort than variable-length queries: simply checking an n-gram's length suffices to discard many non-matching queries. The query processor first reformulates a variable-length query into a set of fixed-length queries, which then are processed in parallel, merging the results.

Moreover, the retrieval engine employs pruning strategies so that only relevant parts of a postlist are read during retrieval, presuming sorted postlists. Head pruning means to start reading a postlist at some entry within, without compromising recall. Given a query q, let τ denote an upper bound for the frequencies of the n-grams in q's result set D_q, i.e., $d \in D_q$ implies $f(d) \leq \tau$. Obviously, in all postlists that are involved within the construction of D_q, all entries whose n-gram frequencies are above τ can safely be skipped, whereas τ is determined in a preprocessing step as the lowest occurrence frequency of a sub-sequence of q that does not include wildcards. Up to this point, the retrieval of n-grams matching a query q is exact—but, not all n-grams that match a query are of equal importance. We consider this fact by applying tail pruning for postlists that are too long to be read at once into main memory. As a consequence, less frequent n-grams that might match a given query can be missed.

3.3 The Web n-Gram Collection

To provide relevant suggestions, a wide cross-section of written text on the web is required. Currently, Netspeak indexes the Google n-gram corpus "Web 1T 5-gram Version 1" (Brants and Franz, 2006),

which consists of 42 GB of phrases up to a length of $n = 5$ words along with their occurrence frequencies on the web in 2006. This corpus has been compiled from approximately 1 trillion words extracted from the English portion of the web, totaling more than 3 billion n-grams. Two post-processing steps were applied: case reduction and vocabulary filtering. For the latter, a white list vocabulary V was compiled and only n-grams whose words appear in V were retained. The vocabulary V consists of the words found in the Wiktionary and various other dictionaries, complemented by words from the 1-gram portion of the Google corpus whose occurrence frequency exceeds 10 000. After post-processing, the size of the corpus has been reduced by about 54%.

3.4 Retrieval Performance in Practice and Public Availability

In practice, the described techniques enable Netspeak to provide search results at a speed similar to modern web search engines. Results are usually returned within a couple of milliseconds. Whenever a user stops typing for more than 300 milliseconds, the current input is submitted as an "instant" query without need for a click. That way, the "search experience" with Netspeak is similar to what users expect from web search engines.

Netspeak is freely available online and has about 300 distinct users on a working day who submit about 2500 queries (half the workload on weekends). Most of its users are returning users. From their feedback and from our own experience, we know that Netspeak helps to resolve uncertainties on formulations in the daily process of writing papers, proposals, etc. However, in the following section we attempt to capture Netspeak's effectiveness in a controlled user study.

4 User Study on the Effectiveness of Usage Search Engines

It is generally assumed that usage search engines are useful, say, that they provide valuable feedback that leads to improved writing. To empirically confirm this "usefulness" assumption, we conduct systematic tests with experienced language learners and analyze whether a usage search engine enables them to improve their writing. We choose Netspeak as a representative of usage search engines for our study.

Our study's underlying rationale is to model the use case of usage search engines by solving cloze tests. In a cloze test, a word or a phrase is removed from a sentence and the participant has to replace the missing words. Although we followed standard procedures on constructing cloze tests (Sachs et al., 1997), it should be noted that our usage of cloze tests is not as originally intended (Taylor, 1953). We do not assess a language learner's reading skills, but use the cloze test to model word choice, which resembles the use case of usage search engines very well. For each participant, we provide two different cloze test questionnaires. The first has to be solved without any help, whereas for the second, participants are allowed to use the search engine. Besides evaluating the answers, we also analyze the submitted search queries.

4.1 Experiment 1: General Usage, Average Learners

In the first experiment, we examine whether the search engine in general can support users in resolving uncertainties on formulations modeled by cloze tests. Our hypothesis is that using a usage search engine helps to improve a human's performance in such tests.

Experimental Design To test our hypothesis, we conduct an empirical study with a within-subjects design (Lazar et al., 2010). This means that our participants are exposed to a cloze test without the help of a search engine and then to another cloze test where our chosen usage search engine is allowed.

The to-be-solved cloze tests are carefully constructed under the guidance of a university-level English teacher who is a native English speaker. From several language learner textbooks, we selected questions in order to have an equal mix of two easy, four medium, and three hard questions for two different cloze test questionnaires A and B (see Appendix A and B).

In order to have objectively comparable test cases, the English teacher provided four possible answers for each of the nine questions from test A and B, from which participants had to choose one in each case. This way, the participants do not have to rely on their subjective own vocabulary knowledge.

Table 1: Results of our user study on the impact of usage search engines on language learners.

Experiment	Question difficulty	Questions answered: manually ✓	manually ×	manually –	manually sum	with search engine available, but not used ✓	but not used ×	but not used –	and used ✓	and used ×	and used –	and used sum
Average Learners	easy	17	41	0	**58**	7	2	1	42	6	0	**58**
	medium	61	100	3	**164**	25	16	1	88	34	0	**164**
	hard	37	72	2	**111**	4	22	2	18	62	1	**111**
	all	115	213	5	**333**	36	40	4	149	102	1	**333**
Highly Experienced Learners	easy	11	5	0	**16**	10	1	0	4	1	0	**16**
	medium	27	17	0	**44**	24	2	0	14	3	1	**44**
	hard	18	12	0	**30**	8	8	0	4	10	0	**30**
	all	56	34	0	**90**	42	11	0	22	14	1	**90**
Specific Operators	easy	147	29	1	**177**	28	2	1	135	11	0	**177**
	medium	117	57	3	**177**	20	6	1	123	24	3	**177**
	hard	135	40	2	**177**	31	5	2	130	18	1	**177**
	all	399	126	6	**531**	79	13	4	378	53	4	**531**
		Search engine not used							**Search engine used**			

Experiment	Search engine used vs not used: p-value	effect size	
Average Learners	**0.0000**	0.73	large
Highly Exp. Learners	0.7030	0.12	small
Specific Operators	**0.0000**	0.58	large

In the left table, ✓ denotes correct answers, × denotes wrong answers, and – denotes unanswered questions.

To evaluate the statistical significance and the effect size, we distinguished cloze test answers for the conditions "Search engine not used" and "Search engine used" in the left table.

The brackets below the bottom row of the left table indicate which cases fall under what condition.

The English teacher first chose the questions independent of knowing the indexed n-grams of the search engine. In a "postprocessing" step, the chosen answers for the questions are checked for existence in the n-gram vocabulary of the search engine. This always was the case, although sometimes the queries required to retrieve them were different from the exact context around the cloze test's missing word. This check ensured that there was a chance of answering each individual question in the cloze tests with the search engine.

During the experiment, the use or non-use of the search engine is the independent variable. The dependent variable is the number of correct answers per questionnaire. There also are confounding variables like whether our engine really was used when it was allowed, the time needed to type queries, or the different numbers of answered questions with and without the search engine. We will further elaborate on how we deal with these variables in the following description of the experimental process.

Experimental Process From three different local high schools, 43 German pupils (23 female, 20 male; mean age 16.2, $SD = 1.2$) with five or more years of English courses participated in six groups. None of the participants had any previous experience with any usage search engine.

When a group arrived in our lab, they were randomly assigned to a lab seat; questionnaire A or B were distributed ensuring that neighboring participants had a different question set. This way, the test distribution was random and the participants could not collaborate (which was also ensured by their accompanying "watchdog" teachers). After seven minutes, the first questionnaires were collected and a short five minute introduction to the search engine and its operator set was given. To ensure that the pupils really followed the introduction, we provided the chance of winning small prices based on correctly answering a question on the underlying technique of usage search engines—the index—in an exit questionnaire. After that, each participant had to solve the opposite questionnaire (A when the first was B, and vice versa) but was allowed to use the search engine this time. In pilot studies, we noticed that pupils of that age often need a lot of time for typing their search queries on a standard keyboard. Thus, we allowed 10 minutes for the second questionnaire. This confounding variable of different timing for the questionnaires could not be avoided. Otherwise, most participants would not have had the chance to complete all questions. In order to check whether our participants actually used the search engine, we logged their querying behavior and manually identified the questions which they had answered without using the search engine.

Results and Discussion Since not all participants answered all questions for both cloze tests, we excluded the six participants from the following analyses, who had a difference of more than one between the number of answered questions for either test.

The aggregated numbers on questionnaire performance for the remaining 37 individuals are given in the first block of rows of Table 1 ("Average Learners"). Note that the ratio of correct vs. incorrect answers goes up when the search engine was used: on average, an individual answered two more questions correctly. Especially interesting is that the short five minute introduction was sufficient for that effect

which shows the strength of the textual interface. To statistically estimate the per-individual effect, we compare the ratio of correct answers among all answers when the search engine was used to the ratio when it was not used (note that this includes the questions where the engine was allowed but was not used; i.e., columns "manually" and "but not used" in Table 1). According to the Shapiro-Wilk test (Razali and Wah, 2011), the individual participants' ratios are not normally distributed for either condition (engine used vs. not used) such that we choose a non-parametric significance test (Lazar et al., 2010). For our within-subjects design with ratio data and two to-be-compared samples, the Wilcoxon signed rank test is known as a suitable significance test (Lazar et al., 2010). For the 37 participants' ratios we get a p-value below 0.001 and thus can reject the null hypothesis that the ratios' distributions are equal. Further estimating the effect size for the Wilcoxon signed rank test, we obtain a value of 0.73 which corresponds to a large effect (Cohen, 1988; Fritz et al., 2012). This result supports our prediction that the search engine can help resolve writing uncertainties.

We also studied the query logs of our participants. Per cloze test question, they submitted 4–5 queries with 2–3 terms on average (a wildcard is counted as a term). The last query in each such "search session" for a single question typically was 3–4 terms long. Almost all participants only used the `?`-operator and most participants chose the strategy of querying with context before and after the operator. Having only context before or only after the operator are less successful strategies with higher error ratios.

4.2 Experiment 2: General Usage, Highly Experienced Learners

In our neighborhood, there also is an international high school, where German pupils have all their classes taught in English. Obviously, such pupils have a much higher experience speaking and writing English than our participants from Experiment 1. For a second experiment, we invited pupils from the international school to our lab. Our hypothesis is that the pupils from the international school will have to use the search engine less frequently but still can benefit from it for individual questions.

Experimental Design and Process We used the same questionnaires, time constraints, and logging strategies as in Experiment 1. From the international school, 12 German pupils (7 female, 5 male; mean age 16.5, $SD = 0.7$) participated in two groups. These pupils are taught all their courses in English for five and more years. None of them had any previous experience with usage search engines. The experimental process was as in Experiment 1.

Results and Discussion Again, not all participants answered all questions for both cloze tests; we excluded the two participants from the following analyses, who had a difference of more than one between the number of answered questions for either test.

The aggregated numbers on questionnaire performance for the remaining 10 individuals are given in the second block of rows of Table 1 ("Highly Experienced Learners"). As expected, the highly experienced pupils used the search engine very rarely. This is not too surprising since our questionnaires were designed with an average German pupil in mind; many questions seemed too easy to the internationals which they also indicated in their exit questionnaires. Still, on a per-question basis, for the medium and difficult questions where the pupils used the search engine, they slightly improved their performance. However, the sample and the effect size are too small to draw any reliable conclusions.

The experiment shows that the highly experienced pupils indeed did not use our engine often. However, the predicted benefit for them cannot be confirmed from our small sample. It is thus an interesting open task to conduct a larger study with highly experienced users and more difficult questions.

4.3 Experiment 3: Specific Operators, Average Learners

Our first experiment revealed that most participants used the `?`-operator to solve the tasks. We thus designed a third experiment specifically targeted at the options, synonyms, and word-order operators of our Netspeak search engine. Our hypothesis is that each individual operator helps improve a human's performance in cloze tests targeted at the individual operator.

Experimental Design As in Experiment 1, we asked the university-level English teacher to design two cloze test questionnaires (see Appendix C and D); for each operator with an easy, a medium, and

a hard question. Here, the questions for the option operator are of a similar kind as the questions from Experiment 1. Four alternatives are given, but the participants are asked to use the option operator `[]` and not the `?`-operator. For synonyms, a complete sentence is given and for a specified word, the best among four given potential synonyms is requested. As for the word order operator, a two-word phrase is missing from the sentence and the two different word orders are provided as options. Like in Experiment 1 and 2, the explicit answer options ensure that the test is objective and not subjective. In a second development step, the questions were checked for solvability using the search engine just like in Experiment 1.

Experimental Process From three different local schools, 66 pupils (45 female, 21 male; mean age 15.9, $SD = 1.4$) participated in six groups. None of the pupils participated in Experiment 1 or 2 nor had they any previous experience with usage search engines. These pupils have learned English in their schools for at least five years. The schedule was similar to Experiment 1 with an emphasis on the three tested operators in the introductory explanations on Netspeak. In the questionnaires, the pupils were asked to use only the specific operator for the respective queries. Logging their queries, we are able to exclude solutions obtained by using a not-allowed operator.

Results and Discussion Again, not all participants answered all questions for both cloze tests; we excluded the seven participants from the following analyses, who had a difference of more than one between the number of answered questions for either test.

The aggregated numbers on questionnaire performance for the remaining 59 individuals are given in the third block of rows of Table 1 ("Specific Operators"). Note that the ratio of correct vs. incorrect answers goes up when the search engine was used: one to two more questions correctly answered on average. As in Experiment 1, the short five minute introduction is sufficient for that effect which shows the strength of our interface. To statistically estimate the per-individual effect, we compare the ratio of correct answers among all answers when the search engine was used to the ratio when it was not used (note that this includes the questions where the engine was allowed but was not used; i.e., columns "manually" and "but not used" in Table 1). For the 59 participants' ratios, we get a p-value below 0.001 and thus can reject the null hypothesis that the ratios' distributions are equal. Further, estimating the effect size for the Wilcoxon signed rank test, we obtain a value of 0.58 which corresponds to a large effect (Cohen, 1988; Fritz et al., 2012). Again, the result supports our prediction that usage search engines can help resolve writing uncertainties.

However, a deeper analysis reveals that the large effect is due to the synonym operator. Only for that operator, a statistically significant performance difference and a large effect size can be shown. For the other two operators, the null hypothesis of no performance difference cannot be rejected. This is in line with the exit questionnaire findings, where the pupils reported the synonym operator to be very helpful while the other questions were perceived as rather easy. In the query log analyses, we found that context before and after the wildcard had a similarly positive effect as before and was generally better than adding context only before the wildcard.

5 Conclusion and Future Work

Search engines for common usage have the potential to become an important tool for second language writers and learners. The possibility to check one's language against what is commonly written forms a unique opportunity to improve one's writing on-the-fly. Such information has not been available at scale so far. Our user study shows that usage search engines can indeed help second language writers solve uncertainties about formulations. Modeling writing uncertainties by carefully designed cloze tests, we are able to show a significant improvement when experienced language learners use the search engine.

Highly experienced language learners represented by our study participants from an international school, however, did not use the search engine often enough to draw meaningful conclusions. This can probably be attributed to the fact that the cloze tests were not tailored to their level of language proficiency. Therefore, the question of whether also highly experienced writers and learners, or even native speakers, can benefit from such search engines remains open and is left for future work.

Another missing piece in determining the effectiveness of usage search engines is whether their users

actually learn something while using them, or whether users frequently submit the same or similar queries again and again. Our user study was not designed to answer this question, since our participants were only around for about 30 minutes for organizational reasons. Even measuring effects on short-term memory is rendered infeasible in this time frame. A longitudinal study would be ideal, in this case, but we also see an exciting, data-driven way to approach this. By analyzing the query logs of Netspeak, which is currently being used hundreds of times per day, we can track returning users. We can then study their online search behavior to determine if and how often they return to submit similar queries, which allows us to draw conclusions about their learning success. More generally, the query logs of usage search engines may form a unique opportunity to observe language learners "in the wild" as opposed to the laboratory.

Finally, regarding the user interface of usage search engines, our user study has revealed ways to improve them. For example, the interface must be optimized for faster typing (especially on mobile devices) as we observed that the pupils were not adept to entering special characters on standard keyboards, which resulted in slow typing speed. Besides this, our user study also showed that the current state of Netspeak's textual user interface as well as the simplified wildcard query language is easy enough to be understood in less than a minute by any newcomer, which demonstrates the low barrier to entry that search engines for common usage have right now.

Acknowledgements

We thank the anonymous participants of our user study as well as Tim Gollub, Martin Trenkmann, Michael Völske, Howard Atkinson, Johannes Kiesel, Matthias Busse, and Alexander Herr for their help in organizing the user study.

References

Djamal Belazzougui, Fabiano C. Botelho, and Martin Dietzfelbinger. 2009. Hash, Displace, and Compress. In *Proceedings of ESA 2009*, pages 682–693.

Joanne Boisson, Ting-Hui Kao, Jian-Cheng Wu, Tzu-Hsi Yen, and Jason S. Chang. 2013. Linggle: A Web-scale Linguistic Search Engine for Words in Context. In *Proceedings of ACL 2013 (Demos)*, pages 139–144.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram Version 1. Linguistic Data Consortium LDC2006T13.

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL Errors Using Phrasal SMT Techniques. In *Proceedings of ACL 2006*, pages 249–256.

Michael J. Cafarella and Oren Etzioni. 2005. A Search Engine for Natural Language Applications. In *Proceedings of WWW 2005*, pages 442–452.

Michael J. Cafarella, Christopher Re, Dan Suciu, and Oren Etzioni. 2007. Structured Querying of Web Text Data: A Technical Challenge. In *Proceedings of CIDR 2007*, pages 225–234.

Jacob Cohen. 1988. *Statistical Power Analysis for the Behavioral Sciences*. Psychology Press.

William H. Fletcher. 2007. Implementing a BNC-Compare-able Web Corpus. In *Proceedings of the 3rd Web as Corpus Workshop*, pages 43–56.

Catherine O. Fritz, Peter E. Morris, and Jennifer J. Richler. 2012. Effect Size Estimates: Current Use, Calculations, and Interpretation. *Journal of Experimental Psychology: General*, 141(1):2.

Andrew Kehoe and Antoinette Renouf. 2002. WebCorp: Applying the Web to Linguistics and Linguistics to the Web. In *Proceedings of WWW 2002 (Posters)*.

Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. 2010. *Research Methods in Human-Computer Interaction*. Wiley Publishing.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Morgan and Claypool Publishers.

Jean-Baptiste Michel, Yuan K. Shen, Aviva P. Aiden, Adrian Veres, Matthew K. Gray, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez L. Aiden. 2011. Quantitative Analysis of Culture Using Millions of Digitized Books. *Science*, 331(6014):176–182.

Raphael Mudge. 2010. The Design of a Proofreading Software Service. In *Proceedings of HLT 2010 Workshop on Computational Linguistics and Writing*, pages 24–32.

Davood Rafiei and Haobin Li. 2009. Data Extraction from the Web Using Wild Card Queries. In *Proceedings of CIKM 2009*, pages 1939–1942.

Nornadiah Mohd Razali and Yap Bee Wah. 2011. Power Comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling Tests. *Journal of Statistical Modeling and Analytics*, 2(1):21–33.

Philip Resnik and Aaron Elkiss. 2005. The Linguist's Search Engine: An Overview. In *Proceedings of ACL 2005 (Posters and Demos)*, pages 33–36.

Patrick Riehmann, Henning Gruendl, Bernd Froehlich, Martin Potthast, Martin Trenkmann, and Benno Stein. 2011. The NETSPEAK WORDGRAPH: Visualizing Keywords in Context. In *Proceedings of PacificVis 2011*, pages 123–130.

Patrick Riehmann, Henning Gruendl, Martin Potthast, Martin Trenkmann, Benno Stein, and Bernd Froehlich. 2012. WORDGRAPH: Keyword-in-Context Visualization for NETSPEAK's Wildcard Search. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1411–1423.

J. Sachs, P. Tung, and R.Y.H. Lam. 1997. How to Construct a Cloze Test: Lessons from Testing Measurement Theory Models. *Perspectives*, 9:145–160.

Satoshi Sekine. 2008. A Linguistic Knowledge Discovery Tool: Very Large N-gram Database Search with Arbitrary Wildcards. In *Proceedings of COLING 2008 (Demos)*, pages 181–184.

Benno Stein, Martin Potthast, and Martin Trenkmann. 2010. Retrieving Customary Web Language to Assist Writers. In *Proceedings of ECIR 2010*, pages 631–635.

W. L. Taylor. 1953. Cloze Procedure: A New Tool for Measuring Readability. *Journalism Quarterly*, 30:415–433.

Dominic Tsang and Sanjay Chawla. 2011. A Robust Index for Regular Expression Queries. In *Proceedings of CIKM 2011*, pages 2365–2368.

Appendix

A Questionnaire A from Experiments 1 and 2

1. I really prefer just anything _____ watching television.
   × against ✓ to × about × on
2. Has Tony's new book _____ yet?
   ✓ come out × published × developed × drawn up
3. If this plan _____ off, I promise you you'll get the credit for it.
   × lets × goes × gets ✓ comes
4. Helen had great admiration _____ her history teacher.
   × in × to ✓ for × on
5. I just couldn't _____ over how well the team played!
   ✓ get × turn × make × put
6. The problem stems _____ the government's lack of action.
   × out ✓ from × under × for
7. It's too late to phone Jill at work, at any _____ .
   × case × time × situation ✓ rate
8. I'm afraid I'm not very good _____ children.
   × about × for ✓ with × at
9. We are _____ no obligation to change goods which were not purchased here.
   × with ✓ under × to × at

B Questionnaire B from Experiments 1 and 2

1. Don't worry about the lunch. I'll _____ to it.
   × look × prepare × care ✓ see
2. I am afraid that these regulations have to be _____ with.
   × provided ✓ complied × faced × met
3. Our thoughts _____ on our four missing colleagues.
   × based ✓ centred × laid × depended
4. Carol doesn't have a very good relationship _____ her mother.
   ✓ with × at × for × to

5. It seems to be your boss who is _____ fault in this case.
   × under × with ✓ at × for
6. Being rich doesn't count _____ much on a desert island.
   × on × to × of ✓ for
7. The policeman _____ me off with a warning as it was Christmas.
   × sent × gave ✓ let × set
8. Tina is an authority _____ Byzantine architecture.
   ✓ on × for × with × in
9. I was _____ the impression that you liked Indian food.
   × at × with × of ✓ under

C Questionnaire A from Experiment 3

Choose the word which fits best using the options operator `[<words>]`.

1. If you spend so much money every day, you will _____ out of money before the end of the month.
   × pay × use ✓ run × take
2. You need to take _____ all your other clothes before you put on your swimming costume.
   × down × away × out ✓ off
3. I'm afraid I'm not very good _____ history.
   × about × for ✓ at × with

Choose the best synonym for the underlined word using the synonym operator ∼`<word>`.

4. I love studying geometry the most.
   × hate × absent ✓ enjoy × difficult
5. My ambition is to become a computer scientist.
   × thought × reward × study ✓ dream
6. Your action will have serious consequences.
   ✓ effects × events × reasons × affects

Choose the correct word order using the word order operator `{<words>}`.

7. The _____ bird! I'm going to help it!
   ✓ poor little × little poor
8. She was wearing a _____ dress.
   × green beautiful ✓ beautiful green
9. I plan on wearing my _____ coat.
   ✓ long black × black long

D Questionnaire B from Experiment 3

Choose the word which fits best using the options operator `[<words>]`.

1. Sometimes Julia speaks very quickly so the other students have to ask her to slow _____ .
   ✓ down × up × out × off
2. The missing plane has apparently disappeared without a _____ .
   × sign × news × word ✓ trace
3. When Gabriel's credit card stopped, he cut it _____ many small pieces.
   × out ✓ into × apart × in

Choose the best synonym for the underlined word using the synonym operator ∼`<word>`.

4. I choose to study the differences between alligators and crocodiles.
   × make × buy ✓ prefer × wash
5. I cannot find my money. Can you get me my billfold?
   ✓ wallet × pocket × watch × bag
6. This is a very rough environment for elephants to live in.
   ✓ harsh × abrasive × coarse × beneficial

Choose the correct word order using the word order operator `{<words>}`.

7. She sold the _____ chairs at a yard sale.
   × wooden old ✓ old wooden
8. The _____ years were fantastic.
   × two first ✓ first two
9. It's close to the _____ building.
   ✓ big blue × blue big

A Framework for Translating SMS Messages

Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Ron Shacham
AT&T Labs
1 AT&T Way, Bedminster, NJ 07921
vkumar,jchen,srini,rshacham@research.att.com

Abstract

Short Messaging Service (SMS) has become a popular form of communication. While it is predominantly used for monolingual communication, it can be extremely useful for facilitating cross-lingual communication through statistical machine translation. In this work we present an application of statistical machine translation to SMS messages. We decouple the SMS translation task into normalization followed by translation so that one can exploit existing bitext resources and present a novel unsupervised normalization approach using distributed representation of words learned through neural networks. We describe several surrogate data that are good approximations to real SMS data feeds and use a hybrid translation approach using finite-state transducers. Both objective and subjective evaluation indicate that our approach is highly suitable for translating SMS messages.

1 Introduction

The preferred form of communication has been changing over time with advances in communication technology. The majority of the world's population now owns a mobile device and an ever increasing fraction of users are resorting to Short Message Service (SMS) as the primary form of communication.

SMS offers an easy, convenient and condensed form of communication that is being embraced by the younger demographic. Due to the inherent limit in the length of a message that can be transmitted, SMS users have adopted several shorthand notations to compress the message; some that have become standardized and many that are invented constantly. While SMS is predominantly used in a monolingual mode, it has the potential to connect people speaking different languages. However, translating SMS messages has several challenges ranging from the procurement of data in this domain to dealing with noisy text (abbreviations, spelling errors, lack of punctuation, etc.) that is typically detrimental to translation quality. In this work we address all the elements involved in building a cross-lingual SMS service that spans data acquisition, normalization, translation modeling, messaging infrastructure and user trial.

The rest of the paper is organized as follows. In Section 4, we present a variety of channels through which we compiled SMS data followed by a description of our pipeline in Section 5 that includes normalization, phrase segmentation and machine translation. Finally, we describe a SMS translation service built using our pipeline in Section 6 along with results from a user trial. We provide some discussion in Section 7 and conclude in Section 8.

2 Related Work

One of the main challenges of building a machine translation system for SMS messages is the lack of training data in this domain. Typically, there are several legal restrictions in using consumer SMS data that precludes one from either using it completely or forces one to use it in limited capacity. Only a handful of such corpora are publicly available on the Web (Chen and Kan, 2013; Fairon and Paumier, 2006; Treurniet et al., 2012; Sanders, 2012; Tagg, 2009); they are limited in size and restricted to a few language pairs.

The NUS SMS corpus (Chen and Kan, 2013) is probably the largest English SMS corpus consisting of around 41000 messages. However, these messages are characteristic of Singaporean chat lingo and not an accurate reflection of SMS style in other parts of the world. A corpus of 30000 French SMS messages

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 974–983, Dublin, Ireland, August 23-29 2014.

was collected in (Fairon and Paumier, 2006) to study the idiosyncrasies of SMS language in comparison with standard French. More recently, (Pennell and Liu, 2011) have used twitter data as a surrogate for SMS messages. Most of these previous efforts have focused on normalization, i.e., translation of SMS text to canonical text while we are interested in translating SMS messages from one language into another (Eidelman et al., 2011).

Several works have addressed the problem of normalizing SMS text. A majority of these works have used statistical machine translation (character-level) to translate SMS text into standard text (Pennell and Liu, 2011; Aw et al., 2009; Kobus et al., 2008). (Beaufort et al., 2010) used a finite-state framework to learn the mapping between SMS and canonical form. A beam search decoder for normalizing social media text was presented in (Wang and Tou Ng, 2013). All these approaches rely on supervised training data to train the normalization model. In contrast, we use an unsupervised approach to learn the normalization lexicon of word forms in SMS to standard text.

While several works have addressed the problem of normalizing SMS using machine translation, there has been little to no work on the translation of SMS messages across languages on a large scale. Machine translation of instant messages from English-to-Spanish was proposed in (Bangalore et al., 2002) where multiple translation hypotheses from several off-the-shelf translation engines were combined using consensus decoding. However, the approach did not consider any specific strategies for normalization and the fidelity of training bitext is questionable since it was obtained using automatic machine translation. Several products that enable multilingual communication with the aid of machine translation in conventional chat, email, etc., are available in the market. However, most of these models are trained on relatively clean bitext.

3 Problem Formulation

The objective in SMS translation is to translate a foreign sentence $\mathbf{f}^{sms} = f_1^{sms}, \cdots, f_J^{sms}$ into target (English) sentence $\mathbf{e} = e_1^I = e_1, \cdots, e_I$. In general it is hard to procure such SMS bitext due to lack of data and high cost of annotation. However, we typically have access to bitext in non-SMS domain. Let $\mathbf{f} = f_1, \cdots, f_J$ be the normalized version of the SMS input sentence. Given $\mathbf{f}^{sms}$, we choose the sentence with highest probability among all possible target sentences,

$$\hat{\mathbf{e}}(\mathbf{f}^{sms}) = \arg\max_{\mathbf{e}} \{\mathrm{P}(\mathbf{e}|\mathbf{f}^{sms})\} \tag{1}$$

$$P(\mathbf{e}|\mathbf{f}^{sms}) \approx P(\mathbf{e}) \sum_{\mathbf{f}} P(\mathbf{f}^{sms}, \mathbf{f}|\mathbf{e}) \tag{2}$$

$$= P(\mathbf{e}) \sum_{\mathbf{f}} P(\mathbf{f}^{sms}|\mathbf{f}, \mathbf{e}) P(\mathbf{f}|\mathbf{e}) \tag{3}$$

If one applies the max-sum approximation and assumes that $P(\mathbf{f}^{sms}|\mathbf{f}, \mathbf{e})$ is independent of $\mathbf{e}$,

$$\hat{\mathbf{e}}(\mathbf{f}^{sms}) = \arg\max_{\mathbf{e}} P(\mathbf{f}^*|\mathbf{e}) P(\mathbf{e}) \tag{4}$$

where $\mathbf{f}^* = \arg\max_{\mathbf{f}} P(\mathbf{f}^{sms}|\mathbf{f})$. Hence, the SMS translation problem can be decoupled into normalization followed by statistical machine translation[1].

4 Data

Typically, one has access to a large corpus of general bitext $\{\mathbf{f}, \mathbf{e}\}$ while data from the SMS domain $\{\mathbf{f}^{sms}, \mathbf{e}\}$ is sparse. Compiling a large corpus of SMS messages is not straightforward as there are several restrictions on the use of consumer SMS data. We are not aware of any large monolingual or bilingual corpus of true SMS messages besides those mentioned in Section 2. To compile a corpus of SMS messages, we used three sources of data: transcriptions of speech-based SMS collected through

[1] One can also use a lattice output from the normalization to jointly optimize over $\mathbf{e}$ and $\mathbf{f}$

smartphones, data collected through Amazon Mechanical Turk[2] and Twitter[3] as a surrogate for SMS-like messages. We describe the composition of each of these data sources in the following subsections.

Corpus	Message	#count	Corpus	Message	#count
Speech SMS	*i love you*	988157	Amazon Mechanical Turk	*ily2*	N/A
	hello	881635		*n a meeting*	
	hi	607536		*check facebook*	
	how are you	470999		*kewl*	
	what's up	251044		*call u n a few*	
	what are you doing	218289	Twitter	*lol*	472556
	where are you	191912		*haha*	232428
	call	191430		*lmao*	102018
	lol	105618		*omg*	709504
	how's it going	102977		*thanks for the rt*	300254

Table 1: Examples of English messages collected from various sources in this work

4.1 Speech-based SMS

In the absence of access to a real feed of SMS messages, we used transcription of speech-based SMS messages collected through a smartphone application. A majority of these messages were collected while the users used the application in their cars. We had access to a total of 41.3 million English and 2.4 million Spanish automatic transcriptions. To avoid the use of erroneous transcripts, we sorted the messages by frequency and manually translated the top 40,000 English and 10,000 Spanish messages, respectively. Our final English-Spanish bitext corpus from this source of data consisted of 50,000 parallel sentences. Table 1 shows the high frequency messages in this dataset.

4.2 Amazon Mechanical Turk

The SMS messages from speech-based interaction does not consist of any shorthands or orthographic errors as the decoding vocabulary of the automatic speech recognizer is fixed. We posted a task on Amazon Mechanical Turk, where we took the speech-based SMS messages and asked the turkers to enter three responses to each message as they would on a smartphone. We iteratively posted the responses from the turkers as messages to obtain more messages. We obtained a total of 1000 messages in English and Spanish, respectively. Unlike the speech data, the responses contained several shorthands.

4.3 Twitter

Twitter is used by a large number of users for broadcasting messages, opinions, etc. The language used in Twitter is similar to SMS and contains plenty of shorthands, spelling errors even though it is typically not directed towards another individual. We compiled a data set of Twitter messages that we subsequently translated to obtain a bilingual corpus. We used the Twitter4j API[4] to stream Twitter data for a set of keywords (function words) over a week. The raw data consisted of roughly 106 million tweets. Subsequently, we performed some basic normalization (removal of @user, #tags, filtering advertisements, web addresses) to obtain SMS-like tweets. Finally, we sorted the data by frequency and picked the top 10000 tweets. Eliminating the tweets present in either of the two previous sources resulted in 6790 messages that we manually translated.

5 Framework

The user input is first stripped of any accents (Spanish), segmented into short chunks using an automatic punctuation classifier. Subsequently, any shorthand in the message is expanded out using expansion dictionaries (constructed manually and automatically) and finally translated using a phrase-based translation

[2]https://www.mturk.com
[3]https://twitter.com
[4]http://twitter4j.org/en/

model. Our framework allows the use of confusion networks in case of ambiguous shorthand expansions. We describe each component of the pipeline in detail in the following sections.

5.1 Tokenization

Our initial analysis of SMS messages from users, especially in Spanish indicated that while some users use accented characters in orthography, several others omit it for the sake of faster responses and convenience. Hence, we decided to train all our models on unaccented characters. Given a message, we convert all accented characters to their corresponding unaccented forms, e.g., baño → bano, followed by lowercasing of all characters. We do not perform any other kind of tokenization.

5.2 Unsupervised SMS Normalization

In Section 5.2, we described a static lookup table for expanding abbreviations and shorthands typically encountered in SMS messages, e.g., *4ever→forever*. While a static lookup table provides a reasonable way of handling common SMS abbreviations, it has limited coverage. In order to build a larger normalization lexicon, we used distributed representation of words to induce the lexicon in an unsupervised manner. Distributed word representations (Bengio et al., 2003; Collobert and Weston, 2008; Turian et al., 2010) induced through deep neural networks have been shown to be useful in several natural language processing applications. We use the notion of distributional similarity that is automatically induced through the word representations for learning automatic normalization lexicons.

Canonical form	Noisy form
love	loveeee, loveeeee, looove, love, wuv, wove, love, laffff, love, wuvvv, luhhhh, love, luvvv, luv
starbucks	starbs, sbucks
once	oncee, 1ce
tomorrow	tmrw, tomorrow, 2moro, tmrrw, tomarrow, tomoro, tomoz, 2mrw, tmr, tm, tmwr, 2mm, tmw, 2morro
forever	foreva, 5ever, foreverrrr, forver, foreeverrr, 4ever, 5eva, 4eva, foreevaa, forevs, foreve
because	cause, cos, coz, 'cos, 'cause, bc, because, becuz, bcuz, cuz, bcus, bcoz, because
homework	hwk, hw, hmwk, hmwrk, hmw, homeworkk, homwork, hmk, honework, homeowork
igualmente	igualmentee, igualment, iwalmente
siempre	simpre, siempre, 100pre, siempre, ciempre, siempre, siiempre, siemore, siempr, siemre, siempe
adios	adi, a10, adio
contigo	contigoo, cntigo, conmigo, contigoooo, kontigo, conmigoo, conmiqo
demasiado	demaciado, demasido, demasiademente, demasiao

Table 2: Examples from the unsupervised normalization lexicon induced through deep learning

We started with the 106 million tweets described in Section 4.3 and used a deep neural network identical to that used in (Collobert and Weston, 2008), i.e., the network consisted of a lookup table, hidden layer with 100 nodes and a linear layer with one output. However, we used a context of 5 words and corrupted the centre word instead of the last word to learn the distributed representations. We performed stochastic gradient minimization over 1000 epochs on the twitter data. Subsequently, we took the English and Spanish vocabularies in our translation model and found the 50 nearest neighbors using cosine distance for each word. We trained the above representations using the Torch toolkit (Collobert et al., 2011).

Feature	English		Spanish	
dimension	Precision	Recall	Precision	Recall
100	70.4	97.4	69.8	97.3
200	72.2	97.5	79.2	100
300	70.4	97.4	71.6	100

Table 3: Performance of the unsupervised normalization procedure. Only 1-best for each word was considered.

Once we obtained the 50 nearest neighbors for each word in the clean vocabulary, we used a combination of cosine metric threshold and Levenshtein distance (weighted equally) between the consonant

skeleton of the strings to construct the mapping lexicon. Finally, we inverted the table to obtain a normalization lexicon. Our procedure currently finds only one-to-one mappings. We took 60 singleton entries from the static normalization tables reported in Section 5.2 and evaluated the performance of our approach. The results are shown in Table 3 and some examples of learned normalizations are shown in Table 2.

5.3 Phrase Segmentation

In many SMS messages, multiple clauses may be concatenated without explicit punctuation. For example, the message *hi babe hope you're well sorry i missed your call* needs to be interpreted as *hi babe. hope you're well. sorry, i missed your call.* We perform phrase segmentation using an automatic punctuation classifier trained on SMS messages with punctuation. The classifier learns how to detect end of sentence markers, i.e. periods, as well as commas in the input stream of unpunctuated words.

An English punctuation classifier and a Spanish punctuation classifier was trained. The former was trained on two million words of smartphone data described in Section 4.1 while the latter was trained on 223,000 words of Spanish subtitles from the OpenSubtitles[5] corpus. From each of these data sets, a maximum entropy classifier was trained. Both classifiers utilized both unigram word and part of speech (POS) features of a window size of two words around the target word to be classified. A POS tagger trained on the English Penn Treebank provided English POS tags. Likewise, a Spanish POS tagger provided Spanish POS tags. The training data for the Spanish tagger, 1.6 million words in size, was obtained by running the Spanish Freeling parser over the Spanish version of TED talk transcripts. Results are shown in Table 4. Both phrase segmenters detect end of sentence well. The Spanish phrase segmenter detects commas better than the English one. This might be due to differences in the training sets; commas appear about 20 times more often in the Spanish data than in the English data.

	Class	Precision	Recall	F-measure
English	*period*	89.7	90.9	90.3
	comma	61.1	10.9	18.5
Spanish	*period*	94.3	87.4	90.7
	comma	74.2	37.4	49.7

Table 4: Performance of automatic phrase segmentation (numbers are in %)

5.4 Machine Translation

We used a phrase-based translation framework with the phrase table represented as a finite-state transducer (Rangarajan Sridhar et al., 2013). Our framework proceeds by using the standard procedure of performing word alignment using GIZA++ (Och and Ney, 2003) and obtaining phrases from the word alignment using heuristics (Zens and Ney, 2004) and subsequently scoring them. The phrase table is then represented as a finite-state transducer (FST). The FST decoder was used with minimum error rate training (MERT) to compute a set of weights for the log-linear model. It is important to note that the cost of arcs of the FST is a composite score (dot product of scores and weights) and hence requires an additional lookup during the N-best generation phase in MERT to obtain the component scores. The model is equivalent to Moses (**?**) phrase translation without reordering.

We noticed from the data collected in Section 4 that in typical SMS scenarios, a lot of phrases are stock phrases and hence caching these phrases may result in high accuracies instead of deriving the translation using a statistical model. We took the data created in Section 4 and created a FST to represent the sentences. The motivation is to increase the precision of common entries as well as reduce the latency involved in retrieving a translation from a statistical model. An example of the FST translation paradigm is shown in Figure 1

We experimented with the notion of using a consensus-based word alignment by combining the alignment obtained through different alignment tools. We used GIZA++ (Och and Ney, 2003), Berkeley

[5]http://www.opensubtitles.org

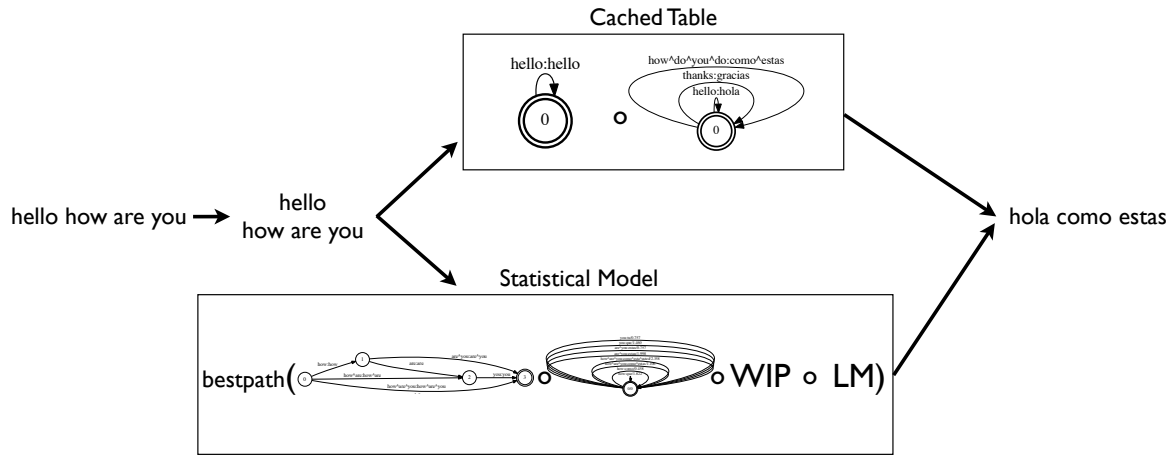


Figure 1: Illustration of the hybrid translation approach using FSTs. *WIP* and LM refer to the finite state automata for word insertion penalty and language model, respectively.

Alignment strategy	en2es	es2en
GIZA++	28.45	31.83
Pialign	28.08	33.48
Berkeley aligner	27.82	32.01
Union	28.01	33.14
Majority voting	27.32	32.96

Table 5: BLEU scores obtained using different alignment strategies. Only the statistical translation model was used in the evaluation.

aligner (Liang et al., 2006) and the Phrasal ITG aligner (Pialign) (Neubig et al., 2011). We combined the alignments in two different ways, taking the union of alignments or majority vote for each target word. For training the translation model, we used a total of 28.5 million parallel sentences obtained from the following sources: Opensubtitles (Tiedemann and Lars Nygaard, 2004), Europarl (Koehn, 2005), TED talks (Cettolo et al., 2012) and Web. The bitext was processed to eliminate spurious pairs by restricting the English and Spanish vocabularies to the top 150k frequent words as evidenced in a large collection of monolingual corpora. We also eliminated bitext with ratio of English to Spanish words less than 0.5. The initial model was optimized using MERT over 1000 parallel sentences from the SMS domain. Results of the machine translation experiments are shown in Table 5. The test set used was 456 messages collected in a real SMS interaction (see Section 6.1). The results indicate that consensus alignment procedure is not superior to the individual alignment outputs. Furthermore, the BLEU scores obtained through both the consensus procedures are not statistically significant with respect to the BLEU score obtained from the individual alignment tools. Hence, we used with the phrase translation table obtained using the Phrasal ITG aligner in all our experiments.

6 SMS Translation Service

In order to test the SMS translation models described in the previous sections, we created the infrastructure to intercept SMS messages, translate and deliver them in the preferred language of the recipient. The users were simply asked to register their numbers with a particular language through a Web portal and subsequently, all messages received by a user would be in the registered language. Some screenshots of interaction between users is shown in Figure 2. For the messages that are translated, we show both the original and translated messages. In cases where the translated message is longer than the character limit per message, we split the message over two message boxes.

6.1 User Evaluation

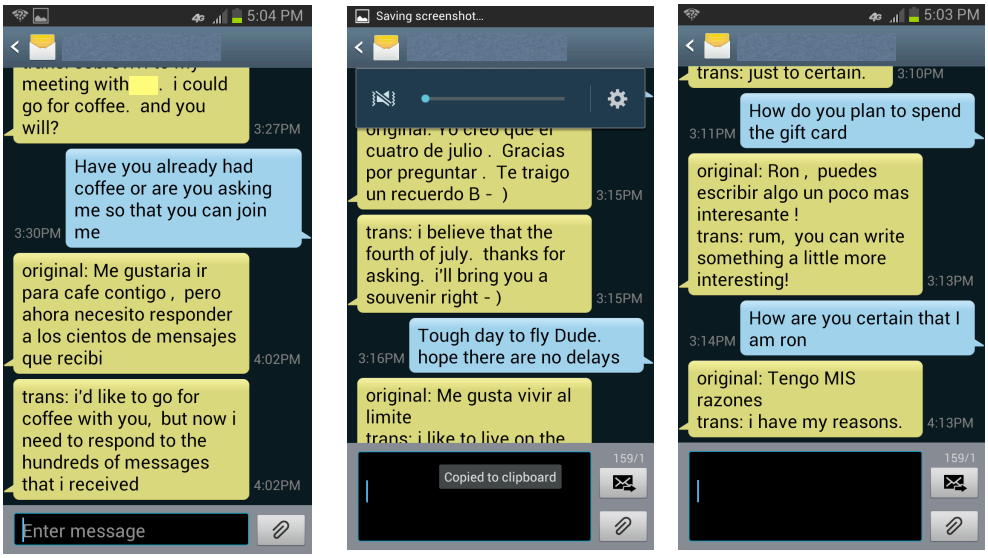


Figure 2: Screenshots of the SMS interface with translation

In order to test the SMS translation models described in the previous sections, we created the infrastructure to intercept SMS messages, translate and deliver them in the preferred language of the recipient. For the messages that are translated, we show both the original and translated messages. In cases where the translated message is longer than the character limit per message, we split the message over two message boxes. As part of the study we enrolled 20 English and 5 Spanish participants. The Spanish participants were bilingual while the English users had little to no knowledge of Spanish. Some of these interactions turned out to be short while others were had a large number of turns. We collected the messages exchanged over 2 days that amounted to 241 English and 215 Spanish messages.

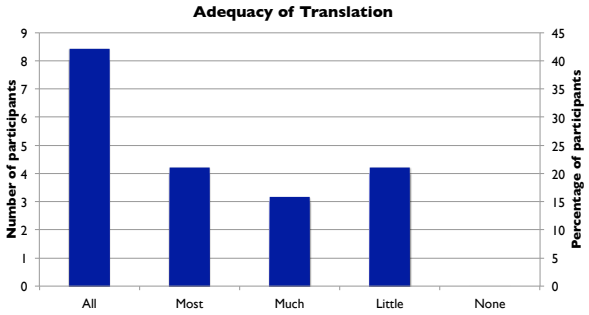


Figure 3: Subjective ratings regarding the adequacy of using SMS translation

We manually translated the 456 messages to create a test data set for evaluation purposes. In the absence of real SMS feeds in training, this test set is the closest we have to real SMS field data. The BLEU scores using the entire pipeline (normalization, punctuation, cached and statistical machine translation) for English-Spanish and Spanish-English was 31.25 and 37.19, respectively. We also created a survey for the participants to evaluate fluency and adequacy (LDC, 2005) Figures 3 and 4 show the survey results for adequacy and fluency, respectively. The results indicate that a majority of the people found the translation quality to be sufficiently adequate while the fluency was between *good* and *non-native*.

7 Discussion

The SMS bitext described in Section 4 consists of a total 58790 unique parallel sentences in the SMS domain. While the bulk of the data (speech-based) does not contain abbreviations and spelling errors, it

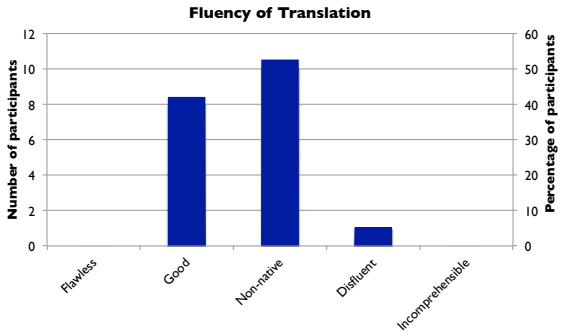


Figure 4: Subjective ratings regarding the fluency of using SMS translation

is highly representative of SMS messages and in fact is perfectly suited for statistical machine translation that typically uses normalized and tokenized data. The iterative procedure using Amazon Mechanical Turk is a good approach to procuring surrogate SMS data. We plan to continue harvesting data using this approach.

The unsupervised normalization lexicon learning using deep learning performs a good job of learning SMS shorthands. However, the induced lexicon contains only one-to-one word mappings. If one were to form compound words for a given dataset, the procedure can be potentially used for learning many-to-one and many-to-many mappings. Our framework also learns spelling errors rather well. It may also be possible to use distributed representations learned through log-linear models (Mikolov et al., 2013) for our task. However, this is beyond the scope of the work presented in this paper. Finally, we used only 1-best match for the unsupervised lexicon used in this work. One can potentially use a confusion network and compose it with the FST model to achieve higher accuracies. Our scheme results in fairly high precision with almost no false negatives (recall is extremely high) and can be reliably applied for normalization. The unsupervised normalization scheme did not yield significant improvements in BLEU score since our test set contained only 4 instances where shorthands were used.

Conventionally, sentence segmentation has been useful in improving the quality of statistical machine translation (Matusov et al., 2006; Matusov et al., 2005). Such segmentation, albeit into shorter phrases, is also useful for SMS translation. In the absence of phrase segmentation, the BLEU scores for English-Spanish and Spanish-English drop to 29.65 and 23.95, respectively. The degradation for Spanish-English messages is quite severe (drop from 37.19 to 23.95) as the lack of segmentation greatly reduces the use of the cached table. In the absence of segmentation, the cached table was used for 12.8% and 14.4% of the total phrases for English-Spanish and Spanish-English, respectively. However, with phrase segmentation the cached table was used for 29.2% and 39.2% of total phrases.

The subjective results obtained from the user trial augur well for the real use of translation technology as a feature in SMS. One of the issues in the study was balancing the English and Spanish participants. Since we had access to more English participants (20) in comparison with Spanish participants (5), the rate of exchange was slow. However, since SMS messages are not required to be real-time, participants still engaged in a meaningful conversation. Subjective evaluation results using LDC criteria indicate that most users were happy with the adequacy of translation while the fluency was rated as average. In general, SMS messages are not very fluent due to character limit imposed on the exchanges and hence machine translation has to use potentially disfluent source text.

8 Conclusion

We presented an application of statistical machine translation for translating SMS messages. We decoupled SMS translation into normalization followed by translation. Our unsupervised SMS normalization approach exploits the distributional similarity of words and learns SMS shorthands with good accuracy. We used a hybrid translation approach to exploit the repetitive nature of high frequency SMS messages. Both objective and subjective evaluation experiments indicate that our system generates translation with high quality while addressing the idiosyncrasies of SMS messages.

References

A. Aw, M. Zhang, J. Xiao, and J. Su. 2009. A phrase-based statistical model for SMS text normalization. In *Proceedings of COLING*, pages 33–40.

S. Bangalore, V. Murdock, and G. Riccardi. 2002. Bootstrapping bilingual data using consensus translation for a multilingual instant messaging system. In *Proceedings of COLING*.

R. Beaufort, S. Roekhaut, L. A. Cougnon, and C. Fairon. 2010. A hybrid rule/model-based finite-state framework for normalizing sms messages. In *Proceedings of ACL*, pages 770–779.

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

M. Cettolo, C. Girardi, and M. Federico. 2012. WIT3: Web Inventory of Transcribed and Translated Talks. In *Proceedings of EAMT*.

T. Chen and M. Y. Kan. 2013. Creating a live, public short message service corpus: the NUS SMS corpus. *Language Resources and Evaluation*, 47(2):299–335.

R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of ICML*.

R. Collobert, K. Kavukcuoglu, and C. Farabet. 2011. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*.

V. Eidelman, K. Hollingshead, and P. Resnik. 2011. Noisy SMS Machine Translation in Low-Density Languages. In *Proceedings of 6th Workshop on Statistical Machine Translation*.

C. Fairon and S. Paumier. 2006. A translated corpus of 30,000 french SMS. In *Proceedings of LREC*.

C. Kobus, F. Yvon, and G. Damnati. 2008. Normalizing sms: Are two metaphors better than one? In *Proceedings of COLING*, pages 441–448.

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

LDC. 2005. Linguistic data annotation specification: Assessment of fluency and adequacy in translations. Technical report, Revision 1.5.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of NAACL-HLT*, pages 104–111.

E. Matusov, G. Leusch, O. Bender, and H. Ney. 2005. Evaluating machine translation output with automatic sentence segmentation. In *Proceedings of IWSLT*, pages 148–154.

E. Matusov, A. Mauser, and H. Ney. 2006. Automatic sentence segmentation and punctuation prediction for spoken language translation. In *Proceedings of IWSLT*, pages 158–165.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at ICLR*.

Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. 2011. An unsupervised model for joint phrase alignment and extraction. In *Proceedings of the ACL*.

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

D. Pennell and Y. Liu. 2011. A character-level machine translation approach for normalization of SMS abbreviations. In *Proceedings of IJCNLP*.

V. K. Rangarajan Sridhar, J. Chen, S. Bangalore, A. Ljolje, and R. Chengalvarayan. 2013. Segmentation strategies for streaming speech translation. In *Proceedings of NAACL-HLT*.

E. Sanders. 2012. Collecting and analysing chats and tweets in SoNaR. In *Proceedings of LREC*.

C. Tagg. 2009. *Across-frequency in convolutive blind source separation*. dissertation, University of Birmingham.

J. Tiedemann and L. Lars Nygaard. 2004. The OPUS corpus - parallel & free. In *Proceedings of LREC*.

M. Treurniet, O. De Clercq, H. van den Heuvel, and N. Oostdijk. 2012. Collecting a corpus of Dutch SMS. In *Proceedings of LREC*, pages 2268–2273.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*.

P. Wang and H. Tou Ng. 2013. A beam-search decoder for normalization of social media text with application to machine translation. In *Proceedings of NAACL-HLT*.

Richard Zens and Hermann Ney. 2004. Improvements in phrase-based statistical machine translation. In *In Proceedings of HLT-NAACL*, pages 257–264.

A Hybrid Approach to Features Representation for Fine-grained Arabic Named Entity Recognition

Fahd Alotaibi
School of Computer Science
University of Birmingham, UK
fsa081@cs.bham.ac.uk
Faculty of Computing
King Abdulaziz University, KSA
fsalotaibi@kau.edu.sa

Mark Lee
School of Computer Science
University of Birmingham, UK
m.g.lee@cs.bham.ac.uk

Abstract

Despite considerable research on the topic of Arabic Named Entity Recognition (NER), almost all efforts focus on a traditional set of semantic classes, features and token representations. In this work, we advance previous research in a systematic manner and devise a novel method to represent these features, relying on a dependency-based structure to capture further evidence within the sentence. Moreover, the work also describes an evaluation of the method involving the capture of global features and employing the clustering of unannotated textual data. To meet this set of goals, we conducted a series of evaluations to evaluate different aspects that demonstrate great improvement when compared with the baseline model.

1 Introduction

Traditionally, the focus of Arabic NER has been on a very limited number of semantic classes, i.e. PERSON, ORGANISATION and LOCATION, utilising the newswire domain such as those described by Benajiba and Rosso (2008), Benajiba et al. (2010) and Abdul-Hamid and Darwish (2010) . This limits higher-level applications (such as question answering) from extracting in-depth knowledge and working on a relatively open domains.

This paper addresses the issue of a fine-grained NER of 50 classes for Arabic and presents a comprehensive set of experiments that evaluate innovative means of representing the features set. Thus, the contribution of this paper falls into different categories with unique outcomes, as follows:

1. A novel approach to representing the features is used, relying on dependency representation. This representation overcomes the drawback of current window-based representations of features.

2. The representation of global evidence involves clustering unannotated textual data, employing hierarchical clusters (Brown et al., 1992).

3. Due to the fact that there is no comparable work to use as a comparison in the task of Arabic fine-grained NER, a baseline model was developed, based on Conditional Random Fields (CRF), using the best features, as established and used elsewhere in the literature.

4. Development of publically available gold-standard fine-grained NER corpora[1] from two different genres, i.e. Newswire and Wikipedia.

Each contribution is discussed in more detail during in the remainder of this paper.

2 Arabic Fine-grained Named Entity Corpora

The majority of Arabic NER approaches are supervised, ensuring that the machine learns from an annotated corpus and aims to predict unseen text. This approach requires a reasonable bank of labelled data. This section examines the availability of such an annotated dataset at the fine-grained level, and the creation of gold-standard corpora.

[1] Available at: http://sourceforge.net/projects/arabic-named-entity-corpora/ and
Mirror at: http://fsalotaibi.kau.edu.sa/Pages-Arabic-NE-Corpora.aspx

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 984–995, Dublin, Ireland, August 23-29 2014.

2.1 Available Corpora

One of the earliest corpora publically released was ANERcorp, developed by Benajiba et al. (2007). This is a newswire based corpus and follows the CoNLL format. It annotates into four coarse-grained classes: PERSON, ORGANISATION, LOCATION, and MISCELLANEOUS. This dataset has been extensively used such as in (Benajiba et al., 2008b; Benajiba et al., 2010; Abdul-Hamid and Darwish, 2010).

Among corpora applying a fine-grained level of classes are those released by the Linguistic Data Consortium[2] (LDC). They released two multilingual NE corpora including Arabic (Mitchell et al., 2005; Walker et al., 2006). Both corpora were used in the Automatic Content Extraction (ACE) technology evaluation, at the coarse-grained level only. However, these corpora are governed by a costly annual license, which prevents the researcher from accessing and utilising them. At present, we are not aware of a study tackling fine-grained Arabic NER using this dataset.

Alotaibi and Lee (2013) released fine-grained Arabic NE corpora - WikiFANE$_{Selective}$ and WikiFANE$_{Whole}$. These were built automatically using the Arabic version of Wikipedia and released under the Creative Commons Attribution-ShareAlike 3.0 Unported Licence.[3]. These corpora apply a similar annotation taxonomy to that of the ACE corpus, but deliver increased coverage through the inclusion of a new class, i.e. PRODUCT, which includes Books, Movies, Sound, Hardware, Software, Food, Drugs and Other. Moreover, the corpora divide the PERSON class into 10 fine-classes, in order to provide wider coverage (i.e. Politician, Athlete, Businessperson, Artist, Scientist, Police, Religious, Engineer and Group). It is notable that this taxonomy can be easily mapped into CoNLL and ACE at either the coarse or fine-grained levels.

2.2 Creating Gold-standard Fine-grained Named Entity Corpora

Since the aim of this paper is to conduct an in-depth experiment for fine-grained Arabic NE, we manually created gold-standard fine-grained NE corpora for Arabic, drawing on two different genres. This gives a critical benchmark for evaluation and comparison with the automatically constructed corpus.

The first corpus is newswire-based, using the same textual data appearing in ANERcorp. The complete corpus was re-annotated to the fine-grained level. The second corpus is drawn from the Arabic version of Wikipedia. The selection of articles was made using a random heuristic, i.e. selecting articles discussing a named entity and maintaining a fair level of distribution among the classes. Moreover, the amount of textual data drawn from the Wikipedia article was restricted by avoiding such elements as lists, headings, and captions on images and tables.

2.3 Annotation Strategy and Quality Evaluation

For both corpora, the two-level taxonomy presented by Alotaibi and Lee (2013) was applied. This consists of 8 coarse-grained classes and 50 fine-grained classes. An in-house tool to facilitate the annotation process was developed. Two independent graduate-level Arabic native speakers were engaged to annotate the entire corpora. They were provided with extended instructions to guide them in the annotation process and a number of feedback sessions were conducted in the early stages of the process to ensure that any difficulties could be resolved.

After its completion, the quality of the annotation was evaluated by calculating the inter-annotator agreement between both annotators. The entity F-measure was used to evaluate the inter-annotation agreement as in (Hripcsak and Rothschild, 2005; Zhang, 2013). The corpora were named NewsFANE$_{Gold}$ and WikiFANE$_{Gold}$, referring to News-based, and Wikipedia-based, Fine-grained Arabic Named Entity Gold corpus, respectively. Micro-averaging was used while matching exact phrases, in order to calculate the agreement. The size and the inter-annotator agreement of NewsFANE$_{Gold}$ is 170K of tokens and 91% while WikiFANE$_{Gold}$ is 500K of tokens and 89% .

[2] https://www.ldc.upenn.edu/

[3] Available at: http://www.cs.bham.ac.uk/~fsa081/resources.html
Mirror at: http://sourceforge.net/projects/arabic-named-entity-corpora/

Corpus	Token level	Phrase level
NewsFANE$_{Gold}$	10.7	6.7
WikiFANE$_{Gold}$	13.1	7.4
WikiFANE$_{Selective}$	10.8	6.4
WikiFANE$_{Whole}$	7.08	4.9

Table 1: The density of NEs on token and phrase levels

Corpus	Length							
	1	2	3	4	5	6	7	8
NewsFANE$_{Gold}$	58.19	30.77	8	1.73	0.82	0.21	0.2	0.04
WikiFANE$_{Gold}$	51.75	31.55	10.88	3.48	1.34	0.46	0.21	0.12
WikiFANE$_{Selective}$	48.27	37.95	10.22	2.98	0.41	0.11	0.05	0.01
WikiFANE$_{Whole}$	66.22	25.85	6.02	1.58	0.05	0.02	0.01	0.01

Table 2: The distribution of NE phrases relative to length.

3 Corpus-based Evaluation and Comparison

It is important to closely evaluate and compare different corpora. The nature of the distribution of NE phrases is expected to differ to some extent, affecting the performance of learning the probabilistic model. Therefore, the coverage of NE phrases related to different aspects was studied, including the distribution of density, length and semantic classes.

3.1 The Density of NE

The density represents the coverage of NEs at the level of tokens and phrases. As can be seen in Table 1, WikiFANE$_{Gold}$ has the greater density at both levels. This demonstrates that the Wikipedia-based gold corpus tends to represent more NE phrases in context than that of the newswire-based. Although WikiFANE$_{Gold}$ is 0.7% denser than NewsFANE$_{Gold}$ in the phrase level, it reveals a notable difference (2.4%) in the token level. This indicates that WikiFANE$_{Gold}$ possess a greater variety in the length of NE phrases than the newswire-based corpus. However, the automatically developed corpus, WikiFANE$_{Selective}$, has a similar density of coverage as NewsFANE$_{Gold}$ whereas the WikiFANE$_{Whole}$ demonstrates a low level of density, due to its method of compilation.

3.2 The Distribution of the Length of Named Entity Phrases

It can be seen in Table 2, NewsFANE$_{Gold}$ and WikiFANE$_{Whole}$ tend to have more single-word NE phrases than other corpora. When it comes to the newswire corpus, this is due to differences in the way the NE phrases are written in a newswire domain. On the other hand, the boundaries of multi-word NE phrases are difficult to detect, in Arabic, due to the fact that the language has a complex morphology. This is demonstrated in the Wikipedia corpora, i.e. the gold and the selective - less than half the NE phrases in WikiFANE$_{Selective}$ are single-word, with a slightly higher rate found in WikiFANE$_{Gold}$.

3.3 The Distribution of the Fine-grained Classes

This demonstrates the distribution of NE phrases into fine-grained classes according to their annotation. As shown in Figure 1, the majority of classes tend (to some extent) to have a relatively harmonic distribution. In general, the newswire-based corpus tends to include more NE phrases related to politics, government, commerce, nations and cities, whereas the automatically-built corpora score a very high frequency on NE types such as 'Nation' and 'Population-centre'. Moreover, WikiFANE$_{Gold}$ shows wide distribution on most of the fine-grained classes of 'PERSON', 'LOCATION', 'FACILITY', 'VEHICLE' and 'PRODUCT', compared to other corpora.

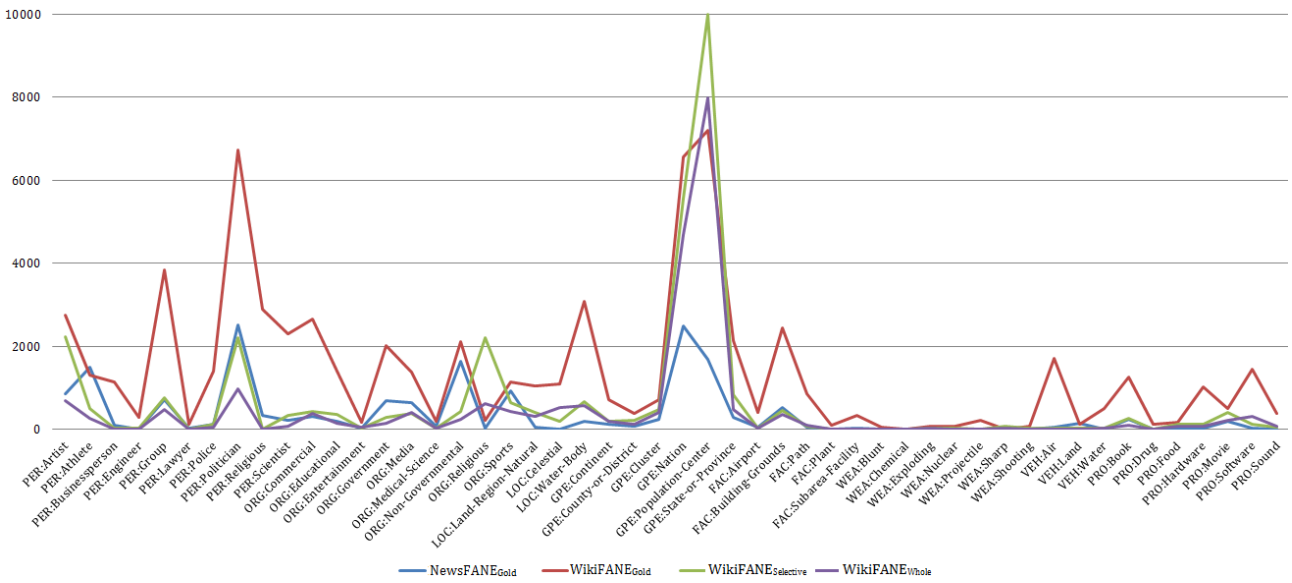


Figure 1: Distribution of Fine-grained Classes

4 The Baseline Model for Fine-grained Arabic NER

In order to prepare the baseline model and conduct successive experiments, the dataset for each corpus was divided into training, development and test. It is important to emphasise that, due to the limitations of computation power and the space allocated for the machine used, only a portion of WikiFANE$_{Selective}$ and WikiFANE$_{Whole}$ were selected with a size of ˜500K tokens each. The following table shows each corpus and its size.

Corpus	Type	Training	Dev	Test
NewsFANE$_{Gold}$	gold-standard	120K	25K	25K
WikiFANE$_{Gold}$	gold-standard	350K	75K	75K
WikiFANE$_{Selective}$	automatically-developed	354K	73K	73K
WikiFANE$_{Whole}$	automatically-developed	356K	72K	72K

Table 3: The size of the training, development and test for each corpus

Since there is no comparative work in the form of a fine-grained Arabic NER to use as a comparison, a baseline model based on Conditional Random Fields (CRF) was developed. It was decided to use the most successful features of the coarse-grained NER. For this purpose, the following features were extracted: ***Lexical and contextual features*** (current token, two tokens before and after the current token, first and last three characters of the token, and length of the token); ***Morphological features*** (gender, number and person); ***Syntactical features*** (part of speech and base phrase chunk); and ***External knowledge*** (the presence of the token in the gazetteer developed by Alotaibi and Lee (2013)). It was decided to use the BILOU scheme representation for the baseline model and successive experiments, as suggested by Ratinov and Roth (2009). The performance of the baseline model is presented in Table 4.

Corpus	Development			Test		
	P	R	F	P	R	F
NewsFANE$_{Gold}$	79.58	57.87	67.01	73.07	53.34	61.67
WikiFANE$_{Gold}$	62.19	43.67	51.31	68.13	44.78	54.04
WikiFANE$_{Selective}$	89.01	68.92	77.69	88.69	60.37	71.84
WikiFANE$_{Whole}$	82.35	49.83	62.09	84.27	58.63	69.15

Table 4: The results of the baseline model by learning CRF classifier with traditional features

5 Dependency based Features Representation

The current representation of the sequence tagging classifier involves using a predefined window of tokens (e.g. with size 5, including the current token) in order to capture local evidence. This representation has the following three drawbacks:

1. It is restricted to only capturing local evidence.
2. It fails to capture the relationship between dependent tokens, particularly for long sentences and multiword NE phrases.
3. Since Arabic has a relatively free word order, the window-based feature representation cannot capture the order variation for different examples.

In this paper, a new approach has been devised to utilise further evidence within a sentence in the classification process. The key idea informing this approach was to rely on the dependency-based representation of sentences in order to extract valuable features.

The dependency structure is one of syntactical representations, where a sentence is analysed by connecting its words in a word-to-word relationship. These relationships specify the head and dependent tokens in context, and assign a grammatical role for each token, e.g. subject, object and modifier.

To elaborate on the amount of knowledge that can be utilised based on the dependency structure, consider the following sentences:

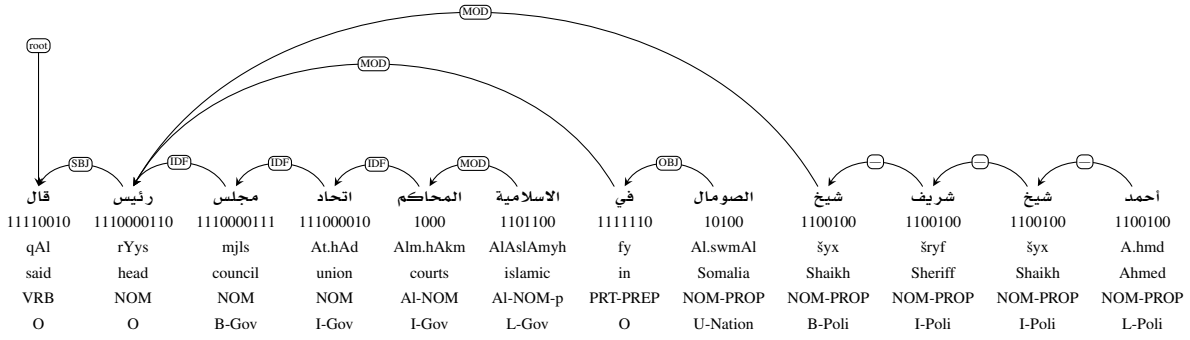
- (قال رئيس مجلس اتحاد المحاكم الاسلامية في الصومال شيخ شريف شيخ أحمد ...الخ /qAl rŷys mjls AtHAd AlmHAkm AlAslAmyħ fy AlSwmAl šyx šryf šyx OHmd fy ...Alx/ 'The head of the Council of the Islamic Courts Union, Sheikh Sharif Sheikh Ahmed, said in Somalia ...etc.')
- (يقول شارلز مورفي السياسي الانجليزي بعد الزيارة الأخيرة التي قام بها جون ميجور رئيس وزراء بريطانيا ...الخ /yqwl šArlz mwrfy AlsyAsy AlAnjlyzy bς d AlzyArħ AlOxyrħ Alty qAm bhA jwn myjwr rŷys wzrA' bryTAnyA ...Alx/ 'Charles Murphy, the English politician, said after the recent visit by John Major, Britain's prime minister ... etc.')
- (يذكر أن صلاد حسن انتخب رئيساً للصومال في اغسطس آب ٢٠٠٠ /yðkr On SlAd Hsn Antxb rŷysAã llSwmAl fy AγsTs Āb 2000/ 'It was mentioned that, Salad Hassan was elected as president of Somalia in August 2000')

The dependency representation and an English gloss of each example are shown in Figure 2. The parsed output includes a new set of information, which can be utilised as features, as follows:

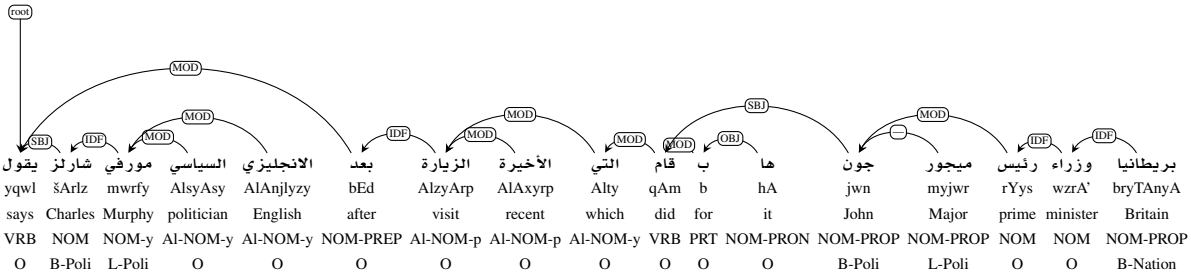
1. Head and Dependent Relation: The relationship between the head and the dependent is one of the most important features to capture. Consider the token (شيخ /šyx/ 'Shaikh'), in Figure 2a; the head (رئيس /rŷys/ 'the head of') is located far away and cannot be captured in the local window-based representation. Moreover, the vice versa relationship between the dependent and head is also useful. Consider the example in Figure 2b: the token (جون /jwn/ 'John') has two dependents (ميجور /myjwr/ 'Major') and (رئيس /rŷys/ 'Prime')[4] where the latter dependent (i.e. 'رئيس') gives a useful clue of the way in which it has been used in political contexts. The sequence of heads or dependents can also be utilised in the same way.

2. Sibling Relation: The sibling tokens are those dependent on the same head. Siblings can be located near each other in context, or appear at a distance. For example: the sibling of the token (شيخ /šyx/ 'Shaikh') is (مجلس /mjls/ 'council'), in Figure 2a, is expected to appear in a political context, which gives a clue towards the target NE class. Meanwhile, the token (في /fy/ 'in') is also a sibling, and can be avoided as it is a stop word. This is also the case in the example presented in Figure 2c, where the token (صلاد /SlAd/ 'Salad') is a sibling to the token (انتخب /Antxb/ 'elected'), which relates to the political context.

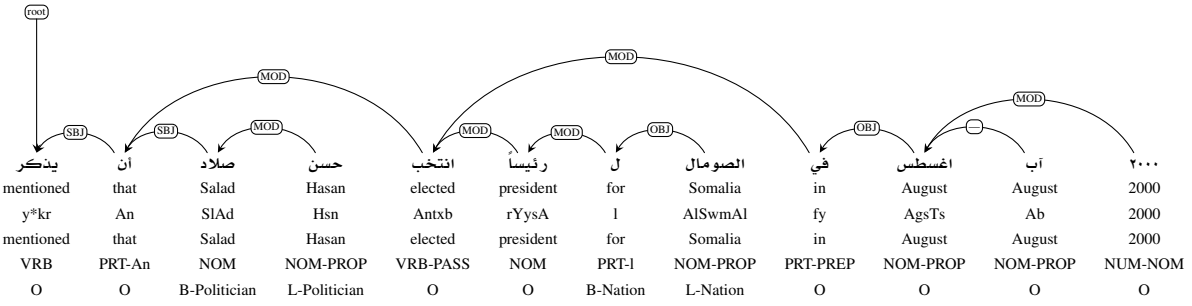
[4]Different contexts yield different English translation of the token "رئيس" as "the head of" and "Prime"



(a) The first example. (The second row represents the clusters according to the Brown algorithm)



(b) The second example



(c) The third example

Figure 2: The examples of a dependency structure. The rows show the Arabic token, Buckwalter transliteration, English gloss, POS and NE tag, respectively (the sentence is displayed left to right).

3. Syntactic Roles: The syntactical roles also benefit by being utilised to capture NE phrases in context. Among those with concern for NER are:

a. SBJ and OBJ: defines which subject and object roles are assigned to the head token of the NE phrase. For example, the tokens (**صلاد** /SlAd/ 'Salad') and (**شارلز** /šArlz/ 'Charles') are tagged as subjects.

b. IDF[5]*:* the Idafa chain is another important syntactical role, which helps to identify multiword NE phrases. For example: the token (**مورفي** /mwrfy/ 'Murphy') is tagged as an IDF of its previous token (**شارلز** /šArlz/ 'Charles'), where this indicates a multiword NE phrase. This is also the case for the example (**مجلس اتحاد المحاكم الإسلامية** /mjls AtHAd AlmHAkm AlIslAmyħ/ 'Council of the Islamic Courts Union') where all tokens are assigned an IDF role except the last token.

c. Flat relation (—): is a special role used by a CATiB pipeline parser for the sequence of proper nouns. For example: NE phrases such as (**شيخ شريف شيخ أحمد** /šyx šryf šyx OHmd/ 'Sheikh Sharif Sheikh Ahmed'), in which all tokens are assigned a flat relation.

[5]The naming of this abbreviation is used in CATiB to represent the syntactical role of idafa.

Corpus	Development			Test			+|-
	P	R	F	P	R	F	
NewsFANE$_{Gold}$	79.84	56.75	66.34	76.14	57.70	65.65	+3.98
WikiFANE$_{Gold}$	71.17	46.95	56.58	75.18	45.10	56.38	+2.34
WikiFANE$_{Selective}$	87.00	73.55	79.71	85.78	69.18	76.59	+4.75
WikiFANE$_{Whole}$	88.58	66.97	72.22	85.15	59.01	69.71	+0.56

Table 5: The results of the dependency-based features representation. ("+|-" represents the variation compared with the previous experiment)

5.1 Dependency-based Features set

The representation of the dependency structure presents each token as a node. A particular token (T) should have one node and only one head (H), except for the root, and zero or more dependents (D). A token (T) can have zero or more siblings (S), where they are connected, (i.e. are dependent), to the same head. Therefore, the following set of features has been extracted:

1. The current token T
2. POS of T
3. The presence of T in the Gazetteer
4. Syntactical role of T
5. Token of 1st, 2nd and 3rd Head H
6. Syntactical role of 1st, 2nd and 3rd H
7. POS of 1st, 2nd and 3rd H
8. Token of 1st, 2nd and 3rd Dependent D or 'NA' otherwise
9. Syntactical role of 1st, 2nd and 3rd D or 'NA' otherwise
10. POS of 1st, 2nd and 3rd D or "NA" otherwise
11. Token of 1st, 2nd and 3rd Sibling S or 'NA' otherwise
12. Syntactical role of 1st, 2nd and 3rd S or 'NA' otherwise
13. POS of 1st, 2nd and 3rd S or 'NA' otherwise

The 1st, 2nd and 3rd 'H' represent the parent, grandparent and great grandparent heads; while the 1st, 2nd and 3rd 'S' represent the first three siblings (if any).

5.2 Evaluation

It was decided to use the CATiB pipeline tool[6] (produced by Marton et al. (2013)), to parse all corpora and produce the set of features presented in the previous section. Since the POS tag set produced using the CATiB pipeline tool is very limited, it was decided instead to rely on the output of the AMIRA tokeniser and POS tagger produced by Diab (2009). The same classifier (CRF) was used, with a similar encoding scheme. Two experiments were conducted: the first was intended to evaluate the dependency-based representations. This was important in examining the effectiveness of the approach, compared with the window-based representation of local evidence. This is shown in Table 5, where in all corpora the performance of dependency-based representation alone outperforms that with window-based representation. The recall metrics reveal improvement across corpora, suggesting that the dependency-base representation has the ability to capture an increased number of NE phrases comparing to the traditional window-based representation.

The second experiment is intended to evaluate the integration in the classification process of dependency-based and window-based representations. This evaluation is expected to attain maximum benefit from both approaches in one model. The results in Table 6 demonstrate that the classifier tends to efficiently utilise both dependency-based and window-based representations in all corpora, apart from WikiFANE$_{Whole}$. The reason behind the degradation of the performance over the WikiFANE$_{Whole}$ dataset is due to the nature of the compiling of the corpus. Alotaibi and Lee (2013) state that this version includes entire sentences from Wikipedia articles, with no further filtering, ensuring that it is

[6]Not yet released to the public. We would like to thank the author for permission for its use.

Corpus	Development			Test			+|-
	P	R	F	P	R	F	
NewsFANE$_{Gold}$	82.08	57.77	67.81	80.21	61.58	69.68	+4.03
WikiFANE$_{Gold}$	89.31	49.11	63.37	83.34	50.48	62.88	+4.63
WikiFANE$_{Selective}$	87.03	73.29	79.57	87.31	76.17	77.81	+1.22
WikiFANE$_{Whole}$	82.44	57.91	68.03	75.88	52.45	62.03	-7.68

Table 6: The results of the hybrid approach using dependency-based and window-based features representation

possible to have sentences including NE phrases that are mistakenly assigned to 'O' class when using an automatic approach, as these NE phrases have no known destination in a Wikipedia article. This variety of mis-annotation is expected to propagate at this stage. It is worth noting that NewsFANE$_{Gold}$ and WikiFANE$_{Gold}$, as gold-standard corpora of different genres, reveal notable improvements of 4.03 and 4.63 F-measure respectively by using hybrid representation.

6 Further Exploiting of Global Evidences

Thus far, this study has examined the window-based and dependency-based representation of evidence, in order to increase the performance of the classification process. However, there is still room for improvement. Both approaches focus only at the sentence level. This section will investigate the approach to capturing global evidence. One means of achieving this is by utilising unannotated textual data, by clustering tokens into semantic groups based on context similarity. The reasoning behind this approach is that a NE token such as (الطائف /AlTAŷf/ 'Taif') (which is not seen in the training process) cannot be correctly classified, as it contains neither window-based nor dependency-based evidence in the training phase. Meanwhile, the token 'الطائف' is assigned to the same cluster of (لندن /lndn/ 'London') where the classifier knows that 'لندن' is a location. In this way, the knowledge capacity of the classifier has been broadened to a global level. A number of efforts have been undertaken for languages other than Arabic that demonstrate the usefulness of injecting clustering into NLP tasks, e.g. PCFG parsing (Candito and Crabbé, 2009) and dependency parsing (Koo et al., 2008). Utilising unannotated textual data in the supervised NER has already been variously studied with reference to English. The studies in (Turian et al., 2009; Turian et al., 2010; Tkachenko et al., 2012; Ratinov and Roth, 2009; Miller et al., 2004) reveal improvements when using the Brown clustering algorithm (Brown et al., 1992) to extract useful features.

This paper focuses on extracting a useful set of features from unannotated Arabic textual data, by relying on the Brown algorithm. We are not aware of any other study employing the Brown algorithm to Arabic textual data and in an Arabic NER task.

6.1 Brown Clustering and NER

The Brown clustering algorithm works by maximising the mutual information of bigrams. It uses hierarchical representation for the clusters. The hierarchal representation of the Brown clusters algorithm allows inclusion of different semantic levels of granularity. The output from the clustering delivers valuable information, which can be utilised by NER. This information can be divided into three categories:

1. The cluster of tokens belongs to the named entity category. For example, (شيكاغو /šykAγw/ 'Chicago') and (طوكيو /Twkyw/ 'Tokyo') belong to the same cluster, where both are NE type 'LOCATION'. In addition, (هديل /hdyl/ 'Hadeel') and (ممدوح /mmdwH/ 'Mamdooh') fall into similar clusters, and are both Personal NE.

2. The cluster of keyword tokens that have an informal insight to the target NE classes. For example, (كتائب /ktAŷb/ 'Brigades') and (منظمة /mnĎmħ/ 'Organisation') are keywords which infer the context of organisational NE. The context is expressed, for instance, as (كتائب شهداء الأقصى /ktAŷb šhdA' AlOqSý/ 'Al Aqsa Martyrs Brigades') or (منظمة العفو الدولية /mnĎmħ Alςfw

Aldwlyħ/ ‘Amnesty International’). Both head tokens in the NE phrases refer to the same cluster, which indicates the ‘ORGANISATION’.

3. The cluster of the head and dependent tokens the current token is pointing to. For example, the token (شيخ /šyx/ ‘Shaikh’), as shown in Figure 2a, is pointed to the head token (رئيس /rŷys/ ‘President’) where the ‘رئيس’ belongs to the cluster ‘1110000111’. This clustering knowledge permits the building of an abstract semantic representation for tokens. This implies that the token ‘رئيس’ can be replaced as (مدير /mdyr/ ‘Manager’) in other sentences where both tokens belong to the same cluster.

Further examples are presented in the Figure 3, where the group’s heading shows both name and cluster.

Locations: 0101101100	**First names: 000011111111101**
(بكين /bkyn/ ‘Beijing’)	(هديل /hdyl/ ‘Hadeel’)
(تكساس /tksAs/ ‘Texas’)	(حميدان /HmydAn/ ‘Homaidan’)
(طوكيو /Twkyw/ ‘Tokyo’)	(ممدوح /mmdwH/ ‘Mumdooh’)
Last names: 0000110001(01|10)	**Organisational keywords: 0111111111111011000**
(الساهر /AlsAhr/ ‘Alsaher’)	(كتائب /ktAŷb/ ‘battalions’)
(البخاري /AlbxAry/ ‘Albokhari’)	(جبهة /jbhħ/ ‘front’)
(الحازمي /AlHAzmy/ ‘Alhazmi’)	(منظمة /mnĎmħ/ ‘organization’)
Locational keywords: 011110110000	**Facility-related keywords: 101101100111011**
(مستوطنة /ktAŷb/ ‘settlement’)	(استاد /AstAd/ ‘stadium’)
(ضاحية /DAHyħ/ ‘suburb’)	(جسر /jsr/ ‘bridge’)
(محمية /mHmyħ/ ‘protectress’)	(مطار /mTAr/ ‘airport’)

Figure 3: Examples of the output of the Brown algorithm when applied to Arabic textual data.

6.2 Evaluation

The goal of this experiment was to evaluate the usefulness of injecting the clustering information from Brown algorithm into the supervised model. However, the actual size of the corpora mentioned in section 2.3 is too small to apply the Brown algorithm. Instead, a different set of different unannotated corpora, of a reasonably large size from different sources, was prepared for use in this experiment, as shown in Table 7.

Source of unannotated dataset	Size	Used for
NewsFANE$_{Gold}$ + Gigaword	1.17M	NewsFANE$_{Gold}$
WikiFANE$_{Gold}$ + 1/2(WikiFANE$_{Selective}$ & WikiFANE$_{Whole}$)	2.1M	WikiFANE$_{Gold}$
WikiFANE$_{Selective}$	2M	WikiFANE$_{Selective}$
WikiFANE$_{Whole}$	2M	WikiFANE$_{Whole}$

Table 7: Different textual data used in Brown algorithm

The first and second columns in Table 7 show the source of the unlabelled textual data used in the Brown algorithm and the respective size. The final column shows the target corpus using the knowledge in the CRF classifier.

Random stories were selected from Arabic Gigaword (Parker et al., 2011) as well as textual data from NewsFANE$_{Gold}$, to form unannotated data sized as 1.17M tokens. The Gigaword subset was selected due to the similarity of its genre to NewsFANE$_{Gold}$. The textual data for WikiFANE$_{Gold}$, and half of both WikiFANE$_{Selective}$ and WikiFANE$_{Whole}$ were compiled into one in order to induce clustering knowledge for WikiFANE$_{Gold}$.

The Brown algorithm was run in order to cluster the tokens into 1000 clusters, as suggested in (Miller et al., 2004; Liang, 2005; Ratinov and Roth, 2009; Tkachenko et al., 2012). The output of the Brown algorithm (which involves 1000 clusters) was injected as a set of features by extracting the clustering

Corpus	Development			Test			+|-
	P	R	F	P	R	F	
NewsFANE$_{Gold}$	86.13	70.38	77.46	81.66	68.36	74.42	+4.74
WikiFANE$_{Gold}$	77.80	62.36	69.23	79.87	60.19	68.64	+5.76
WikiFANE$_{Selective}$	89.17	74.04	80.90	88.64	73.18	80.17	+2.36
WikiFANE$_{Whole}$	90.39	69.97	78.88	84.98	65.00	73.66	+11.63

Table 8: The results of the injecting the output of Brown clustering into the CRF model

bits of (4, 6, 8, 10, 12) in a way that is similar to that presented by (Turian et al., 2010; Tkachenko et al., 2012). The reason behind this representation of the output is to allow a flexible level of grouping tokens into semantic clusters. For example, the tokens '**البخاري**' and '**الحازمي**' are clustered into '000011000101' and '000011000110', respectively, where both are personal NE. They share the first 10 bits of the cluster. This information allows for the extraction of useful knowledge to classify both tokens into the same class.

Table 8 shows notable improvement across all corpora. WikiFANE$_{Whole}$ and WikiFANE$_{Gold}$ score the highest, while other corpora gain improvements. It can be seen that the recall has sharply improved for approximately 7 to 13 points for NewsFANE$_{Gold}$, WikiFANE$_{Gold}$ and WikiFANE$_{Whole}$. This implies that the injecting of Brown clusters has improved the recall metric as a means of delimiting an increased number of NE phrases.

7 Related Work

This paper has addressed a series of issues, along with a discussion of the literature relevant to the context discussed in each section. Additional works of particular relevance are noted here. A large number of studies undertaking traditional Arabic NER have been developed, using a variety of methodologies to attain different goals. Using machine learning for the traditional task of NER has been addressed in different dimensions. Sequence labelling has also emerged, i.e. Maximum Entropy (Benajiba et al., 2007; Benajiba and Rosso, 2007); Support Vector Machine (Benajiba et al., 2008a); Conditional Random Fields (Benajiba and Rosso, 2008) and Structured Perceptron (Farber et al., 2008). Other hybrid approaches reliant on rule-based and ML are presented by (Shaalan and Oudah, 2013), a semi-supervised pattern is described in (AbdelRahman et al., 2010; Althobaiti et al., 2013) and the involvement of machine translation system to boost the performance of NER presented by (Zitouni and Florian, 2008). The researcher is not aware of any study tackling the fine-grained level of Arabic NER. Even that which has been developed for other languages (such as English) remains limited (Ling and Weld, 2012).

In terms of the representation of features, almost all studies in the Arabic NER apply the predefined window-based representation as examples when using this approach (Shaalan and Oudah, 2013; Benajiba et al., 2009). In English, Ratinov and Roth (2009) implemented two ways of capturing non-local features. The first approach is 'context aggregation'. This works by searching the entire document for a given token and returning the context of size two around each matched token. Ratinov and Roth (2009) limited the search to within 200 tokens. The second approach is 'extended prediction history', which looks up the 1000 previous tokens and counts the frequency of the label of the target class.

8 Conclusion

The majority of attempts to date to address NER focus on a limited number of semantic classes. This limitation has implications for other applications, such as question answering. This paper has presented an extended series of experiments and ideas, with the aim of constructing a fine-grained NER detailing resource creation to evaluation. Two approaches have been presented that rely on the output of the dependency parser and the clustering algorithm, instead of on a local window-based representation.

References

Samir AbdelRahman, Mohamed Elarnaoty, Marwa Magdy, and Aly Fahmy. 2010. Integrated machine learning techniques for arabic named entity recognition. *IJCSI International Journal of Computer Science*, 7(4):27–36.

Ahmed Abdul-Hamid and Kareem Darwish. 2010. Simplified feature set for arabic named entity recognition. In *Proceedings of the 2010 Named Entities Workshop*, pages 110–115, Uppsala, Sweden. Association for Computational Linguistics.

Fahd Alotaibi and Mark Lee. 2013. Automatically developing a fine-grained arabic named entity corpus and gazetteer by utilizing wikipedia. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 392–400, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Maha Althobaiti, Udo Kruschwitz, and Massimo Poesio. 2013. A semi-supervised learning approach to arabic named entity recognition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 32–40, Hissar, Bulgaria, September. INCOMA Ltd. Shoumen, BULGARIA.

Yassine Benajiba and Paolo Rosso. 2007. Anersys 2.0: Conquering the ner task for the arabic language by combining the maximum entropy with pos-tag information. In *Proceedings of the Workshop on Natural Language-Independent Engineering, 3rd Indian Int. Conf. on Artificial Intelligence, IICAI-2007*, Pune, India.

Yassine Benajiba and Paolo Rosso. 2008. Arabic named entity recognition using conditional random fields. In *Proceedings of the Workshop on HLT & NLP Within the Arabic World. Arabic Language and Local Languages Processing: Status Updates and Prospects, 6th International Conference on Language Resources and Evaluation*, volume 8, pages 26–31, Marrakech, Morocco. LREC-2008.

Yassine Benajiba, Paolo Rosso, and José Miguel Benedíruiz. 2007. Anersys: An arabic named entity recognition system based on maximum entropy. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 4394 of *Lecture Notes in Computer Science*, pages 143–153. Springer Berlin / Heidelberg.

Yassine Benajiba, Mona Diab, and Paolo Rosso. 2008a. Arabic named entity recognition: An svm-based approach. In *Proceedings of 2008 Arab International Conference on Information Technology (ACIT)*, pages 16–18, Amman, Jordan. Association of Arab Universities.

Yassine Benajiba, Mona Diab, and Paolo Rosso. 2008b. Arabic named entity recognition using optimized feature sets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 284–293, Honolulu, Hawaii. Association for Computational Linguistics.

Yassine Benajiba, Mona Diab, and Paolo Rosso. 2009. Using language independent and language specific features to enhance arabic named entity recognition. *IEEE Transactions on Audio, Speech and Language Processing. Special Issue on Processing Morphologically Rich Languages*, 17(5).

Y. Benajiba, I. Zitouni, M. Diab, and P. Rosso. 2010. Arabic named entity recognition: Using features extracted from noisy data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 281–285, Uppsala, Sweden. Association for Computational Linguistics.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Marie Candito and Benoît Crabbé. 2009. Improving generative statistical parsing with semi-supervised word clustering. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 138–141. Association for Computational Linguistics.

Mona Diab. 2009. Second generation amira tools for arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *2nd International Conference on Arabic Language Resources and Tools*.

Benjamin Farber, Dayne Freitag, Nizar Habash, and Owen Rambow. 2008. Improving ner in arabic using a morphological tagger. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, pages 2509–2514, Marrakech, Morocco. European Language Resources Association (ELRA).

George Hripcsak and Adam S Rothschild. 2005. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American Medical Informatics Association*, 12(3):296–298.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing.

Percy Liang. 2005. *Semi-supervised learning for natural language*. Ph.D. thesis, Massachusetts Institute of Technology.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Proceedings of the 26th Conference on Artificial Intelligence (AAAI)*.

Yuval Marton, Nizar Habash, and Owen Rambow. 2013. Dependency parsing of modern standard arabic with lexical and inflectional features. *Computational Linguistics*, 39(1):161–194.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *HLT-NAACL*, volume 4, pages 337–342. Citeseer.

Alexis Mitchell, Stephanie Strassel, Shudong Huang, and Ramez Zakhary. 2005. Ace 2004 multilingual training corpus [ldc2005t09], March 15. [accessed 20 December 2013].

Robert Parker, David Graff, Ke Chen, Junbo Kong, and Kazuaki Maeda. 2011. Arabic gigaword fifth edition [ldc2011t11], October 21. [accessed 20 December 2013].

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Khaled Shaalan and Mai Oudah. 2013. A hybrid approach to arabic named entity recognition. *Journal of Information Science*.

Maksim Tkachenko, Andrey Simanovsky, and St Petersburg. 2012. Named entity recognition: Exploring features. In *Proceedings of KONVENS*, pages 118–127.

Joseph Turian, Lev Ratinov, Yoshua Bengio, and Dan Roth. 2009. A preliminary evaluation of word representations for named-entity recognition. In *NIPS Workshop on Grammar Induction, Representation of Language and Language Learning*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus [ldc2006t06], February 15. [accessed 20 December 2013].

Ziqi Zhang. 2013. *Named entity recognition: challenges in document annotation, gazetteer construction and disambiguation*. Ph.D. thesis.

Imed Zitouni and Radu Florian. 2008. Mention detection crossing the language barrier. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 600–609. Association for Computational Linguistics.

Prior-informed Distant Supervision for Temporal Evidence Classification

Ridho Reinanda
University of Amsterdam
Amsterdam, The Netherlands
r.reinanda@uva.nl

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

Abstract

Temporal evidence classification, i.e., finding associations between temporal expressions and relations expressed in text, is an important part of temporal relation extraction. To capture the variations found in this setting, we employ a distant supervision approach, modeling the task as multi-class text classification. There are two main challenges with distant supervision: (1) noise generated by incorrect heuristic labeling, and (2) distribution mismatch between the target and distant supervision examples. We are particularly interested in addressing the second problem and propose a sampling approach to handle the distribution mismatch. Our prior-informed distant supervision approach improves over basic distant supervision and outperforms a purely supervised approach when evaluated on TAC-KBP data, both on classification and end-to-end metrics.

1 Introduction

Temporal relation extraction is the problem of extracting the temporal extent of relations between entities. A typical solution to the temporal relation extraction problem has three main components: (1) *passage retrieval*, (2) *temporal evidence classification*, and (3) *temporal evidence aggregation*. A community-based effort to evaluate temporal relation extraction was introduced in 2011 as a TAC Knowledge Base Population task: Temporal Slot Filling, or TSF for short (Ji et al., 2011).

An illustration of temporal slot filling is as follows. Having identified a `per:spouse` relation between two entities (Freeman Dyson, Imme Dyson), a system must establish the temporal boundaries from its supporting sentence. In the case of the sentence "*In 1958, he married Imme Dyson*", the goal is to find that the relation lasts from 1958 until the present day. Within the TSF setting, the boundaries are represented as beginning and ending intervals in a tuple (T_1, T_2, T_3, T_4) instead of an exact time expression, so as to allow uncertainty in the system output. We investigate temporal relation extraction following this setting. We focus on the temporal evidence classification part.

One of the challenges with relation extraction is the limited amount of training data available to capture the variations in a target corpus: temporal relation extraction faces the same challenge. Employing distant supervision (Mintz et al., 2009) is a way to address the challenge. But generating example training data in the temporal setting is not straightforward: we have to find not only the query and related entity, but also the time expression, in a single text segment.

Employing distant supervision for temporal evidence classification will introduce noise, in the form of labels and additional contexts (e.g., lexical features). A lot of previous work in distant supervision has been dedicated to reducing noise in distant supervision (Bunescu and Mooney, 2007; Riedel et al., 2010; Wei et al., 2012). We are interested in another phenomenon: the class distributions found in training data generated by a distant supervision approach. These distributions become an issue if the distant supervision corpus has a different structure and different characteristics compared to the target corpus, e.g., Wikipedia vs. news articles. We observe that in the case of temporal evidence, news articles and Wikipedia do indeed contain different class distributions. Our working hypothesis is that incorporating prior information about temporal class distribution helps improve our distant supervision approach. We

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 996–1006, Dublin, Ireland, August 23-29 2014.

test this hypothesis by comparing a distant supervision strategy with class priors to a distant supervision without class priors. We also demonstrate the effectiveness of our method by contrasting it with a purely supervised approach. In addition, we investigate how the difference in performance in temporal evidence classification affects the final score obtained in the overall end-to-end task.

We discuss related work in Section 2. In Section 3, we describe our distant supervision approach for temporal evidence classification. Our experimental setup is detailed in Section 4. We follow with results in Section 5 and a conclusion in Section 6.

2 Related Work

We discuss two groups of related work: on temporal slot filling and on distant supervision.

2.1 Temporal slot filling

Some previous work uses a pattern-based approach (Byrne and Dunnion, 2011); patterns are defined in terms of query entity, temporal expression, and slot value. For example, the word *divorce* should trigger that the relation *per:spouse* is ending. Other work uses temporal linking between time expressions and events in an event-based approach (Burman et al., 2011), where the source documents are annotated with TimeML event annotations (Pustejovsky et al., 2003); the authors use intra-sentence event-time links, and inter-sentence event-event links, following a TempEval approach (UzZaman et al., 2012). Garrido et al. (2012) use a graph-based document representation; they convert document context to a graph representation and use TARSQI to determine links between time expressions and events in documents and later map the resulting links into five temporal classes.

Li et al. (2012) combine flat and structured approaches to perform temporal classification. Their approach relies on a custom SVM kernel designed around flat (window and shallow dependency) features and structured (dependency path) features. The structured approach is designed to overcome the long context problem. They use a distant supervision approach for the temporal classification part, obtained on Freebase relations. They further extend their approach with self-training and relabeling (Ji et al., 2013).

Finally, Surdeanu et al. (2011) use n-grams around temporal expressions to train a distant supervision system. To be able to use Freebase facts, they find example sentences in Wikipedia, and use a window of five words from the temporal expression, using Freebase facts as *start* and *end* trigger. They use Jaccard correlation between n-grams to determine the association to *start* and *end*. Sil and Cucerzan (2014) performed distant supervision using facts obtained from Wikipedia infoboxes. From Wikipedia infoboxes, they retrieve the relevant sentences and build n-gram language models of the relations. In a slightly different setting (exploratory search), Reinanda et al. (2013) establish the temporal extent of entity associations simply by looking at their co-occurrence within documents in the corpus.

Our approach to temporal evidence classification differs from most existing approaches in its distant supervision scheme. We use distant supervision to directly perform a multi-class classification of temporal evidence against the five main temporal classes (including the *before* and *after* class), where most of the previous systems train a model to detect the beginning and ending of relationships only.

2.2 Reducing noise in distant supervision

With distant supervision (Mintz et al., 2009), indirect examples in the form of relations from a knowledge base such as Freebase and DBPedia are used. From these relation tuples, instances of relations in the form of sentences in the corpus are searched. Text features are later extracted from these sentences that are then used to train classifiers that can identify relations in the text corpus.

Reducing noise is an important ingredient when working with a distant supervision assumption. Relabeling is one such approach; Tamang and Ji (2012) perform relabeling based on semi-supervised lasso regression to reduce incorrect labeling. Wei et al. (2012) show that instances may be labeled incorrectly due to the knowledge base being incomplete. They propose to overcome the problem of incomplete knowledge bases for distant supervision through passage retrieval model with relation extraction.

Ritter et al. (2003) focus on the issue of missing data for texts that contain rare entities that do not exist in the original knowledge base. Riedel et al. (2010) work with a relaxed distant supervision assumption; they design a factor graph to explicitly model whether two entities are related, and later train this model with a semi-supervised constraint-driven algorithm; they achieve a 31 percent error reduction.

Bunescu and Mooney (2007) introduce multiple instance learning to handle the weak confidence in the assigned label. They divide the instances into a positive bag (at least one positive example) and a negative bag (all negative examples). They design a custom kernel to work with this weaker form of supervision. Surdeanu et al. (2012) operate on the same principle, but model the relation between entities and relation classes using graphical models. Hoffmann et al. (2011) also use multi-instance learning, but focus on overlapping relations.

What we add on top of existing work is the use of sampling techniques to correct for skewed distributions introduced through distant examples. We propose prior sampling, correcting the distributions of the classes in the generated examples to fit the target corpora.

3 Method

The temporal slot filling task is defined as follows: given a relation R = (q, r, s), where q is a query entity, r is a related entity, and s is a slot type, one must find T_R, a tuple of four dates (T_1, T_2, T_3, T_4) where R holds, where T_1 and T_2 form the beginning interval of the relation, and T_3 and T_4 is the ending interval. A system first must retrieve all passages or sentences expressing the relation between q and r. Each sentences and any time information within them will serve as intermediate evidence. This temporal evidence will later be aggregated and converted to tuple representation T_R.

In this paper, we focus on *temporal evidence classification*. That is, assuming the passage retrieval component has retrieved the relevant passages as intermediate evidence of temporal relations, we must classify whether the time expression t in the passage belongs to one these classes: BEGINNING, ENDING, BEFORE, AFTER, and WITHIN. In the training and evaluation data available to us, only the offsets of the time expression within the document are given for each intermediate evidence, therefore we first extract the paragraph and find the context sentence mentioning t.

Distant supervision for temporal classification The temporal slot filling task, as specified by TAC-KBP, defines 7 types of temporal-intensive relations. In our distant supervision approach, we use a separate knowledge base to find instances of the equivalent relations. We use Freebase as our reference knowledge base. That is, we use the temporal information found in Freebase to generate training examples. We manually map the TAC-KBP's 8 temporal relations into 6 Freebase mediator relations. The complete mapping of the relations can be found in Table 1.

In an article, entities and time expressions are not always referred to using their full mentions within a single sentence. Sometimes information is scattered around several sentences: the query entity q in the first sentence, later referred to using a pronoun in the second sentence that contains a time expression, etc. One common way to deal with this problem is to run full co-reference resolution, therefore ensuring all mentions are resolved. We handle this problem by relaxing the distant supervision rule. Rather than retrieving sentences, we retrieve passages containing the query entity q, and related entity r instead. We later replace every pronoun found within the passage with q. Based on our analysis of the Wikipedia articles, this simple heuristic should work, because most Wikipedia articles are entity-centric, and a lot of the pronouns mentioned in the articles will refer to the query entity q.

Each relation that we mapped from Freebase has temporal boundaries *from* and *to*. Following Li et al. (2012), we use Algorithm 1 to generate the training examples, but adapt it suit to our assumption.

Sampling the DS examples We manually compared our main corpus (TAC document collection) and our distant supervision corpus (Wikipedia) and noticed some discrepancies. The main corpus mainly consists of newswire articles; one of the main difference between Wikipedia articles and newswire articles is that Wikipedia articles mainly consist of milestone events. In terms of class distribution, this means that most of the generated examples will be in the form of BEGINNING and ENDING class, followed by the BEFORE and AFTER class, with the smallest number of examples belonging to the

TAC Relations	Freebase Relations
per:spouse	marriage
per:title	employment-tenure, goverment-position-held
per:employee-of	employment-tenure
per:member-of	political-party-tenure
per:cities-of-residence	places-lived
per:stateorprovinces-of-residence	places-lived
per:countries-of-residence	places-lived
org:top-employees/members	organization-leadership

Table 1: Relation mapping to Freebase.

Data: Freebase temporal relation (q, r, $from$, to)
Result: labeled training examples
Retrieve the Wikipedia article of the query entity q;
Split article into passages;
Retrieve the passages containing q, r;
Extract all time expressions from the passages;
for *time-expression* t **do**
    Retrieve the context sentence s containing t;
    If t is $from$: use s, t as BEGINNING example;
    If t is to : use s, t as ENDING example;
    If t before $from$: use s, t as BEFORE example;
    If t after to : use s, t as AFTER example;
    If t between $from$ and to : use s, t as WITHIN example;
end

Algorithm 1: Training data generation.

WITHIN class. In newswire, however, we tend to see something different; most of the time expressions will belong to the WITHIN class.

We argue that using the training data with a "smarter" prior is important. More data not only means more information, but may also mean more noise. This is particularly important with the *relaxed distant supervision* assumption that we have. Therefore, we choose to sample instead of using all of the generated training examples.

We employ two sampling strategies: *uniform*, sampling from our generated training data and deliberately fitting them to a uniform distribution; and *prior-sampling*, where we deliberately construct training data to fit a prior distribution. One way to estimate such a prior is by looking at the distributions of classes in the gold-standard training data that we have. In the case where gold-standard data is not available, we can use a heuristic to estimate the distributions of temporal classes based on domain knowledge or on observations of the target corpora.

In summary, we generate the final training data according to the following steps. First, generate training data with the DS approach described before. Next, estimate class distributions from the (supervised) training data. Then, sample examples from the generated DS data with the probability estimated from the supervised training data (i.e., the empirical prior). Keep sampling the training examples until we

reach the target percentage of the DS data. Finally, use the sampled training data to train the multi-class classifier.

Feature representation Both for the training, evaluation, and DS data, we extract the context sentence, i..e, the sentence containing the relation and time expression t.

We normalize the context sentence as follows. First, we detect named entities within the sentence and replace the mentions with their entity types (PERSON, ORGANIZATION, or LOCATION). Second, we detect other time expressions within the context and normalize them with regard to the main time expression t, i.e., by normalizing them into TIME-LT and TIME-GT. The idea is to capture the relationships between time expressions as features.

We extract lexical features from the normalized sentence. This comprises tokens surrounding the query entity, related entity (slot filler), and time expression. We consider the following four models as our feature representations:

Model-1: bag-of-words All tokens within the normalized sentences are used as features.

Model-2: context window All tokens within the proximity of 3 tokens from the query entity, related entity, and time expression are used as features.

Model-3: context window with trigger words lexicon All tokens within the proximity of 3 token from the query entity, related entity, and time expression are used as features. In addition, a list of keywords which might indicate the beginning and ending of relationships are used as gazetteer features. These list of keywords are expanded by using WordNet to extract related terms.

Model-4: context window with position All tokens within the proximity of 3 tokens from the query entity, related entity, and time expression are used as features. Rather than considered as bag-of-words tokens, the positions of word occurrences are now taken into account as features.

4 Experimental Setup

We introduce the dataset and the setup of our experiments. Before that we formulate our research questions as these dictate our further choices.

Research questions We aim to answer the following research questions:

RQ1 How does a purely supervised approach with different features and learning algorithms perform on the task of temporal evidence classification?

RQ2 How does the performance of a distant supervision approach compare to that of a supervised learning approach on the task of temporal evidence classification?

RQ3 How does the performance of a prior-informed distant supervision approach compare to that of a basic distant supervision approach on the task of temporal evidence classification?

RQ4 How do the approaches listed above compare in terms of their performance on the end-to-end temporal relation extraction task?

Corpora and knowledge base We use the TAC 2011 document collection, which contains 1.7M documents, consisting of news wires, web texts, broadcast news, and broadcast conversation. We use a recent version of Freebase (October 2013) as our knowledge base and retrieve the latest version of Wikipedia as our distant supervision corpus.

Ground truth We use the TAC-KBP 2011 Temporal Slot Filling Task dataset (Ji et al., 2011) as the ground truth in our experiments. The ground truth comes in two forms: intermediate evidence (with classification labels) and tuples (boundaries of each relation). We use the intermediate evidence to evaluate our temporal evidence classification framework. We later use the provided tuples to evaluate the end-to-end result.

The dataset contains 173 examples in the training set and 757 examples in the evaluation set. The distribution of the classes is shown in Table 2.

Class	Training	Evaluation	DS Training
WITHIN	66	357	6,129
BEGINNING	59	217	22,508
ENDING	30	110	16,775
BEFORE	9	45	24,932
AFTER	9	28	12,499

Table 2: Class distribution statistics.

Evaluation metric We use F1 as the main evaluation metric for the temporal evidence classification task. For the end-to-end temporal information extraction task, we use the evaluation metric proposed in TAC-KBP 2011, i.e., the Q score. Given a relation r and the ground truth interval tuple G_r, $Q(T_r)$, the quality score of a tuple T_r returned by system S is computed as follows:

$$Q(T_r) = \frac{1}{4} \sum_{i=1}^{4} \frac{1}{1+d_i},$$

where d_i is the absolute difference between T_i in system response and the ground truth tuple G_i (measured in years). To obtain an overall system Q score, we average the Q scores obtained from each relation tuple returned.

Experiments We run four contrastive experiments. In Experiment 1, we contrast the performance on the temporal evidence classification task of the different choices for our supervised methods (Model-1, -2, -3, -4), using either Support Vector Machine, Naive Bayes, Random Forest, or Gradient Boosted Regression Tree. In Experiment 2 we examine our distant supervision method and contrast its performance with the supervised methods from Experiment 1. In Experiment 3, we contrast different sampling methods for our distant supervision method.

In Experiment 4 we consider the overall performance on the temporal relation extraction task of our methods; in this experiment we use three "oracle runs" that we have not introduced yet: first, the *Label-Oracle* run uses the actual temporal classification label from the ground truth, use these ground truth label to aggregate the evidence and create the temporal tuples, and compute the end-to-end score; second, *Within-Oracle* assigns all temporal evidence to the WITHIN class; third, *Nil-Baseline* is a lower-bound run that assigns NIL to every element of the temporal tuples.

We use the implementations of the learning algorithms in the Scikit-learn machine learning package (Pedregosa et al., 2011).

5 Results and Discussion

We present the outcomes of the four experiments specified in the previous section.

5.1 Preliminary experiment

To answer RQ1, *How does the performance of the supervised learning approaches on the temporal evidence classification task vary with different representations and learning algorithms?*, we start with a preliminary experiment. The aim of this experiment is to get an idea of the classification performance with a purely supervised approach. The results are shown in Table 3.

As shown in Table 3, Model-4 with the SVM and NB classifiers achieves the best overall performance. There seems to be a gradual increase in performance from the simpler to the more complex model with SVM and NB classifiers, with the exception of RF. Interestingly, GBRT seems only slightly affected by the different choice of model in this supervised setting.

5.2 Distant supervision experiments

Next, we evaluate the distant supervision approach. We aim to answer RQ2, *How does the performance of the distant supervision approach compare to that of the supervised learning approach?* We generate

Model	SVM	NB	RF	GBRT
Model-1	0.405	0.361	0.402	0.422
Model-2	0.409	0.417	0.354	0.420
Model-3	0.412	0.418	0.361	0.420
Model-4	**0.426**	0.424	0.241	0.422

Table 3: Experiment 1. Supervised approaches to temporal evidence classification.

training examples with the approach described in Section 3, and use the full generated training data to train SVM and Naive Bayes classifiers with the same representation models that we use in the previous experiments. The results are shown in Table 4.

Model	Supervised	DS	DS-uniform	DS-prior
Model-1 SVM	0.405	0.212	0.379	0.408
Model-2 SVM	0.409	0.185	0.389	0.450
Model-3 SVM	0.412	0.183	0.384	0.452
Model-4 SVM	0.426	0.200	0.400	0.463
Model-1 NB	0.361	0.413	0.379	0.431
Model-2 NB	0.417	0.299	0.372	0.451
Model-3 NB	0.418	0.300	0.368	0.446
Model-4 NB	0.424	0.270	0.400	**0.486**
Model-1 RF	0.402	0.162	0.406	0.397
Model-2 RF	0.354	0.177	0.399	0.418
Model-3 RF	0.361	0.176	0.391	0.403
Model-4 RF	0.241	0.171	0.399	0.446
Model-1 GBRT	0.422	0.142	0.316	0.344
Model-2 GBRT	0.420	0.137	0.343	0.418
Model-3 GBRT	0.420	0.138	0.343	0.403
Model-4 GBRT	0.422	0.140	0.399	0.433

Table 4: Experiment 2 and 3. Supervised, distant supervision, and distant supervision with sampling approaches to temporal evidence classification.

We observe that the distant supervision approach trained on the full set of generated examples (the column labeled "DS") performs poorly, well below the supervised approach. We hypothesize that the accuracy drops due to the amount of noise generated with our distant supervision assumption trained from full data, and different class distribution statistics.

In Section 3, we proposed our prior-sampling approach for distant supervision. The next experiment is meant to answer RQ3, *How does the performance of our prior-informed distant supervision approach compare to that of the basic distant supervision approaches?* We sample 20 percent of the generated examples datasets with the following strategies: *uniform* and *prior*. The results are also shown in Table 4, in the columns labeled "DS-uniform" and "DS-prior," respectively.

By observing the results in Table 4, we notice that distant supervision with prior sampling performs the best, for every combination of model and classification method. *Uniform* sampling already helps in improving the performance, and prior sampling successfully boosts the performance of the basic distant supervision (for all four models) further. Distant supervision with prior sampling also performs consistently better than the supervised approaches (Table 3) in many cases—interestingly, for GBRT, DS-prior only outperforms the supervised methods with sufficiently complex queries (Model-4 GBRT).

5.3 End-to-end experiments

Next, we answer RQ4. That is, we consider how the classification performance on temporal evidence classification affects the end-to-end result. We take the best performing models from the previous experiments and evaluate their end-to-end scores. The results are shown in Table 5.[1]

Model	Avg-Q	F1
Label-Oracle	0.925	1.000
Within-Oracle	0.676	0.302
Nil-Baseline	0.393	N/A
Supervised		
Model-4 SVM	0.657	0.426
Model-4 NB	0.648	0.424
Model-4 RF	0.573	0.241
Model-4 GBRT	0.649	0.422
Distant supervision		
Model-4 SVM	0.669	0.463
Model-4 NB	0.679	0.486
Model-4 RF	0.653	0.446
Model-4 GBRT	0.669	0.433

Table 5: Experiment 4. End-to-end scores (Avg-Q) next to F1 scores for temporal evidence classification.

From Table 5, we see that Model-4 RF (F1 on temporal evidence classification 0.446) and Model-4 GBRT (F1 on temporal evidence classification 0.433) translate into 0.653 and 0.669, respectively, in terms of Q-score. This means that the misclassifications that Model-4 RF produces have a larger impact than those of Model-4 GBRT. However, the difference in performance is not large.

The evaluation of this end-to-end task is important because not every misclassification has a similar cost. Misclassification of class A into class B can result in a huge increase/decrease in performance. First, the classification performance does not directly map to the end-to-end score. Second, several relations have more pieces of evidence than others; performing misclassifications on relations that have a lot of supporting evidence would probably have less effect on the final score.

The state of the art performance, using distant supervision (Li et al., 2012), achieves an end-to-end Avg-Q score of 0.678 (on training data), where we achieve 0.679 (on evaluation data). However, our scores are not directly comparable since we reduce the number of classes (and the amount of evidence) in our evaluation. It is important to note that Li et al. (2012) use a complex combination of flat and structured features as well as the web, where we use relatively simple features with Wikipedia and prior sampling.

Furthermore, our approach manages to achieve the same level of end-to-end performance as the Within-Oracle run, while achieving a significantly better F-score. More pieces of evidence were actually classified correctly, though this was not reflected directly in the end-to-end score due to issues described above.

5.4 Error Analysis

We proceed to analyse parts of our end-to-end results to see what is causing errors in the temporal evidence classification task. We found several common problems.

Semantic inference Some problems had to do with the fact that several snippets require semantic inference. The fact that someone dies effectively ends any relationships that this person had. Another example is when someone marries someone (A marries C), and this beginning of relationships effectively

[1] As the Nil-Baseline is applied directly to the final tuples rather than the classification labels, there are is no F1 score for this run.

means the end of relationships for previous relations (A and B). A more complex method to deal with this type of semantic inference is needed, simple classification does not work so well. Here is an example:

> *Angela Merkel is married to Joachim Sauer, a professor of chemistry at Berlin's Humboldt University, since 1998. Divorced from Ulrich Merkel. No children.*

For this example the fact is that the time expression 1998 happens *after* with regard to the *spouse* relation between Angela Merkel and Ulrich Merkel.

Concise temporal representations Newspaper articles contain lots of temporal information in a concise way. For example in the form (X–Y). This implicit interval range is not expressed in a lexical context but rather with symbolic conventions. In several articles, the information encoded is almost tabular rather than expressed in explicitly. For example:

> *Elected as german chancellor Nov. 22, 2005. Chairwoman, christian democratic union, 2000-present. Chairwoman, christian democratic parliamentary group, 2002–2005.*

Complex time-inference BEFORE and AFTER are especially tricky to deal with because they require additional inference. Even if a passage contains the word *after*, the time expression linked to it would probably contain the *before* relation.

> *He was called up by the Army in the spring of 1944, after marrying bea silverman in 1943, and was sent to The Philippines.*

For the above example, 1943 happens *before* the "person joined the Army" event.

We observe quite a number of these cases on the evaluation data. Furthermore, the lack of context on some examples and evidence that is scattered around multiple sentences complicates the problem even more. Because of semantic and implicit evidence, temporal evidence classification remains a challenging task. In order to achieve a better absolute performance, collective classification/inference of evidence seems an interesting option.

6 Conclusion

We have presented a distant-supervision approach to temporal evidence classification. The main feature of our distant supervision approach is that we consider the prior distribution of classes in the target domain in order to better model the task. We show that our prior-informed distant supervision approach manages to outperform a purely supervised approach. Our method also achieves state-of-the-art performance on end-to-end temporal relation extraction with fewer and simpler features than previous work.

Our error analysis on the temporal evidence classification task revealed several issues that inform our future work aimed at further improving the performance on the subtask of temporal evidence classification, and the overall temporal relation extraction task. In particular, we intend to deal with the challenging aspect of semantic inference over relations found in the evidence passage. Another interesting direction that we aim to tackle is dealing with evidence that is scattered across multiple sentences.

Acknowledgements

This research was supported by the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreements nrs 288024 and 312827, the Netherlands Organisation for Scientific Research (NWO) under project nrs 727.011.005, 612.001.116, HOR-11-10, 640.006.013, the Center for Creation, Content and Technology (CCCT), the QuaMerdes project funded by the CLARIN-nl program, the TROVe project funded by the CLARIAH program, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), the Netherlands eScience Center under project nr 027.012.105, the Yahoo! Faculty Research and Engagement Program, the Microsoft Research PhD program, and the HPC Fund.

References

Razvan C. Bunescu and Raymond J. Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL'07)*, Prague, Czech Republic, June.

Amev Burman, Arun Jayapal, Sathish Kannan, Ayman Kavilikatta, Madhu abd Alhelbawy, Leon Derczynski, and Robert Gauzauskas. 2011. USFD at KBP 2011: Entity linking, slot filling and temporal bounding. In *Proceedings of the TAC-KBP 2011 Workshop*. NIST.

Lorna Byrne and John Dunnion. 2011. UCD IIRG at tac 2011. In *Proceedings of the TAC-KBP 2011 Workshop*. NIST.

Guillermo Garrido, Anselmo Peñas, Bernardo Cabaleiro, and Álvaro Rodrigo. 2012. Temporally anchored relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 107–116, Stroudsburg, PA, USA. Association for Computational Linguistics.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 541–550, Stroudsburg, PA, USA. Association for Computational Linguistics.

Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. Overview of the TAC 2011 knowledge base population task. In *Proceedings of the TAC-KBP 2011 Workshop*. NIST.

Heng Ji, Taylor Cassidy, Qi Li, and Suzanne Tamang. 2013. Tackling representation, annotation and classification challenges for temporal knowledge base population. *Knowledge and Information Systems*, pages 1–36.

Qi Li, Javier Artiles, Taylor Cassidy, and Heng Ji. 2012. Combining flat and structured approaches for temporal slot filling or: how much to compress? In *Proceedings of the 13th international conference on Computational Linguistics and Intelligent Text Processing - Volume Part II*, CICLing'12, pages 194–205, Berlin, Heidelberg. Springer-Verlag.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL '09)*, pages 1003–1011, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Mattheiu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003. TimeML: Robust specification of event and temporal expressions in text. In *Fifth International Workshop on Computational Semantics (IWCS-5*.

Ridho Reinanda, Daan Odijk, and Maarten de Rijke. 2013. Exploring entity associations over time. In *SIGIR 2013 Workshop on Time-aware Information Access*, August.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part III*, ECML PKDD'10, pages 148–163, Berlin, Heidelberg. Springer-Verlag.

Alan Ritter, Luke Zettlemoyer, Mausam, and Oren Etzioni. 2003. Modeling missing data in distant supervision for information extraction. In *Transactions of the Association for Computational Linguistics*, TACL'13, pages 367–378, Stroudsburg, PA, USA. Association for Computational Linguistics.

Avirup Sil and Silviu Cucerzan. 2014. Temporal scoping of relational facts based on Wikipedia data. In *CoNLL: Conference on Natural Language Learning*.

Mihai Surdeanu, Sonal Gupta, John Bauer, David McClosky, Angel X. Chang, Valentin I. Spitkovsky, and Christopher D. Manning. 2011. Stanford's distantly-supervised slot-filling system. In *Proceedings of the TAC-KBP 2011 Workshop*. NIST.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 455–465, Stroudsburg, PA, USA. Association for Computational Linguistics.

Suzanne Tamang and Heng Ji. 2012. Relabeling distantly supervised training data for temporal knowledge base population. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12, pages 25–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

Naushad UzZaman, Hector Llorens, James F. Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. 2012. TempEval-3: Evaluating events, time expressions, and temporal relations. *CoRR*, abs/1206.5333.

Xu Wei, Raphael Hoffmann, Le Zhao, and Ralph Grishman. 2012. Filling knowledge base gaps for distant supervision of relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 825–834, Stroudsburg, PA, USA. Association for Computational Linguistics.

Identification of Basic Phrases for Kazakh Language using Maximum Entropy Model

Gulila Altenbek*,+ Xiaolong Wang* Gulizhada Haisha+

*School of Computer Science and Technology, Harbin Instituteof Technology,150001,China.

+College of Information Science and Engineering, Xinjiang University,830046, China.
+The Base of Kazakh and Kirghiz Language of National Language Resource Monitoring and Research Centre Minority Languages, Xinjiang, 830046, China.

gla@insun.hit.edu.cn, gla@xju.edu.cn, wangxl@insun.hit.edu.cn

Abstract

This paper proposes the definition, classification and structure of the Kazakh basic phrases, and sets up a framework for classifying them according to their syntactic functions. Meanwhile, the structure of the Kazakh basic phrases were analyzed; and the determination of the Kazakh basic phrases collocation and extraction of the Kazakh basic phrases based on rules were followed. The Maximum Entropy (ME) model uses for the identification of the phrases from texts and achieved a result of automatic identification of Kazakh phrases with an accuracy of 78.22% based on rules System and additional artificial modification. Design feature of this ME model join rely on templates of Kazakh Word, part of speech, affixes. Experimental results show that the accuracy rate reached 87.89%.

1 Introduction

Automatic phrase identification is an important task in natural language processing. A phrase is a group of words that work together. Phrase recognition is a grammatical unit agent between words and sentences in natural language processing. Phrase identification Parser has been developed for different languages, for example, the Church's Base NP Recognition for English (Church, 1988). The rule-based Model and Maximum Entropy Model (ME) are the most commonly used technology for phrase representation and parsing.

Kazakh Language belongs to the Turkish Language group in the Altaic language family. It is an agglutinative language with word structures formed by adding derivational or inflectional affixes to root words. Phrase identification is also an indispensable part for Kazakh information processing. In the past a few year, we have put forward methods for Kazakh morphological analysis, which includes stem extraction, part of speech(POS) tagging, spelling check, etc. Recently, we are working on syntax parsing, analysis of phrase structure, automatic identification of phrase and in-depth analysis of sentence structure.

Kazakh phrases are syntactic units consisting of two or more than two words. The phrases can be classified into two categories, which are free phrase and fixed phrase. We are exploring methods which are more suitable for shallow syntactic parsing of Kazakh according to the nature of Kazakh language. The research includes a systematic study on information regularity and disambiguation of the Kazakh phrase, and automatic recognition of basic phrases of Kazakh language. We have developed a rule-based method for the automatic recognition of Kazakh basic phrases, and automatic identification of verb phrase, noun phrase and adjective phrase based on maximum entropy in Kazakh language at the same time. Moreover, the ambiguity of structures is also resolved based on rules.
This study solves the problem of Kazakh phrase recognition by providing some effective methods. This sets up a basis for further syntactic process and tree bank building. This research also provides a way to build database for various fields like knowledge acquisition, syntactic understanding, Chinese-Kazakh machine translation, the process of large-scale corpus, etc.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1007–1014, Dublin, Ireland, August 23-29 2014.

In this paper, our work focuses on identifying noun phrases, adjective phrase and verb phrases, which are the most difficult aspects of Kazakh phrase recognition analysis. This is achieved by using rules are ME method.

2 Related work

There are a variety of techniques used for phrase recognition, which include rule-based technique, statistical technique, and a combination of them. Church's (1988) approach used manual or semi-automatic annotation phrase corpus as a training corpus. Another popular method is to use a Chunk parsing for statistics model to determine the boundary (Koeling, 2000). Chunk parsing was first introduced by Abney (1991), which is one of the most widely used syntactic parsing methods. The main idea of chunk parsing lies in seeking the appropriate breakthrough point, and decomposing the full parsing problems into a syntax topology statistical structure and syntactic relations. Zhao and Huang (1998) are pioneers in Chinese phrase studies; Tsinghua University had also completed its TCT (Tsinghua Chinese Treebank) for Chinese (Zhou, 2004). The method has been also applied into studies of other languages, such as Kazakh Base NP recognition (Altenbek et al, 2009), and Uyghur Base VP Recognition by CRF (Mamatmin et al, 2012).

Maximum Entropy was first introduced to NLP area by Berger et al (1996) and Della Pietra et al. (1997). Maximum Entropy is an extremely flexible technique for linguistic modelling. It can use a virtually unrestricted and rich feature set in the framework of a probability model. It is a conditional, discriminative model and allows mutually dependent variables (Ratnaparkhi, 1999).

3 Kazakh Phase Parsing

3.1 Kazakh Morphology

Morphological analysis is an important task in natural language processing research. It was developed for different languages, included English (Porter, 1980), Finnish (Karttunen, 1983), Turkish (Oflazer, 1994; Gülşen, 2004), and Arabic (Beesley, 1996).
Comparing with other languages, the Kazakh morphological system uses a large number of suffixes and a small number of prefixes. Every word has a root, or a stem (Milat, 2003;Zhang 2004). The basic Kazakh phrase is an adjacent and non-nested phrase which does not contain recursive structure.

3.2 The Categories of Kazakh Phrase

Parsing is one of the most basic and fundamental components in natural language processing. Chunk parsing intends to obtain a fragment without thinking deeply.

A Kazakh phrase is composed of two or more than two words which connected with meaning and grammatical structure. There is only a core word in a Kazakh phrase. In the case of Kazakh, Kazakh phrases can be divided into fixed phrases and temporary phrases by the meanings of the phrases.

Abney propose the first complete description of lexical chunks system. In this study the basic phrase chunks base was found according to Abney's system. The five most common phrase in Kazakh are

NO.	Category	Explanation	Example (Kazakh)	Example (English)
1	NP	noun phrase	التىن كۇز	The golden autumn
2	VP	verb phrase	مۇراتقا جەتۋ	Achieve dreams
3	ADJP	adjective phrase	تاپ – تازا	Very clean
4	NUMP	Numeral phrases	سەگىز توعىز مىڭ	Eight & nine thousand
5	ADVP	Adverb phrase	مڭ الدىندا عى	The front of

Table 1. Part of Kazakh phrase categories.

noun phrase, verb phrase, adjective phrase, Numeral phrases, Adverb phrase as shown in table 1. Kazakh language is rich in the external morphology which shows prominent in phrase structure.

3.3 The Basic Kazakh phrase mark specification

Basic Kazakh phrase marks both its own attribute, for example part of speech, stems and affixes, and types of phrase. We used IOB Tagging to mark the start and end of chunks.

Basic Kazakh phrase	start of chunks	Inner tag of chunks	Out tag of chunks
noun phrase	B-NP	I-NP	O
verb phrase	B-VP	I-VP	
adjective phrase	B-ADJP	I-ADJP	
Adverb phrases	B-ADVP	I-ADVP	
Numeral phrase	B-NUMP	I-NUMP	

Table 2. The Basic Kazakh phrase IOB Tagging.

4 Statistics and Analysis of Kazakh Phrase Structure

Referring to modern Kazakh grammar (Milat, 2003; Dingjing Zhong. 2004), the basic rules of phrase structure of Kazakh language was summarized. The phrase structures are extracted from the corpus, and a set of rules are created based on it as well.

In the representation of basic phrase structures, the following part of speech tagging symbols are used in XML documents of Kazakh corpus: v (verb), n. (noun), adj. (adjective), num. (number), adv. (adverb), pron. (pronoun), ono. (onomatopoeia), int.(interjections), conj. (conjunction), part. (partical). The Kazakh phrases Structure divided by the function of phrases in our system are shown below.

Kazakh verb phrase structure:

1) n+v; 2) v+v; 3) adv+v; 4) adj+v; 5) v+adv; 6) v+v+v; 7) pron+v; 8) n+part+v; 9) n+conj+v; 10) ono+v; 11) int+v; 12) v+part+v; 13)v+part; 14) v+conj+v; 15)pron+part+v.

Kazakh noun phrase structure:

1) n+n; 2) n+conj+n; 3) pron+conj+pron; 4) pron+n; 5) adj+conj+adj; 6) adj+n; 7) adj+adv+n;
8) num+n; 9) v+n; 10) [ ]+n.

Kazakh adjective phrase structure:

1) adj+n; 2) adj+v; 3) adj+n+v; 4) pron+adj; 5) adv+adj+n; 6) adj+adj+n; 7) num+adv+n;

Collocations, like v+adv, n+part+v, pron+adv, v+part+v, v+part, also exist in other phrase except verb phrase. These conditions easily cause ambiguity.

5 Rule-based phrase tagging

Kazakh language has two characteristics that have to be taken into account: agglutinative morphology and rather free word order with explicit case marking.

The corpus we used in this process has been already segmented. The way we extracted stem and affix was briefly mentioned in the paper. In this paper we used the segmented results of early work, as it is not the core part of the algorithm.

Input: word segmentation (extraction stem and affix) and POS tagged corpus (test.xml);

Output: First: Phrase tagged file; Second: Phrase file;

Based on the basic rules of phrase, we have done extraction of phrases from POS tagged Kazakh corpus. The extraction process is as follows:

(a) First roughly segmented XML corpus. The common segmentation marks include semicolon, comma, full stop, exclamation mark, question mark.

(b) For the segmented data, we extract the three elements of basic phrase: part of speech (POS), affix, and the word.

(c) Look for the matched rule in the rule set. If found, save the basic phrase. Otherwise go back step 1. According to combination rules of basic Kazakh phrase, basic phrase was extracted from corpus and modified by manual work. The correct combination of basic Kazakh phrase was marked.

6 Analysis of Kazakh phrase structure ambiguity

Ambiguity computer analysis of language structure has been one of the difficulties problems. This article from the delimitation ambiguity and structural relationship is to study two aspects of phrase structure ambiguity.

One of the difficulties in Kazakh phrase research is the phrase disambiguation problem. Ambiguous reasons is word POS ambiguity, phrase boundaries is not easy to determine, POS with the same sequence, E.g. there are five ambiguous forms:

(1) VD form (v + adv)

Eg.1a : adv/تومەندەۋ v/قابىلداۋى is verb phrase. (Admission to reduce)

Eg.1b : adv/مەكشە v/قابىلدۇى is adverb phrase. (Admission to more than)

(2) ND for (n+adv, pron+adv)

Eg.2a : adv/جاڭالاۋ n/كيىمىن is verb phrase.(Change a new clothes)

Eg.2b : adv/كەرەمەت n/ناتيجەسى is adverb phrase.(Good record)

(3) NPV form （ n+part+v, pron+part+v)

Eg.3a : v /ۇيرەنۋ part /تۇرالى n /بىرلىك is verb phrase.(Learn about unity)

Eg.3b : v /دەى part / عانا n / اسان is noun phrase.(only Ashan)

(4) VPV form (v+part+v)

Eg.4a : v /كەتتى part /دە v / كەلدى is verb phrase.(came then left)

Eg.4b : v / تۇسىنۋ part /تۇرالى v / زەرتتەۋ is adverb phrase. (Relevant research to understand)

(5) VP form (v+part)

Eg.5a : part / بۇرىندىاۋ v / سويلەۋدەن is verb phrase.(Speaking before)

Eg.5b : part / جونىندە v /رەتتەۋ is verb phrase.(Organize the relevant)

For these ambiguities, we can't simply use the rules to match ways to eliminate, but rather to use maximum entropy model to solve the problem.

7 Kazakh Phrase Identification based Maximum Entropy Model

Maximum Entropy Model is an effective machine learning model which is proposed to solve the POS tagging problem, it using ME model is the ability to incorporate various features into the conditional probability. The Kazakh phrase recognition task is presented as follow.

The entropy model P: $H(p) \equiv -\sum_{x,y} p(x,y)\log(x,y)$ (1)

Note: X represents the environmental context words to be marked and y is the output.

Maximum Entropy Model： Such a model can be shown to have the following form:

$$p^* = \arg\max_{p \in C} H(p) \quad (2)$$

Goal: select a distribution p from a set of allowed distributions that maximizes H(y|X).

7.1 Feature defined

Kazakh language is an agglutinative language with word structures formed by adding derivational, inflectional affixes or suffixes to root words. The features include words, part of speech (POS), inflectional affixes of the training corpus. It seems that the features are naïve. However, these three kinds of features are the most important components of Kazakh language, and they reflect the characteristic of Kazakh language.

According to its own characteristics of a Kazakh, this feature space is defined as follows:

(1) *the word*, including the current word, the previous word and next word.

(2) *part of speech(POS)*, including the part-of-speech types of the current word, previous word and next word.
(3) *Affix ingredients*, including the current word and the word about the additional ingredient infomation.
(4) *Phrase tag* that contains the current word and the words to the right and the left two words Phrase marker.

This rule-based approach was applied to generate the maximum entropy model training corpus. Based on Kazakh linguistics, the atomic feature space is as shown in table 3.

Feature tag	Feature explanation	Feature tag	Feature explanation
W(-1)	previous one word	POS (-2) POS (-1)	POS of previous two word and POS of previous one word
W(0)	the current word	POS (-1) POS (0)	POS of previous one word and POS of the current word
W(+1)	next one word	POS (0) POS (+1)	POS of the current word and POS of next one word
W(-1) W(0)	previous one word and the current word	POS (+1) POS (+2)	POS of next one word and POS of next two word
W(0) W(+1)	the current word and next one word	POS (-2） POS (-1) POS (0)	POS of previous two word and POS of previous one word and POS of the current word
W(-1) W(0） W(+1)	previous one word and the current word and next one word	POS (-1) POS (0) POS (+1)	POS of previous one word and POS of the current word and POS of next one word
POS (-2)	POS of previous two word	POS (0) POS (+1) POS (+2)	POS of the current word and POS of next one word and POS of next two word
POS (-1)	POS of previous one word	Affix(-1)	affix of previous word
POS (0）	POS of the current word	Affix(0)	affix of current word
POS (+1)	POS of next one word	Affix(1)	affix of next one word
POS (+2)	POS of next two word		

Table 3. Atomic feature templates.

7.2 Feature selection

Basic phrases with statistical model recognition need to select a high correlation, and the Kazakh language features to train with good effect. Establish model based on rule of the language, this work selected feature through templates. After several rounds of experimental debugging, then used artificial selection, twenty one templates were selected for Kazakh verb phrase, only considered important features. According to each one's feature, templates were defined as follow.

No.	template	No.	template	No.	template
1	LPos,Cpos,RPos	8	CVP,RVP,RRVP	15	CWord,RWord
2	LLPos,Lpos,CPos	9	LVPCPosRVP	16	LPos,LVP
3	CPos,Rpos,RRPos	10	LPos, LAffix, LVP	17	RWord,RPos
4	CPos,CAffix,RPos	11	Cpos, CAffix, CVP	18	RPos,RVP
5	LPosLAffixCPos	12	CWord,RWord,RAffix	19	CPos,RPos
6	LVP,CVP,RVP	13	CWord,CPos	20	LPos,CPos
7	LLVP,LVP,CVP	14	LWord,LPos	21	LWord,LVP

Table 4. Combined feature of Kz Base VP.

In order to get the best template, this work structured and processed six template based on Table 4.

Each information function valued in the context of current word, combine the various function values into the premise of features, got the characteristics of the movement through the word tag, then it can extract features.

Template A: [RRPos, RRVP, RWord, RAffix, RPos, RVP, CPos, CVP, CWord, CAffix, LLPos, LLVP, LWord, LAffix, LPos, LVP] Observation of effects of all the words in the feature space on the result of the experiment.

Template B: [CPos, CVP, CWord, CAffix, LLPos, LLVP, LWord, LAffix, LPos, LVP] Observation of effects of left side two words of the candidate word on the result of the experiment.

Template C:[RRPos, RRVP, RWord, RAffix, RPos, RVP, CPos, CVP, CWord,CAffix] Observation of effects of right side two words of the candidate word on the result of the experiment.

Template D:[RWord, RAffix, RPos, RVP, CPos, CVP, CWord,CAffix, LWord, LAffix, LPos, LVP] Observation of effects of each side one word of the candidate word on the result of the experiment.

Template E:[RWord, RAffix, RPos, RVP, CPos, CVP, CWord, CAffix, LLPos, LLVP, LWord, LAffix, LPos, LVP] Observation of effects of left side two words and right side one word of the candidate word on the result of the experiment.

Template F:[RRPos, RRVP, RWord, RAffix, RPos, RVP, CPos, CVP, CWord, CAffix, LWord, LAffix, LPos, LVP] Observation of effects of left side one word and right side two words of the candidate word on the result of the experiment.

We selected some corpus from *Xinjiang Daily* tested on six features above, we got different influences of different characters. It shows that the C and F template give us the most highest result, namely the two words on the right have the biggest influence to the result. It proves Kazakh verb phrases are commonly at the end of the sentence.

7.3 General threshold selection

There are two general feature selection methods: incremental feature selection and feature selection of based on frequency threshold. The frequency is greater than a threshold value equal to a characteristic. Through repeating them many times, the frequency threshold value was characterized k = 5, characterized in that the use of the frequency characteristic is greater than 5.

8 Kazakh Phrase Recognition System

Kazakh phrase recognition system, which based on Maximum Entropy Model, consists of four modules, namely, pre-processing module, training module, Feature selection module, identification module. System training process as shown flow as figure 1.

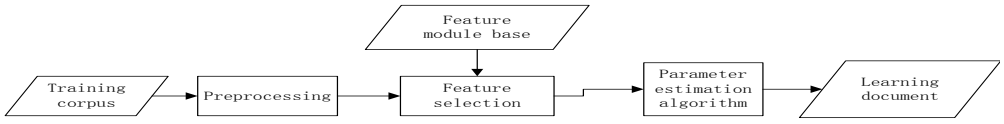


Figure 1. Training data flow diagram.

System testing process as shown flow as figure 2.

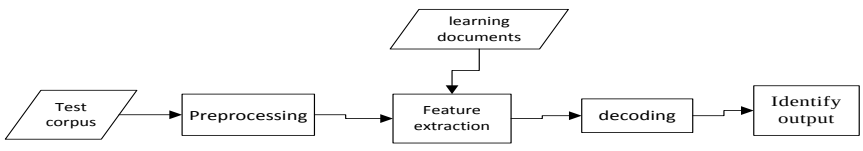


Figure 2. Testing data flow diagram.

The Kazakh basic verb phrase recognition results such as shown figure 3:

```
‹kaza_xml›
 ‹article id="&lt;*1b_1jN01001*&gt;"›
  ‹title›التاي شارۋالارى شەتەلگە شىعىپ جۇمىس ىستەپ، كىرىس تاپتى‹/title›
  ‹paragraph id="1"›
   ‹word pos="n" stem="ءتىلشى" affix="/ىمىز" var="0" vp="O"›ءتىلشىمىز‹/word›
   ‹word pos="n" stem="چىن" affix="" var="3" vp="O"›چىن‹/word›
   ‹word pos="n" stem="چىن" affix="" var="3" vp="O"›چىن‹/word›
   ‹word pos="n" stem="التاي" affix="/دان" var="4" vp="B"›التايدان‹/word›
   ‹word pos="v" stem="حابارلايدى" affix="" var="0" vp="I"›حابارلايدى‹/word›
   ‹punction›،‹/punction›
   ‹word pos="n" stem="قۇربان" affix="" var="5" vp="O"›قۇربان‹/word›
   ‹word pos="n" stem="ايت" affix="" var="1" vp="O"›ايت‹/word›
   ‹word pos="v" stem="كەل" affix="/ۋ" var="0" vp="B"›كەلۋ‹/word›
   ‹word pos="prep" stem="جونىندە" affix="" var="0" vp="I"›جونىندە‹/word›
   ‹punction›،‹/punction›
```

Figure 3. The Kazakh language basic verb phrase recognition.

By following a comprehensive analysis of Kazakh words, the following is the Kazakh shallow parsing process:

（1）Sentence：

قوڭىر كۇز كەلىپتى، قامبار سول جەرگە، قوي باعىپ كەلسە، ازىناعان كۇزدىڭ جەلى سوعىپ تۇرىپتى.

Golden autumn is coming, Hambar came to the place whish has very strong winds together with sheep.

（2）POS:

قوڭىر/n كۇز/n كەلىپتى/v، قامبار/n سول/pron جەرگە/n، قوي/n باعىپ/v كەلسە/v، ازىناعان/adj كۇزدىڭ/n جەلى/n سوعىپ/v تۇرىپتى/v.

（3）Phrase POS:

[[قوڭىر/n (Golden) كۇز/n (autumn) NP[كەلىپتى/v (is coming) VP[، [قامبار(Hamubar)/n سول/pron جەرگە/n] NP[[ازىناعان/adj كۇزدىڭ/n (the جەلى] AP]]] [كەلسە/v (came) VP[، [باعىپ/v VP] [قوي/n (sheep) NP[RP[، [سوعىپ/v very strong winds) NP[[تۇرىپتى/v (blowing. VP].

9 Experiment Results and Analysis

9.1 Data set

In this paper, according to the data set, we used the data of January 2008 of the *Xinjiang Daily* (Kazakh version) corpus. The corpus consists of the raw texts and the POS tagged XML format texts, experiments were done for phrase extraction.

9.2 Experiment results

The experiments of the accuracy rates are evaluated using as follow standard evaluation measures:

Precision: $P = \frac{a}{b} \times 100\%$ (3)

Recall $R = \frac{c}{d} \times 100\%$ (4)

F-measure $F = \frac{2 \times R \times P}{R + P}$ (5)

Note: a is number of correctly identified phrases. b is number of identified phrases. c is number of all phrases, d is number of should correct identify.

In the test corpus, there are 3000 correct tagged sentences as training data, and other 1000 sentences are for the test.

Method	Precision (%)	Recall (%)	F-measure (%)
Rule	78.22	70.01	85.25
ME	87.89	83.13	87.46

Table 5. Phrase recognition test.

10 Conclusion

This paper provided solution for identifying Kazakh basic phrases. We have tried rule-based and the maximum entropy methods. The Kazakh words, part of speech, affixes context information are used to design template of features for maximum entropy model. Based on statistical methods, higher accuracy could be obtained in the test, but it was requires more training data.

The recognition of basic Kazakh phrase could simplify sentence structure, reduce the difficulty of syntactic analyzer. This work put maximum entropy model into recognition of basic Kazakh phrase. However, there are still space for improvement on scale and accuracy rate comparing to English and Chinese. In the future, our work will focus on completing of corpus and other models.

Acknowledgments

This work is funded by the Natural Science Foundation of P.R. China (NSFC)(No.61363062, No. 61063025 and No.61272383), Science and Technology Research and Development Funds of Shenzhen City (No. JC201005260118A).

Reference

Church K. *A stochastic parts program and noun phrase parser for unrestricted text*. 1988. In Proceedings of the Second Conference on Applied Natural Language Processing. Texas, USA. 19(8):136-143.

Rob Koeling . *Chunking with Maximum Entropy Models*. 2000. Proceedings of CoNLL-2000 and LLL-2000. 109(15):139-141.

Steven Abney. *Parsing by chunks*. 1991. Dordrecht: Kluwer Academic Publishers. 257-278.

Zhao Jun and Huang Changning. 1999. *Chinese basic noun phrase structure analysis model*, Computer science . 22(2):141-146.

Qiang Zhou. 2004. Annotation scheme for Chinese Treebank, Journal of Chinese Information Processing. Vol 18(4):1-8.

Gulila Altenbek, Ruina-Sun. 2010. *Kazakh Noun Phrase Extraction based on N-gram and Rules*, International Conference on Asian Language Processing (IALP2010). Harbin, China. 305-308.

Gulila A. and Dawel, A. and Muheyat, N. 2009. *A Study of Word Tagging Corpus for the Modern Kazakh Language*, Journal of Xinjiang University. 26(4):394-401.

Zulpiya Mamatmin et al, 2012.*Uyghur Base Verb phrases Recognition* . A master's degree thesis, Beijing university of posts and telecommunications.

Adam Berger, Stephen Della Pietra, and Vincent Della Pietra. 1996. *A Maximum Entropy Approach to Natural Language* ,Processing Computational Linguistics, 22(1):39-71.

Adwait Ratnaparkhi. 1999. *Learning to parse natural language with maximum entropy models*. Machine Learning, 341(3):151-176

Porter, M.F. 1980. An algorithm for suffix stripping, Program, 14(3):130−137.

Karttunen, Lauri. 1983. KIMMO: A general morphological processor. Texas Linguistic Forum, 22:163–186.

Kemal Oflazer. 1994. Two-level description of Turkish morphology. Literary and Linguistic Computing, 9(2):137-148.

Gülşen, E. and Eşref, A. 2004. An affix stripping morphological analyzer for Turkish, Proceedings of the International Conference on Artificial Intelligence and Application, Austria, 299-304.

Beesley, K.R. 1996. Arabic finite-state morphological analysis and generation. In COLING-96, Copenhagen, 89-94.

Milat, A. 2003. Modern Kazakh language, Xinjiang People's press, China.

Dingjing Zhang. 2004. Practical Grammar of Modern Kazakh Language. Beijing: Central University for Nationalities Press.

Collecting Bilingual Audio in Remote Indigenous Communities

Steven Bird
Dept of Computing and Information Systems, University of Melbourne; Linguistic Data Consortium, University of Pennsylvania

Lauren Gawne
Department of Linguistics and Multilingual Studies, Nanyang Technological University, Singapore

Katie Gelbart
School of Oriental and African Studies, University of London

Isaac McAlister
Department of Languages, Literatures, and Cultures, University of Massachusetts, Amherst

Abstract

Most of the world's languages are under-resourced, and most under-resourced languages lack a writing system and literary tradition. As these languages fall out of use, we lose important sources of data that contribute to our understanding of human language. The first, urgent step is to collect and orally translate a large quantity of spoken language. This can be digitally archived and later transcribed, annotated, and subjected to the full range of speech and language processing tasks, at any time in future. We have been investigating a mobile application for recording and translating unwritten languages. We visited indigenous communities in Brazil and Nepal and taught people to use smartphones for recording spoken language and for orally interpreting it into the national language, and collected bilingual phrase-aligned speech recordings. In spite of several technical and social issues, we found that the technology enabled an effective workflow for speech data collection. Based on this experience, we argue that the use of special-purpose software on smartphones is an effective and scalable method for large-scale collection of bilingual audio, and ultimately bilingual text, for languages spoken in remote indigenous communities.

1 Introduction

Past the top one to three hundred economically significant languages, there are few prospects for resourcing the production of annotated corpora. Advances in natural language processing have relied on such corpora – including treebanks and wordnets – though they are expensive to produce and depend on substantial prior scholarship on the language. An alternative is to collect bilingual aligned text, relating a low-resource language to a high-resource language, and then infer lexical and syntactic information from the high-resource language via alignments (Abney and Bird, 2010; Baldwin et al., 2010; Palmer et al., 2010; Das and Petrov, 2011).

This approach only works for written languages. Over half the world's languages lack a literary tradition. In some cases they have a writing system, but it is not in regular use and so these languages remain effectively unwritten. Collecting data for unwritten languages necessarily involves speech.

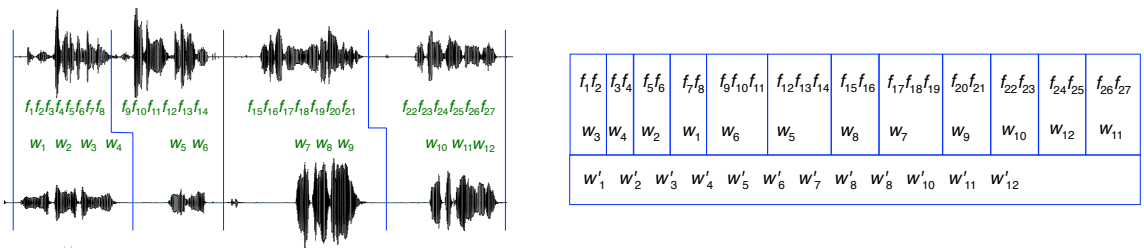


Figure 1: The Vision: phrase-aligned bilingual audio from an unwritten language to a language of wider communication, along with extracted acoustic features and crowdsourced transcription (left); interlinear glossed text with word segmentation, word-level glosses, and sentence-level translations (right).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1015–1024, Dublin, Ireland, August 23-29 2014.

While the physical isolation of these languages presents a logistical challenge, it is still possible to collect hundreds of hours of speech using mobile devices (de Vries et al., 2014). Furthermore, there are promising signs that natural language processing methods and speech processing methods can be integrated (Zhang et al., 2004; Dredze et al., 2010; Vu et al., 2011; Siniscalchi et al., 2012; Lee and Glass, 2012). Thus, the challenge is to collect substantial quantities of bilingual aligned audio, transcribe the translations, extract phonetic features from the source language and, ultimately, produce bilingual aligned text (see Figure 1).

We have chosen to focus on endangered languages because of the interesting and difficult challenges that are faced in collecting data. However, the resource problem exists even for vital languages having large speaker populations. For example, Shanghainese (Wu) is spoken by 77 million people in China, but is almost never written down because written Chinese is based on Mandarin; Oromo is spoken by 17 million people in Ethiopia, but few of its speakers know how to write it. Such languages are collectively spoken by billions, yet remain seriously under-resourced. Thus, while our focus is on endangered languages, the approach applies to under-resourced languages in general.

Several other promising approaches to the problems raised by endangered languages are being actively pursued in computational linguistics, however they typically focus on written language with annotations, often with the goal of making optimal use of human expertise (Probst et al., 2002; Levin et al., 2006; Clark et al., 2008; Palmer et al., 2010; Bender et al., 2012; Beale, 2012; Bender et al., 2013). The research reported here is unique in its focus on securing spoken language data in a form and on a scale that will be usable even once the languages in question are no longer spoken.

This paper explores ways that networked smartphones can be used for collecting bilingual aligned audio. We have used a prototype Android application for collecting audio and phrase-aligned translations (or consecutive interpretations). We took a set of phones to villages in Brazil and Nepal, and worked with languages Tembé, Nhengatu and Kagate. We visited at the invitation of the local communities and collaborated closely with them in each stage of the process, including setting the goals and agreeing on the form of dissemination, cf. (Rice, 2011). We compiled small collections of recorded texts and translations in each language.

We describe and evaluate this novel resource-creation activity, and argue that it can be used effectively for large-scale collection of bilingual aligned audio. This paper is organised as follows. In section 2, we give an overview of the mobile software. The next three sections report the activities in the three communities. We reflect on the work in section 6.

2 Mobile applications for recording and translating endangered languages

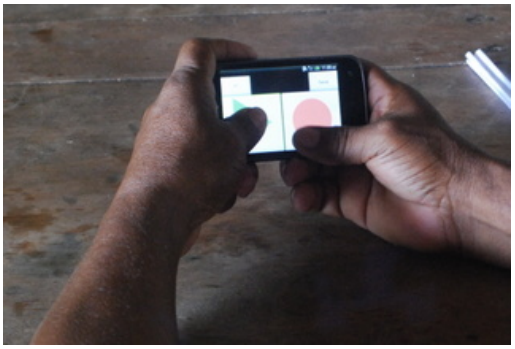
Smartphones are proliferating: they are part of the vanguard of technologies that make it into many isolated communities. Even in the most remote villages, many people own a mobile phone, keep it on their person, and are able to get it charged when mains electricity is unreliable or non-existent. These phones can be inexpensive (US$100-200) and some models have sufficient audio quality to be useful for speech data collection. With suitable software it is possible to collect metadata along with recordings, including location, date, the identity of the speaker, and possibly some information about the content such as the title and genre. The networking capability of a smartphone facilitates wireless sharing and backup.

The speech collection task calls for a variety of individual contributions. The best speakers of the language are not necessarily the best translators; they may be monolingual. Similarly, the best translators may not be the best transcribers; they may be illiterate. Thus, for reasons of skill, not just scale, we need to involve a whole team of people in the data collection activity. In the medium term, we assume that this work would take place under the supervision of a linguist who provides hardware and training, and who monitors the balance of the collection, including coverage of various discourse types, getting everything translated, and so forth).

Aikuma is open source software that supports recording of speech directly to phone storage (Hanke and Bird, 2013; Bird et al., 2014). Recordings are synchronized with other phones that are connected to the same WiFi LAN, so that any user can listen to recordings made on any phone in the same local

network. A user can "like" a recording to improve its overall ranking. A user can also provide a phrase-by-phrase spoken translation of the recording, using the interface shown in Figure 2. This functionality is based on the protocol of "Basic Oral Language Documentation" (Reiman, 2010; Bird, 2010).

(a) Recording a Tembé narrative



(b) Translating Tembé into Portuguese

Figure 2: Recording and translating using the Aikuma Android app

Users press and hold the left *play* button to hear the next segment of audio source. They can press it multiple times to hear the same segment over again. Once ready, they press the right *record* button and speak a translation of the source. This process continues until the source has been fully translated. It generates a second audio file that is time-aligned with the source (cf Figure 1). The app supports playback of the source, the translation, and the source with interleaved translation.

Aikuma maintains a database of speakers and synchronizes this to the other phones along with the recordings and titles, and keeps track of which speaker provided which recording. In this way, basic metadata resides with the recordings, and recordings are effectively backed up several times over. If the contents of one phone are periodically archived, then we have a permanent copy of all the recordings and metadata from all of the phones.

We used HTC Desire C and HTC Desire 200 phones which cost US$160 each. We chose these phones for their support of Android 4 and their recording quality. Unlike a professional audio set-up, mobile phone audio recording includes built-in noise suppression that is optimised for near-field voice sources and attenuates background noise. The software stores audio in uncompressed 16kHz 16-bit mono. The quality of the audio from these phones is more than sufficient to support phonetic analysis (Bettinson, 2013). We expect these materials to be considered of archival quality in those cases where the original recording environment was quiet and where the content itself has linguistic and cultural value.

Another advantage of smartphones compared with professional recording equipment is ease of recharging. Many remote indigenous communities without mains electricity are still able to keep phones charged with the help of generators and car batteries. By choosing to use mobile phones, we can piggy-back on the existing infrastructure.

The cost and usability of smartphones relative to professional recording equipment makes it easy to consider giving them out to people to make recordings in their own time. Apart from significantly increasing the amount of recorded and translated material that a linguist can collect, this gives speakers direct control over the content and context of the recordings, and it may lead to the collection of more naturalistic materials. In some cases, speakers already own an Android phone and can simply install the software and get started.

In the following three sections we report on our experience of using these phones with indigenous communities in the Amazon and the Himalayas.

3 Tembé, Pará State, Brazil

The Tembé language is spoken by approximately 150 people amongst a larger community numbering about 1,500, in a group of villages in the Reserva Alto Rio Guam in the vicinity of Paragominas in the Pará state of Brazil. Bird, Gelbart, and McAlister spent five days in the village of Cajueiro (*Akazu'yw*),

the gateway to several other Tembé villages that can only be accessed by river. Like many Indian villages, Cajueiro is laid out around a soccer field. The village was connected to the electricity grid ten years ago.

We recorded 14 texts from 8 speakers, mostly personal narratives but also a song and a dialogue. Most texts were orally translated; some were translated twice. Of two hours of source audio, 35 minutes were orally translated, producing an extra 25 minutes of audio.

Our visit to Cajueiro is mostly interesting for the great variety of unanticipated challenges, and how we were still able to use the platform to collect data.

Previous contact with the Tembé community was mediated by staff at the Goeldi Museum in Belém. The Tembé community had been discussing prospects for installing an antenna in Cajueiro to enable an Internet connection. On arrival, the chief asked about our plans to set up Internet access, and we explained that we were not able to do this because there was no signal for 100km. After this, the chief lost interest in our activities and we were not able to hold a village meeting as we had hoped, in order to discuss our work, invite participation, and demonstrate the use of the technology for recording and translation. Instead, we could only work one-on-one.

Our first 24 hours in the village was spent on video documentation of a coming-of-age ceremony. More elaborate versions of this ceremony had been filmed in the past, so there was minimal documentary value in recording this event. However, it was the basis for our invitation to the village, cf. (Crowley, 2007, 80), and it enabled us to meet the whole community and to observe the limited social interaction, almost exclusively conducted in Portuguese.

In the following days, we went around the village showing the app to people, explaining our work, playing existing recordings in Tembé and other languages, and trying to find fluent speakers who were motivated to preserve Tembé linguistic heritage. Few people claimed to be fluent and we only found six who were willing to be recorded, all men. No women would consent to being recorded until a Tembé man, trained as a computer technician, learnt how to use the app and took a handset and found two female speakers and recorded them. They were in their thirties, less confident with the language, and could only read haltingly from a small storybook. For the fluent speakers we were able to find, the documentary activity proceeded naturally; they easily recounted histories and gave phrase-by-phrase translations. We prepared a selection from our recordings and made audio CDs to give away for people to play on their personal stereos.

We experienced a variety of technical difficulties with the smartphones, none of which had been apparent during lab testing. The most obvious were due to people's unfamiliarity with smartphones. Signing in required entering the participant's name using a touchscreen keyboard, then selecting an ISO 639 language code via a search interface, then taking a photo using the phone. The photo could not be taken easily by the participant as the phones lacked a front-facing camera. Consequently, we generally took care of these tasks on behalf of speakers. Similarly, upon completion of a recording, the participant was prompted to enter a name for the recording, and we would reclaim the phone and enter a title after a brief discussion with the participant about a suitable choice.

Further problems concerned the translation task. A couple of participants began to give a Portuguese paraphrase immediately on finishing a story. Despite the obvious value of capturing an immediate paraphrase from the same speaker, the software was not designed for this and we had no way to capture the paraphrase as a separate audio file and link it back to the original. The thumb-controlled interface (Figure 2b) was also slightly problematic. Often a speaker would still be holding down the play button with his left thumb at the moment he went to press the record button with his right thumb. Sometimes, speakers would begin to speak and then notice that playback was still continuing, and only then release the play button. By the time they had pressed the record button again, they had already spoken a word or two, and this speech was not captured by the app. This problem happened often enough to interfere with the flow of the translation task. Possible solutions are to have the controls operated by a single thumb, or else to change the behavior of the app so that the most recent thumb gesture overrides a existing button press. Several other interactional issues with the software were identified and resolved with similar minor changes to behavior.

A final set of issues concerned dissemination. Many Indian villages are now equipped with computer

rooms and have desktop machines with CD burners, though mains electricity may be intermittent, or else depend on a generator. We were able to transfer files from the phones to a local machine using a USB connection, though it was a slow process to identify the recordings of interest to the participants and to compile an audio CD. Instead, we realised that any user of a phone should be able to export selected recordings to a local folder that could be burnt to CD.

The key problem for us, however, was lack of participation. The main reason for this, we believe, was the limited local interest in the Tembé language. A secondary factor was the misunderstanding about our contribution ("bringing the Internet") and the fact that the product, a CD of stories, was not necessarily something that the community wanted.

4 Nhengatu, Amazonas State, Brazil

The Nhengatu language is a creole spoken by 10,000 people across a wide area, including the village of Terra Preta, 50km NW of Manaus. Nhengatu used to be the language of communication amongst Indians from different tribes along the Rio Negro, and between Indians and non-Indians in the Brazilian Amazon. Although most of the inhabitants of Terra Preta are ethnically Baré, the only indigenous language spoken in the village is Nhengatu. Younger generations are monolingual in Portuguese. Unusually, there are also some non-Indians living in the village. The villagers were open to receiving us, partly due to their proximity to Manaus and the fact they were accustomed to meeting tourists and showing white people around and selling handcrafts. Compared with Cajueiro, there was a stronger sense of community in Terra Preta: on weekends they would have breakfast together in a communal meeting place, and agree on community service tasks for the weekend.

We made a preliminary visit and presented our work at a public meeting. We called for a volunteer to tell a story to the group and then invited another volunteer to provide an oral translation. Both individuals did a perfect job even though neither one had used the software before. One of them, a former village chief, addressed the group and explained the significance of our work. He then asked if we would help in the preparation of a DVD. Since we did not have the necessary equipment, we offered to create a bilingual storybook instead. They agreed, and said this could be used in their local school. We had already intended to propose this as our contribution to the community after our experience with Tembé, where most people did not grasp the value of us only leaving audio recordings. A booklet would be a natural extension to our documentary goals, and it offered to draw in the whole community including the children who could provide illustrations.

Three weeks later, once the necessary approvals had been obtained, we arrived in Terra Preta and launched our activities with another public meeting. At this meeting, and again at public meetings on the following two mornings, we invited anyone who was interested to take a phone and record a story. Sometimes a storyteller held a phone while addressing a small group (often involving children), and recounted a folktale.

After three days, we recorded 35 texts from nine speakers (including two children), mostly folklore and personal narratives. Most texts were orally translated. Of 2.5 hours of source audio, approximately one hour of recordings were orally translated (some two or more times), producing an extra two hours of audio. Seven short texts by children or directed at children were delivered in Portuguese, and we did not translate these back into Nhengatu.

During the second half of the visit, four men who were literate in Nhengatu joined us in the task of transcribing the stories, focussing on those that would be most interesting for inclusion in the storybook. They worked in parallel, playing back the recordings on the phones, transcribing them on paper, then bringing the sheets back to be typed and proof-read. This work was arduous, continuing through the heat of the day, but they were keen to process as many stories as possible.

Two weeks after our visit, we published a small booklet of stories and translations and sent copies back to the village, and posted a digital copy in the Internet Archive (Bird et al., 2013).

We encountered some additional technical difficulties that we had not experienced in Cajueiro. First, a bug in the recording app which appeared on the last day caused one recording to overrun and produced a three hour (350MB) file. After this, WiFi synchronisation was too slow to be effective, and it was

necessary to perform synchronisation manually, copying the files from all phones onto a laptop, then copying the collection back onto each phone. Second, the presence of an audience for some stories encouraged the storyteller to speak loudly. Since speakers were holding the phone close to their mouths, this resulted in clipped audio. Third, at the height of our intensive transcription and translation process, we needed to keep track of the activities of several participants, and created a checklist. Finally, there was an issue with the power supply. Unlike Cajueiro, Terra Preta is not attached to the electricity grid, but it has a generator which is turned on for four hours every evening, and sometimes during the mornings for brief periods. We could use this to keep the phones charged and to power the router for long enough to synchronise the phones a couple of times each day. But the village became very noisy when power was available, thanks to an abundance of stereo systems and power tools, and this made it difficult to get good quality recordings during these times.

In spite of these problems, there were some successes. The most notable was that participants took no more than a minute to become adept with the recording functionality and the thumb-controlled oral translation functionality (Figure 2b). Second, the availability of multiple networked recording devices meant that we could collect materials in parallel. For example, we could discuss a story we wanted to record and then send several people off at the same time to record their own versions. Then they could synchronise their recordings and hear what each other said. Finally, automatic synchronisation greatly facilitated concurrent transcription activities. We could assign people to transcribe or translate a particular source recording without having to keep track of device it had been recorded with: it was already available on all of the phones.

Figure 3: Transcribing a spoken translation

5 Kagate, Ramechhap district, Nepal

A third field test with a later version of the app was undertaken in Nepal. Kagate, known to its speakers as Syuba, is a Tibeto-Burman language spoken by around 1,500 people in the Ramechhap district, east of Kathmandu. Handsets with the Aikuma app were taken by Gawne and were deployed in parallel, in the context of a project to video record traditional folk narratives and history. Twelve original recordings were made, totalling 80 minutes. Four of these recordings were translated into Nepali, and two recordings were also carefully "respoken" to aid later written transcription (Woodbury, 2003). Although the recordings represent a more modest total than at other fieldsites, this field test demonstrates that Aikuma can operate in conjunction with, and to the benefit of, more traditional field methods. A number of challenges were addressed.

The first challenge was the lack of mains electricity, with the village only having a number of small solar panels for charging mobile phones and running small lights. Much like at the Nhengatu site, mobile phones enabled work to proceed in the absence of mains electricity. Indeed, this was greatly beneficial because it meant that more recordings could be made without rapidly depleting the video camera battery, which required charging at a village a one hour walk away. The lack of proximal mains electricity meant that it was not possible to run the router and synchronise the data on each phone. As a result of this (and participation issues discussed below) the researcher only kept two devices in use at a time, making it easier to keep track of what was on each device. This field trip demonstrated that even without the data synchronization feature Aikuma is still a useful fieldwork tool.

The second challenge was fostering participation. As a number of anthropologists working in related communities have observed, the centre of village life for Kagate people is the household (Fürer-Haimendorf, 1964; Desjarlais, 1992). Relationships beyond this are negotiated through extended familial relations of reciprocity. Therefore, there were no opportunities to arrange community meetings as in

Terra Preta, or even to find an individual who was an officially designated leader. As a result, much time was spent engaging a small number of enthusiastic participants and working with them to engage other members of the community through existing social networks. The benefit of the mobile devices was that they could be carried about and then demonstrated to people during a lull in other activities. Because of this portability and ease of demonstration, the mobile phones became a key part of negotiations with all participants, even those who the community members wanted to video record. Having the handsets meant that we could immediately show people the outcome of a recording session. Sometimes, even after this demonstration, people were reluctant to participate in recording with video cameras or phones. We took this as a positive sign that participants had a better level of informed consent with which to make this choice than they otherwise would have. Many community members were reluctant to take the phones, as even basic smartphones that we chose for their affordability are an expensive commodity and out of the price bracket of many. A small number of people became comfortable enough to take the phones away to work with, but would return them immediately after a specific task had been completed. With a longer period of presence in the village it is likely more people would become more comfortable with the process.

The final issue, like at other sites, concerned the process of saving recordings once they had been made. Processes that are taken for granted with some audiences, like naming a recording, presume a great deal of cultural knowledge about iconography, the layout of keyboards, and spelling conventions. It was only on the final day that one of the more frequent participants saved a file without assistance. Fortunately, an import feature had been built into the app, which meant that when participants returned with files that they had not managed to save they could still be loaded into the list. While some of the issues faced can be overcome through further refining the design, others are useful educational tools to help familiarize participants with key features of digital literacy.

Throughout the above discussion we have touched on some benefits to using Aikuma at this field site. There are some other advantages that are also worth noting. The first is that the portability of the handsets meant that there was a wider range of participants recorded. The limited electricity available for the parallel video documentation, and community attitudes about who was a suitable participant in that work, meant that only a small section of the community (mostly older males) would have been documented. The lower formality of using the phones, compared with a bulky video camera, meant that people also felt quite relaxed, often telling stories with an audience present.

The use of phones also meant that there were fewer missed opportunities for recording. One evening we used the phones when the light was too poor for video. Another morning when the researcher was unwell, she gave one of the handsets to a member of the community who recorded some traditional stories with an older man who had not been able to remember them the day before. On yet another occasion, a man took one of the handsets away and recorded a translation while the researcher was filming a video with another participant. Although the linguist was still needed for the saving of recordings, people became less dependent on her presence to do their own documentation work.

6 Discussion

Reflecting on our experience in the Tembé, Nhengatu, and Kagate communities, further issues warrant discussion.

The mobile device was a major attraction. People gathered round to see how it was used, then explained it to others in their native language. They brought elders to see the work, and encouraged them to tell stories. This impact convinced us that the mobile phone is an effective platform for engaging with participants and helping them quickly grasp the collection and dissemination aspects of language documentation work, cf. (Rice, 2011). Note that the phones were not equipped with SIM cards, and so there was no distraction of them being used for voice calls or for downloading extraneous software.

However, the device was also an obstacle. Although some people had used smartphones, few had experienced touchscreens. Creating a user profile required entering a name using the touchscreen keyboard. It seemed like overkill to train individuals to use a keyboard and to go through a process they would only perform once. Moreover, the language selection process displayed a searchable list of 7,000 languages,

and it would have been easier to have a small selection of local languages to choose from. In Tembé, the man who was trained as a computer technician learned to create user profiles for other people. By the time of the Kagate experiment, we added support for default languages, and set these as Kagate and Nepali. This simplified the task, though it also meant that we did not capture information about people's competencies in other languages. These issues with the device only occurred at the outset, and highlight the need to simplify the metadata collection process. The impact of the problem would be reduced with improved software design.

The device helped with the process of obtaining informed consent. We played an existing recording, either one collected during an earlier phase of documenting the language, or one from another endangered or extinct language. In this way we communicated the idea that language recordings can be preserved and transmitted over distance and time, even once the language is no longer spoken. We also asked what people thought about the idea of others hearing their language, and they were generally enthusiastic. In the case of a further Brazilian language, one community leader asked for substantial donations of hardware and another cited intellectual property concerns, and so we did not record this language. A related open issue concerns the process for *documenting* informed consent, particularly when working with monolingual speakers.

Most of the collected material consisted of personal narratives, folklore, and a limited amount of singing. Other discourse types that we did not collect include dialogue, oratory, and procedural discourse, cf. (Johnson and Aristar Dry, 2002). On many occasions, people listened to a traditional narrative and then asked to recount their own version. Consequently, we see the possibility for achieving substantial lexical overlap in recordings by different speakers, which could help with speech modelling, dialect identification, and lexicon production.

7 Conclusions

We have investigated the use of Aikuma, an Android app designed for recording and translating unwritten languages. We taught members of indigenous communities in Brazil and Nepal to use smartphones for recording spoken language and for orally interpreting it into the national language, and we collected a sample of bilingual phrase-aligned speech in the languages. We collected approximately 8.5 hours of audio, approximately 100,000 words, and in the process, we demonstrated that the platform is an effective way to engage indigenous communities in the task of building phrase-aligned bilingual speech corpora. The built-in networking capability of the phone was used to good effect in Nhengatu for leveraging the contribution of multiple members of the community who have differing linguistic aptitudes.

We identified several areas for additional functionality: support for adding a paraphrase as soon as a story has been told; support for exporting playlists to CD; a checklist that shows which recordings have been translated; permitting handwritten transcriptions to be photographed and linked back to the original audio; and redesigning the interface to remove some remaining English prompts and confusing icons. These and other enhancements are being developed in our open source project.[1]

Above all, we have found that this approach to linguistic data collection greatly facilitates work on indigenous languages that are falling out of use. It bypasses the need for expensive equipment by piggybacking on the burgeoning adoption of mobile phones and wireless broadband networks. We are optimistic about the prospects of using this approach to collect substantial new corpora for supporting linguistic research and language technology development, even for some of the most isolated linguistic communities in the world.

Acknowledgments

This research was supported by NSF Award 1160639 *Language Preservation 2.0: Crowdsourcing Oral Language Documentation using Mobile Devices* (Bird and Liberman), ARC Award 120101712 *Language Engineering in the Field* (Bird), and Firebird Foundation project *Documenting the Traditional Songs and Stories in Kagate, a language of Nepal* (Gawne). Bird, Gelbart, and McAlister are grateful to Dr Denny Moore and the Goeldi Museum (Belém) for facilitating their work in Brazil.

[1] `https://github.com/aikuma`

References

Steven Abney and Steven Bird. 2010. The Human Language Project: building a universal corpus of the world's languages. In *Proceedings of the 48th Meeting of the Association for Computational Linguistics*, pages 88–97. Association for Computational Linguistics.

Timothy Baldwin, Jonathan Pool, and Susan Colowick. 2010. PanLex and LEXTRACT: Translating all words of all languages of the world. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 37–40, Beijing, China.

Stephen Beale. 2012. Documenting endangered languages with Linguist's Assistant. *Language Documentation and Conservation*, 6:104–134.

Emily Bender, Robert Schikowski, and Balthasar Bickel. 2012. Deriving a lexicon for a precision grammar from language documentation resources: A case study of Chintang. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 247–262.

Emily Bender, Michael Wayne Goodman, Joshua Crowgey, and Fei Xia. 2013. Towards creating precision grammars from interlinear glossed text: Inferring large-scale typological properties. In *Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 74–83. Association for Computational Linguistics.

Mat Bettinson. 2013. The effect of respeaking on transcription accuracy. Honours Thesis, Dept of Linguistics, University of Melbourne.

Steven Bird, Katie Gelbart, and Isaac McAlister, editors. 2013. *Fábulas de Terra Preta*. Internet Archive.

Steven Bird, Florian R. Hanke, Oliver Adams, and Haejoong Lee. 2014. Aikuma: A mobile app for collaborative language documentation. In *Proceedings of the Workshop on the Use of Computational Methods in the Study of Endangered Languages*. Association for Computational Linguistics.

Steven Bird. 2010. A scalable method for preserving oral literature from small languages. In *Proceedings of the 12th International Conference on Asia-Pacific Digital Libraries*, pages 5–14.

Jonathan Clark, Robert Frederking, and Lori Levin. 2008. Toward active learning in data selection: Automatic discovery of language features during elicitation. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation*.

Terry Crowley. 2007. *Field Linguistics: A Beginner's Guide*. Oxford University Press.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 600–609. Association for Computational Linguistics.

Nic de Vries, Marelie Davel, Jaco Badenhorst, Willem Basson, Febe de Wet, Etienne Barnard, and Alta de Waal. 2014. A smartphone-based ASR data collection tool for under-resourced languages. *Speech Communication*, 56:119–131.

Robert R. Desjarlais. 1992. *Body and emotion: the aesthetics of illness and healing in the Nepal Himalayas*. Philadelphia: University of Pennsylvania Press.

Mark Dredze, Aren Jansen, Glen Coppersmith, and Ken Church. 2010. NLP on spoken documents without ASR. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 460–470. Association for Computational Linguistics.

Christoph von Fürer-Haimendorf. 1964. *The Sherpas of Nepal: Buddhist highlanders*. London: John Murray.

Florian R. Hanke and Steven Bird. 2013. Large-scale text collection for unwritten languages. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*, pages 1134–1138. Asian Federation of Natural Language Processing.

Heidi Johnson and Helen Aristar Dry. 2002. OLAC discourse type vocabulary. `http://www.language-archives.org/REC/discourse.html`.

Chia-ying Lee and James Glass. 2012. A nonparametric bayesian approach to acoustic model discovery. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 40–49. Association for Computational Linguistics.

Lori Levin, Jeff Good, Alison Alvarez, and Robert Frederking. 2006. Parallel reverse treebanks for the discovery of morpho-syntactic markings. In Jan Hajič and Joakim Nivre, editors, *Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories*, pages 103–114.

Alexis Palmer, Taesun Moon, Jason Baldridge, Katrin Erk, Eric Campbell, and Telma Can. 2010. Computational strategies for reducing annotation effort in language documentation. *Linguistic Issues in Language Technology*, 3:1–42.

Katharina Probst, Lori Levin, Erik Peterson, Alon Lavie, and Jaime Carbonell. 2002. MT for resource-poor languages using elicitation-based learning of syntactic transfer rules. *Machine Translation*, 17(4):225–270.

Will Reiman. 2010. Basic oral language documentation. *Language Documentation and Conservation*, 4:254–268.

Keren Rice. 2011. Documentary linguistics and community relations. *Language Documentation and Conservation*, 5:187–207.

S.M. Siniscalchi, Dau-Cheng Lyu, T. Svendsen, and Chin-Hui Lee. 2012. Experiments on cross-language attribute detection and phone recognition with minimal target-specific training data. *IEEE Transactions on Audio, Speech, and Language Processing*, 20:875–887.

Ngoc Thang Vu, Franziska Kraus, and Tanja Schultz. 2011. Rapid building of an ASR system for under-resourced languages based on multilingual unsupervised training. In *Interspeech*, pages 3145–3148.

Anthony C. Woodbury. 2003. Defining documentary linguistics. In Peter Austin, editor, *Language Documentation and Description*, volume 1, pages 35–51. London: SOAS.

Ruiqiang Zhang, Genichiro Kikui, Hirofumi Yamamoto, Taro Watanabe, Frank Soong, and Wai Kit Lo. 2004. A unified approach in speech-to-speech translation: integrating features of speech recognition and machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1168–1174.

Inclusive yet Selective: Supervised Distributional Hypernymy Detection

Stephen Roller*, **Katrin Erk**†, **Gemma Boleda**†
* Department of Computer Science
† Department of Linguistics
The University of Texas at Austin
roller@cs.utexas.edu, katrin.erk@mail.utexas.edu,
gemma.boleda@upf.edu

Abstract

We test the Distributional Inclusion Hypothesis, which states that hypernyms tend to occur in a superset of contexts in which their hyponyms are found. We find that this hypothesis only holds when it is applied to relevant dimensions. We propose a robust supervised approach that achieves accuracies of .84 and .85 on two existing datasets and that can be interpreted as selecting the dimensions that are relevant for distributional inclusion.

1 Introduction

One of the main criticisms of distributional models has been that they fail to distinguish between semantic relations: Typical nearest neighbors of *dog* are words like *cat*, *animal*, *puppy*, *tail*, or *owner*, all obviously related to *dog*, but through very different types of semantic relations. On these grounds, Murphy (2002) argues that distributional models cannot be a valid model of conceptual representation. Distinguishing semantic relations are also crucial for drawing inferences from distributional data, as different semantic relations lead to different inference rules (Lenci, 2008). This is of practical import for tasks such as Recognizing Textual Entailment or RTE (Geffet and Dagan, 2004).

For these reasons, research has in recent years started to attempt the detection of specific semantic relationships, and current results suggest that distributional models can, in fact, distinguish between semantic relations, given the right similarity measures (Weeds et al., 2004; Kotlerman et al., 2010; Lenci and Benotto, 2012; Herbelot and Ganesalingam, 2013; Santus, 2013). Because of its relevance for RTE and other tasks, much of this work has focused on hypernymy. Hypernymy is the semantic relation between a superordinate term in a taxonomy (e.g. *animal*) and a subordinate term (e.g. *dog*).

Distributional approaches to date for detecting hypernymy, and the related but broader relation of lexical entailment, have been unsupervised (except for Baroni et al. (2012)) and have mostly been based on the Distributional Inclusion Hypothesis (Zhitomirsky-Geffet and Dagan, 2005; Zhitomirsky-Geffet and Dagan, 2009), which states that more specific terms appear in a subset of the distributional contexts in which more general terms appear. So, *animal* can occur in all the contexts in which *dog* can occur, plus some contexts in which *dog* cannot – for instance, *rights* can be a typical cooccurrence for *animal* (e.g. "animal rights"), but not so much for *dog* (e.g. #"dog rights").

This paper takes a closer look at the Distributional Inclusion Hypothesis for hypernymy detection. We show that the current best unsupervised approach is brittle in that their performance depends on the space they are applied to. This raises the question of whether the Distributional Inclusion Hypothesis is correct, and if so, under what circumstances it holds. We use a simple supervised approach to relation detection that has good performance (accuracy .84 on BLESS, .85 on the lexical entailment dataset of Baroni et al. (2012)) and works well across different spaces.[1] Furthermore, we show that it can be interpreted as selecting dimensions for which the Distributional Inclusion Hypothesis does hold. So, our answer is to propose the *Selective Distributional Inclusion Hypothesis*: The Distributional Inclusion Hypothesis holds, but only for relevant dimensions.

[1] Code and data are available at http://stephenroller.com/research/coling14.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1025–1036, Dublin, Ireland, August 23-29 2014.

2 Background

Distributional models. Distributional models represent a word through the contexts in which it has been observed, usually in the form of a vector representation (Turney and Pantel, 2010). A target word is represented as a vector in a high-dimensional *space* in which the *dimensions* are context items (for example, other words) and the coordinates of the vector indicate the target's degree of association with each context item. In this paper, we also use dimensionality reduced spaces in which dimensions do not stand for individual context items anymore.

Pattern-based approaches to inducing semantic relations. Early work on automatically inducing semantic relations between words, starting with Hearst (1992), uses textual patterns. For example, "[NP_1] and other [NP_2]" implies that NP_2 is a hypernym of NP_1. Pattern-based approaches have been applied to meronymy (Berland and Charniak, 1999; Girju et al., 2003; Girju et al., 2006), synonymy (Lin et al., 2003), co-hyponymy (Snow et al., 2005), hypernymy (Cimiano et al., 2005), and several relations between verbs (Chklovski and Pantel, 2004). Pantel and Pennachiotti (2006) generalize the idea to a wide variety of relations. Turney (2006) uses vectors of patterns to determine similarity of semantic relations. A task related to semantic relation induction is the extension of an existing taxonomy (Buitelaar et al., 2005). Snow et al. (2006) do this by using hypernymy and co-hyponymy detectors.

Lexical entailment, hypernymy, and the Distributional Inclusion Hypothesis. Weeds et al. (2004) introduce the notion of *distributional generality*, where v is distributionally more general than u if u appears in a subset of the contexts in which v is found, and speculate that hypernyms (v) should be more distributionally general than hyponyms (u). Zhitomirsky-Geffet and Dagan (2005; 2009) introduce the term *Distributional Inclusion Hypothesis* for the idea that distributional generality encodes hypernymy or the more loosely defined relation of *lexical entailment*.

Weeds and Weir (2003) measure distributional generality using a notion of precision (eq. 1). Here and in all equations below, u is the narrower term, and v the more general one. Abusing notation, we write u for both a word and its associated vector $\langle u_1, \ldots, u_n \rangle$. Kotlerman et al. (2010) predict lexical entailment with the *balAPinc* measure, a modification of the Average Precision (AP) measure (eq. 2). The general notion is that scores should increase with the number of dimensions of v that u shares, and also give more weight to the highly ranked dimensions (i.e. largest magnitude) of the narrower term u. This is captured in *APinc* by computing precision $P(r)$ at every rank r among u's dimensions – where precision is the fraction of dimensions shared with v –, and weighting by the rank of the same dimension in the broader term, $rel'(v, r, u)$. The final measure, *balAPinc*, smooths using the *LIN* similarity measure (Lin, 1998). (We only sketch this measure here due to its complexity; details are given in Kotlerman et al. (2010).)

$$1(x) = \begin{cases} 1 \text{ if } x > 0; \\ 0 \text{ otherwise} \end{cases}$$

$$WeedsPrec(u, v) = \frac{\sum_{i=1}^{n} u_i \cdot 1(v_i)}{\sum_{i=1}^{n} u_i} \tag{1}$$

$$\text{APinc}(u, v) = \frac{\sum_{r=1}^{|1(u)|} P(r) \cdot rel'(v, r, u))}{|1(u)|} \tag{2}$$

$$\text{balAPinc}(u, v) = \sqrt{\text{APinc}(u, v) \cdot \text{LIN}(u, v)}$$

The *ClarkeDE* measure (Clarke, 2009) computes degree of entailment as the degree to which the narrower term u has lower values than v across all dimensions (eq. 3). Lenci and Benotto (2012) introduce the *invCL* measure, which uses *ClarkeDE* to measure both distributional inclusion of u in v and distributional *non-inclusion* of v in u (eq. 4). While all other measures interpret the Distributional Inclusion Hypothesis as the degree to which a $\subseteq$ relation holds, Lenci and Benotto test the degree to which proper inclusion $\subsetneq$ holds. They consider not only the degree to which the contexts of the narrower terms are included in the contexts of the wider term, but also determine the degree to which the wider term has contexts that the narrower term does not have.

$$\text{CL}(u, v) = \frac{\sum_{i=1}^{n} \min(u_i, v_i)}{\sum_{i=1}^{n} u_i} \tag{3}$$

$$\text{invCL}(u, v) = \sqrt{\text{CL}(u, v) \cdot (1 - \text{CL}(v, u))} \tag{4}$$

Like Lenci and Benotto, we focus on the stricter hypernymy relation, rather than lexical entailment. We believe that the different relations that make up lexical entailment have different distributional indications and that, for that reason, it will be easier to detect the relations separately than together.

Baroni et al. (2012) proposes a supervised approach to hypernymy detection that represents two words as the concatenation of their vectors. They also mention in passing another supervised approach that represents two words as the component-wise difference of their vectors. These are broadly the two approaches that we test, though we introduce significant modifications.

3 Data

3.1 Distributional Vector Spaces

We use three standard types of distributional spaces.

U+W2: This space is based on a concatenation of the Gigaword, BNC, English Wackypedia and ukWaC corpora (Baroni et al., 2009). The corpora are POS-tagged and lemmatized. We keep only content words (nouns, proper nouns, adjectives and verbs) with a corpus frequency of 500 or larger. The resulting U+ corpus has roughly 133K word types and 2.8B word tokens. We created a vector space by counting co-occurrences of these word types within a window of two words on the left and the right, using the top 20k most frequent content words as dimensions. The space was transformed using Positive Pointwise Mutual Information (PPMI).

U+Sent: The U+Sent space is constructed the same way as U+W2, but uses full sentence contexts instead of 2-word windows.

TypeDM: This space is extracted from the TypeDM tensors (Baroni and Lenci, 2011). TypeDM contains a list of weighted tuples, $\langle\langle w_1, l, w_2\rangle, \sigma\rangle$, where w_1 and w_2 are content words, l is a corpus-derived syntagmatic relationship between the words, and σ is a weight estimating saliency of the relationship. We construct vectors for every unique w_1 using the set of $\langle l, w_2\rangle$ pairs as dimensions and corresponding σ values as dimension weights. We select TypeDM for its excellent performance in previous comparisons of distributional hypernymy measures (Lenci and Benotto, 2012).

Reduced Spaces: In some experiments, we use dimensionality reduced spaces. We reduce all three spaces to 300 dimensions using Singular Value Decomposition. We use a subscript to denote reduced spaces, e.g. U+W2$_{300}$. When necessary, we use the term *original dimensions* to refer to the vector dimensions from the original, non-reduced spaces (e.g. U+W2); the term *latent dimensions* refers to the dimensions in the reduced spaces (e.g. U+W2$_{300}$).

3.2 Evaluation Data Sets

BLESS: The BLESS data set (Baroni and Lenci, 2011) covers 200 *concepts*, or concrete and unambiguous terms (divided into 17 different general *concept classes*, including *vehicle* and *ground mammal*), and their relationships to other nouns, called *relata*. Example concepts include *van* and *horse*. Each concept is related to several relata through different *semantic relations*. Following Lenci and Benotto (2012), we focus on the four semantic relations where both concepts and relata are nouns, for a total 14K data points: Hypernymy, denoting a superset relationship (e.g. *animal-dog*); Co-hyponymy, denoting words that share a common hypernym (e.g. *dog-cat*); Meronymy, denoting a part-whole relationship (e.g. *tail-dog*); and Random, denoting no relationship between the words (e.g. *dog-computer*).

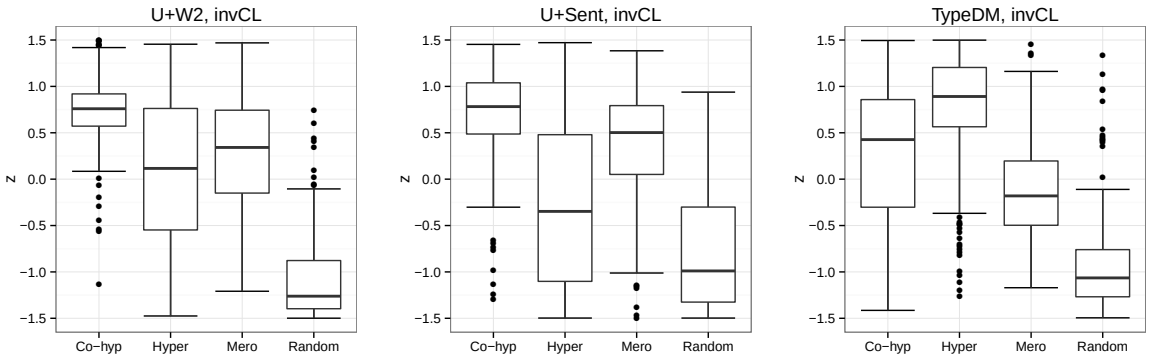


Figure 1: Distributions of relata *invCL* scores for the U+W2, U+Sent, and TypeDM spaces for each of the semantic relations, after per-concept z-normalization.

ENTAILMENT: (Baroni et al., 2012): The ENTAILMENT data set consists of 2,770 word pairs, balanced between positive (*house-building*) and negative (*leader-rider*) examples of hypernymy, with 1376 unique hyponyms and 1016 unique hypernyms. The positive examples were generated by selecting direct hypernym relationships from WordNet, the negative examples by randomly permuting the hypernyms of the positive examples, and then manually checking correctness.

4 Distributional Inclusion across Spaces

We test several unsupervised distributional approaches to hypernymy detection from the literature, focusing on the underlying vector space representation as the main parameter that we vary. We use the three spaces described in Section 3. We test four hypernymy detection approaches, all of them similarity measures based on the Distributional Inclusion Hypothesis: *WeedsPrec*, *balAPinc*, *ClarkeDE*, and *invCL*. Our baseline is the standard *cosine* measure. We evaluate on the BLESS dataset.

To evaluate on BLESS, we follow the evaluation scheme laid out in Baroni and Lenci (2011). Given a space and similarity measure, we compute similarity for each concept and relatum. For each concept, we select its nearest neighbors (according to the given similarity measure) in each of the four relations (CO-HYP, HYPER, MERO, RANDOM), and transform the corresponding four similarities to z-scores. Across all concepts, this yields four sets of z-normalized similarity scores, one for each relation. These four sets describe the relative similarity of concepts to their nearest neighbors in different relations. Tukey's Honestly Significant Difference test is used for testing whether scores differ significantly between relations (threshold: $p < 0.05$).

Figure 1 shows the distributions of z-scores for *invCL* for the four relations, with one graph for each of the three spaces we consider. For this illustration, we focus on *invCL* because it shows the overall best performance at identifying hypernymy. The rightmost plot in Figure 1 replicates the analysis of Lenci and Benotto (2012), who used the TypeDM space. It confirms their finding that *invCL* gives significantly higher values to hypernyms than co-hyponyms – at least on this space. However, in the U+W2 and U+Sent spaces (leftmost and middle plot), *invCL* clearly loses any ability to rank hypernyms the highest; indeed, in both spaces, co-hyponymy and meronymy both have significantly higher z-scores than hypernymy. Concerning the other measures, we found that they patterned with *invCL*. On TypeDM, *ClarkeDE* and *WeedsPrec* had significantly higher nearest-neighbor values for hypernyms than co-hyponyms.[2] On U+W2 and U+Sent, all measures ranked co-hyponyms significantly higher than hypernyms. With the baseline measure, *cosine*, the similarity ratings for the CO-HYP relation are always the highest, no matter the space, followed by HYPER, MERO, RANDOM in this order.

Following Kotlerman et al. (2010) and Lenci and Benotto (2012), we also report the performance of the measures using Mean Average Precision (MAP). Average Precision (AP) is a measure often used in

[2] *balAPinc* could not be evaluated on TypeDM due to computational issues.

Measure	CO-HYP	HYPER	MERO	RANDOM
U+W2				
cosine	.68	.20	.27	.27
ClarkeDE	.66	.19	.28	.28
invCL	.60	.18	.31	.28
U+Sent				
cosine	.66	.18	.28	.28
ClarkeDE	.66	.15	.29	.28
invCL	.59	.13	.34	.29
TypeDM				
cosine	.78	.19	.20	.29
ClarkeDE	.45	.35	.25	.32
invCL	.38	.36	.27	.33

Table 1: Mean Average Precision for the unsupervised measures on three spaces.

the Information Retrieval community with a maximal AP score of 1 when all relevant documents (relata with the right relationship, in our case) are ranked at the top. We compute AP on a per-concept basis and report the mean over all 200 AP values. An advantage of MAP is that, while the BLESS analysis method focuses on nearest neighbors, MAP evaluates the ranking of all relata. A disadvantage of MAP is that it does not test the degree to which a similarity measure separates different semantic relations, like Tukey does, so it may overstate the discriminative power of a particular measure. However, it provides a more intuitive accuracy-like number compared to the BLESS evaluation.

Table 1 shows the Mean Average Precision values for *cosine*, *ClarkeDE*, and *invCL* on all three spaces. We also computed *WeedsPrec* and *balAPinc* results, obtaining the same picture; we focus on *ClarkeDE* and *invCL* because *ClarkeDE* is a component of *invCL*, and *invCL* is the current best measure. The results corresponding to Lenci and Benotto's are shown in the lowest part of Table 1, where we report numbers for TypeDM. Like Lenci and Benotto, we find that unsupervised measures other than *invCL* rank co-hyponyms the highest, and obtain relatively low results for hypernyms. For *invCL* in TypeDM, Lenci and Benotto obtain 0.38 MAP for co-hyponyms and a slightly higher 0.40 for hypernyms, though they do not report significance testing results. We obtain 0.38 for co-hyponyms and 0.36 for hypernyms, and the difference is not significant.[3] Even though our results are slightly different from those in Lenci and Benotto (2012), both our results and theirs point to at most a weak preference of *invCL* for hypernyms over co-hyponyms. Moreover, in the U+W2 and U+Sent spaces we see that all three measures are very poor at identifying hypernyms, and the co-hyponymy relation stubbornly persists as most relevant to all three measures, by a large margin.

Our results thus constitute a puzzle for the Distributional Inclusion Hypothesis. It seems that there must be some merit to the hypothesis: On one particular space, namely TypeDM, the nearest neighbors in the hypernymy relation had higher similarity scores than any other relation by a significant margin. This was true for all the hypernymy detectors we studied. But even on TypeDM, the MAP evaluation showed at most a weak hypernymy signal, and when spaces other than TypeDM were used, the effect vanished altogether. So how strong an indication for hypernymy can we expect from distributional inclusion measures in general? We will return to this question below, where our answer will be: The Distributional Inclusion Hypothesis seems to hold after all, but it needs to be applied to the right kind of dimensions – and a supervised approach can help in picking the right dimensions.

As the unsupervised approaches struggle to detect hypernymy and do not seem robust to changes in standard space parameters, we think it is time to consider supervised approaches. In the next section, we explore two simple supervised approaches that show good performance and are robust to changes in the underlying space.

[3] Wilcoxon signed-rank test.

5 Supervised Hypernymy Detection

We use two simple, supervised models for predicting BLESS and ENTAILMENT relations. The first (Concat) is a model previously proposed by Baroni et al. (2012). The second (Diff) takes up an idea from a footnote in Baroni et al. (2012), but while that footnote stated that the approach in question did not work, we find that, with a few modifications, it obtains the best performance – and can be interpreted as a supervised version of the Distributional Inclusion Hypothesis. Note that while we used unreduced spaces in the previous section, we now use reduced spaces throughout (these are the spaces with the $_{300}$ subscript), in order not to have more features than data points.

5.1 Models, Features, and Method

Concat: We use a standard Support Vector Machine (SVM) classifier with a concatenation of vectors as input features. SVMs are binary classifiers which learn the maximum margin hyperplane separating the two classes. SVMs employ kernel functions to find the hyperplanes in higher dimensional spaces which are nonlinear in the original space. As feature vectors for the classifier, we follow Baroni et al. (2012) and use the concatenation of the latent dimension vectors representing words. For the ENTAILMENT dataset, we use the concatenation of the hyponym latent vector and the hypernym latent vector for each word pair as training features, and the *entails/doesn't entail* annotations as binary targets. For BLESS, we use the concatenation of the concept latent vector and the relatum latent vector as training features, and the four relationship classes as targets. We choose the four-way task rather than a "hypernymy vs. other" classification because BLESS contains many more co-hyponymy and random than hypernymy pairs, which would give a very high baseline in the two-way task. Additionally, the other relations in BLESS, in particular meronymy, may be interesting in their own right.

Since SVMs are binary classifiers, we use SciKit-Learn's default setting to train 6 pairwise-relation one-vs-one classifiers which vote on the final answer. We use a polynomial kernel with a degree of 3 and a penalty term of $C = 1.0$, and all other hyperparameters are chosen using the SciKit-Learn default values (Pedregosa et al., 2011). No hyperparameters are tuned in any experiment.

Diff: Our second classifier is a Logistic Regression (aka MaxEnt) model trained on difference vectors. Logistic Regression is a statistical model for binary classification. It learns a linear hyperplane separating the classes and estimates a probability for classes using a logistic function. We selected Logistic Regression over other possible linear classifiers for its natural ability to give likelihood estimates, which we believe will be useful in future work in an application of hypernymy classification to RTE.

As feature vectors, we use a Mikolov-inspired method of representing word pairs as the *difference vectors* between the two words.[4] Baroni et al. (2012) suggested the use of difference vectors as input to a classifier, but reported them as unsuccessful. We found difference vectors to be excellent features, with three important modifications: a linear classifier is better than a nonlinear one; vectors must be normalized to have a magnitude of 1 before taking the difference; and squared difference vectors must also be included as features. So, we represent each word pair with latent vectors (u, v) as a two part vector $\langle f; g \rangle$, where

$$f_i = \frac{u_i}{\|u\|} - \frac{v_i}{\|v\|},$$
$$g_i = {f_i}^2.$$

These differences features[5] are analogous to a *supervised* distributional inclusion measure. The difference between two words on a particular dimension captures the degree of distributional inclusion on that dimension. The primary distinction between the difference features and the unsupervised measures is that the supervised classifier learns to weight the importance of different dimensions. The f features encode directional aspects of distributional inclusion: that the hyponym contexts should be included in

[4] After recent work using subtraction to represent analogy in certain neural-network spaces (Mikolov et al., 2013).

[5] We also tried variations, such as not normalizing vectors and removing the difference squared vector, but found this setting the best. We also tried the Diff features with an SVM and other nonlinear classifiers, but they performed worse.

Data set	BLESS		ENTAILMENT	
Baseline	.46		.50	
Classifier	Concat	Diff	Concat	Diff
U+W2$_{300}$	.76	**.84**	.81	**.85**
U+Sent$_{300}$	.73	.80	.78	.82
TypeDM$_{300}$	-	.82	.65	**.85**

Table 2: Average accuracy of Concat and Diff on BLESS and ENTAILMENT using different spaces for feature generation.

those of the hypernym (the weight learned is positive), and the hypernym contexts should not be included in those of the hyponym (the weight learned is negative). So like *invCL*, this model uses a "proper subset" interpretation of the Distributional Inclusion Hypothesis, but only considers selected dimensions (i.e. those that the model assigns nonzero weights).

The difference-squared features (g), on the other hand, typically identify dimensions that are *not* indicative of hypernymy, by learning negative weights on them (more about this in Section 6). Thus, rather than helping identify hypernyms, they help separate random relations from the rest.

We use a L1 regularizer with a strength of $C = 1.0$. All other hyperparameters are chosen using the SciKit-Learn defaults. Since Diff is also a binary classifier, we use SciKit-Learn's default setting of training 4 one-vs-all classifiers for BLESS, with the most confident classifier choosing the final answer.

Method: For evaluation on BLESS, we hold out one concept and train on the remaining 199 concepts. We also exclude from the training set any pair containing a relatum which appears in the test set. This way, no word that appears in the test set has been seen in training. We report the average accuracy across all concepts. We use the most frequent relation type (random) as our baseline. For the ENTAILMENT data set, we hold out one hyponym and train on all remaining hyponyms. Again, we exclude from training any pair containing a hypernym which appears in the test set. We report average accuracy across all hyponyms. The data set is balanced, so the baseline is 0.5.

5.2 Results

Table 2 shows the performance of the two classifiers, Concat and Diff, on both the BLESS and ENTAILMENT datasets, using three underlying spaces. We use the reduced versions of the three spaces, indicated by the subscript $_{300}$. Note that the Concat classifier could not converge using features from TypeDM$_{300}$, so we omit the result. With both methods, we obtain a high accuracy on the two datasets, with results around .8 against baselines around .5. Our best result is .84 on BLESS and .85 on ENTAILMENT. Moreover, both approaches are in general robust to changes in space parameters (with TypeDM/Concat an outlier). Still, the U+W2$_{300}$ space seems to be the best for this task: Its scores are significantly[6] higher than the rest, except for TypeDM on ENTAILMENT, which achieves the same score as U+W2$_{300}$. Diff achieves significantly higher results than Concat.

When provided more information, Concat outperforms Diff. For instance, if cross-validation is done over all pairs in BLESS in the U+W2$_{300}$ space, Concat achieves .98 accuracy, while Diff obtains .90. However, in this setting the same words appear in the training and test sets (albeit in different pairs). We take this to mean that Concat is memorizing, rather than learning the hypernymy relation. This emphasizes the need for our stricter evaluation that prevents repetition between training and test sets.

Clearly, both classifiers do fairly well at predicting hypernymy relations between words, regardless of space. Naturally, one should ask what are the classifiers capturing that the unsupervised measures are missing? We propose that the supervised classifiers perform essentially the same operation as the unsupervised measures, but are learning to determine the relevance of dimensions. In particular, Diff is learning weights on vector difference features. This is equivalent to doing selective distributional inclusion. In the next section, we test this Selective Distributional Inclusion Hypothesis.

[6]Wilcoxon signed-rank test, $p < .001$.

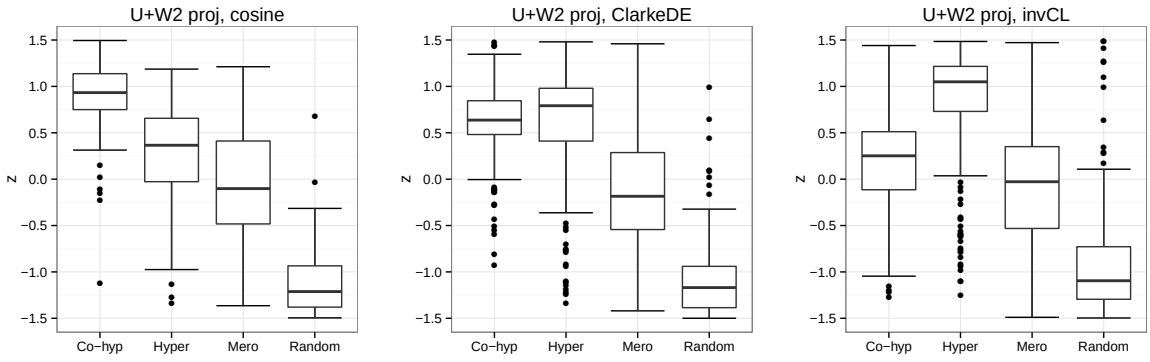


Figure 2: Distributions of relata scores across concepts using the *cosine*, *ClarkeDE*, and *invCL* measures (after per-concept z-normalization). Here we use the selected dimensions of the U+W2$_{proj}$ space.

6 Selective Distributional Inclusion

In order to test how well our supervised model is capturing the notion of selective distributional inclusion, we test each of the unsupervised measures on a smaller space, limited only to the dimensions preferred by the classifier. We emphasize that we do *not* aim to show that our supervised method outperforms unsupervised methods, but rather that the unsupervised methods benefit greatly from feature selection. Additionally, we analyze which dimensions are selected by the classifier to facilitate understanding of why these dimensions are important.

6.1 Experiment

We train the Diff classifier using the dimensionality-reduced U+W2$_{300}$ space with the same method we use in Section 5. We take the classifier's learned hyperplane separating hypernyms from other relations, and project the hyperplane back into the original U+W2 space.[7] We select the 500 dimensions in the original space that are most relevant according to the classifier weights, and test the unsupervised measures on this new space, which we denote as U+W2$_{proj}$.[8]

The 500 most relevant dimensions are selected as follows: We select the 250 most negatively weighted original dimensions using the difference features f. These are the features that have smaller values for hyponyms (e.g. *dog*) than for hypernyms (e.g. *animal*), so they characterize hypernymy. We further select the 250 most positively weighted original dimensions using the squared-differences features g. These are the ones where a large difference does not indicate hypernymy.

Figure 2 shows the boxplots for the BLESS analysis: the distributions of nearest-neighbor similarity scores for the four different semantic relations, for the measures *cosine*, *ClarkeDE*, and *invCL*. We see that *invCL* now easily discriminates hypernymy from the other relations in the backprojected space. (The difference of HYPER and CO-HYP is significant.) This is even though the space is based on U+W2, where *invCL* failed to rate hypernyms higher than co-hypernyms in Section 4. Unsurprisingly, *cosine*, which does not measure distributional inclusion, still prefers CO-HYP.

Table 3 shows the MAP scores for three of the measures in the new U+W2$_{proj}$ space. (The results for *balAPinc* and *WeedsPrec* are slightly worse than *ClarkeDE*.) All measures except for *cosine* assign higher scores to hypernyms than they did in the original space (compare to U+W2 part of Table 1). But it is only *invCL* that ranks hypernyms significantly higher than co-hyponyms.[9]

[7] Ideally we would train on the original space to inspect the relevant dimensions. However, there are more dimensions than examples, so we train on the SVD space and backproject.

[8] Note that U+W2$_{proj}$ varies slightly from concept to concept, since the hyperplane is learned on a per-concept basis. It is important that we use the linear Diff classifier for this reverse-projection procedure, as the separating hyperplane *must* be linear in order to complete the projection. In particular, the hyperplane in the Concat classifier cannot be easily backprojected, since it exists in a higher dimensional space than the projection matrix. Furthermore, it is important that we use a classifier trained using the difference features because of its analogy to the Distributional Inclusion Hypothesis.

[9] Wilcoxon signed-rank test, $p < .001$. To check that the measures are being improved by the dimension selection and not

Measure	CO-HYP	HYPER	MERO	RANDOM
U+W2$_{proj}$				
cosine	.69	.20	.24	.28
ClarkeDE	.55	.39	.24	.29
invCL	.42	**.58**	.24	.29

Table 3: Mean Average Precision for the unsupervised measures after selecting the top dimensions from a supervised model.

For this experiment, we train on all of BLESS except for one concept and then evaluate the unsupervised models on the held-out concept – that is a setting that could, in principle, be used as a hypernymy detector. If we instead train the supervised model on all of BLESS to determine an upper bound of how well dimension selection can do on this dataset, MAP for *invCL* rises to .67.

Overall, these experiments provide strong evidence for the Selective Distributional Inclusion Hypothesis: The Distributional Inclusion Hypothesis holds, but only for relevant dimensions. In addition, hypernymy detectors need to test for "proper inclusion" of distributional contexts in order to really find hypernyms.

Analysis of Selected Dimensions. We examine the 500 dimensions selected by the above procedure, in order to see what the classifier is learning. As this is for analysis only, the dimensions were selected by training on all data.

Recall that the difference-squared g features can be interpreted as dimensions that the classifier deems not indicative of hypernymy. 200 out of the 250 most relevant dimensions by g are Computer Science related terms like *software*, *configure*, or *Linux*. Since ukWaC, the largest corpus we use, is web-based, it makes sense that it has many CS-related terms, which are noise when it comes to hypernymy detection for BLESS concepts. Also, we find that while the supervised approach needs the negative information from the g features (for Diff in the U+W2$_{300}$ space, omitting g features yields a drop from .84 to .8), the unsupervised measures cannot use it. Dropping g features improves *invCL* results from .58 to .61. The g-based dimensions are explicitly those for which distributional inclusion should *not* hold, so they constitute noise to the unsupervised approaches.

The f features can be interpreted as dimensions that characterize hypernyms. An inspection reveals two clear patterns. First, the features are topically relevant for the BLESS dataset. The 17 concept classes in the dataset belong to three broader groups: animals, plants, and artifacts. An annotation of the 250 dimensions by one of the authors showed that 58 dimensions are typical of animals (*parasite*, *extinct*), 14 typical of vegetables (*flora*, *nutrient*), 80 typical of artifacts (*repair*, *mechanical*), 49 are general terms (*find*, *worthy*), and 49 have no clear interpretation (*thee*, *enigmatic*). Second, the features are general terms. For instance, for animals we find terms like *animal*, *insect*, *creature*, *fauna*, *species*, *evolutionary*, *pathogen*, *nature*, *ecology*. We also find many hypernyms, including many concept class names.

Clearly, the selected features are domain dependent; most are directly related to the concepts and concept classes of BLESS. We expect that our method should work well for other data sets, given its high accuracy and the strict training procedure. However, these features are unlikely to be global indicators of hypernymy. This emphasizes the need, in future work, to find a way to automatically determine relevance on a per-word basis.

7 Conclusion

In this paper, we have tested the Distributional Inclusion Hypothesis, the basis for distributional approaches to hypernymy. We have found that the hypothesis only works if inclusion is selectively applied to a set of relevant dimensions.

just by restricting to a smaller space, we evaluated the similarity measures on a variation of the U+W2 space which uses 500 randomly selected dimensions from the original space. The results are approximately unchanged from those on the original U+W2 space.

We have tested two simple supervised approaches to distributional hypernymy detection and have found that they show good performance, and are robust to changes in the underlying space. Our best classifier achieves .84 accuracy on BLESS and .85 on the ENTAILMENT dataset of Baroni et al. (2012). It uses features that encode dimension-wise difference between vectors. This classifier can be interpreted as selecting the dimensions necessary for the Distributional Inclusion Hypothesis to work, thus as an effective way to implement *selective* distributional inclusion.

The next natural step is to use the supervised features to guide development of an unsupervised measure for hypernymy detection: Now that we have examples, we hope to propose a method which selects relevant features automatically. We also would like to explore detection of other relationships, such as meronymy. Finally, we would like to perform an extrinsic evaluation of our hypernymy detection approach in an actual RTE system.

Acknowledgements

This research was supported by the DARPA DEFT program under AFRL grant FA8750-13-2-0026. The authors acknowledge the Texas Advanced Computing Center (TACC)[10] for providing grid resources that have contributed to these results. We thank the anonymous reviewers and the UTexas NLP group for their helpful comments and suggestions.

References

Marco Baroni and Alessandro Lenci. 2011. How we BLESSed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10, Edinburgh, UK, July. Association for Computational Linguistics.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32, Avignon, France, April. Association for Computational Linguistics.

Matthew Berland and Eugene Charniak. 1999. Finding parts in very large corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 57–64, College Park, Maryland, USA, June. Association for Computational Linguistics.

Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. 2005. *Ontology Learning from Text: Methods, Evaluation and Applications*. Frontiers in Artificial Intelligence and Applications Series. IOS Press, Amsterdam.

Timothy Chklovski and Patrick Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 33–40.

Philipp Cimiano, Aleksander Pivk, Lars Schmidt-Thieme, and Steffen Staab. 2005. Learning taxonomic relations from heterogeneous sources of evidence. *Ontology Learning from Text: Methods, evaluation and applications*.

Daoud Clarke. 2009. Context-theoretic semantics for natural language: an overview. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 112–119, Athens, Greece, March. Association for Computational Linguistics.

Maayan Geffet and Ido Dagan. 2004. Feature vector quality and distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics*, page 247. Association for Computational Linguistics.

Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 1–8. Association for Computational Linguistics.

[10] `http://www.tacc.utexas.edu`

Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2006. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1):83–135.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th Conference on Computational Linguistics*, pages 539–545, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aurélie Herbelot and Mohan Ganesalingam. 2013. Measuring semantic content in distributional vectors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 440–445, Sofia, Bulgaria, August. Association for Computational Linguistics.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16:359–389, 10.

Alessandro Lenci and Giulia Benotto. 2012. Identifying hypernyms in distributional semantic spaces. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 75–79, Montréal, Canada, 7-8 June. Association for Computational Linguistics.

Alessandro Lenci. 2008. Distributional approaches in linguistic and cognitive research. *Italian Journal of Linguistics*, 20(1):1–31.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In *Proceedings of the 18th international Joint Conference on Artificial intelligence*, pages 1492–1493.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, volume 98, pages 296–304.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June. Association for Computational Linguistics.

Gregory L. Murphy. 2002. *The Big Book of Concepts*. MIT Press, Boston, MA.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertran Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, MMatthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Enrico Santus. 2013. SLQS: An entropy measure. Master's thesis, University of Pisa.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304, Cambridge, MA. MIT Press.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 801–808, Stroudsburg, PA, USA. Association for Computational Linguistics.

Peter Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Peter D. Turney. 2006. Similarity of semantic relations. *Computational Linguistics*, 32(3):379–416.

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In Michael Collins and Mark Steedman, editors, *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 81–88.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1015–1021, Geneva, Switzerland, Aug 23–Aug 27. Association for Computational Linguistics, COLING.

Maayan Zhitomirsky-Geffet and Ido Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 107–114, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Maayan Zhitomirsky-Geffet and Ido Dagan. 2009. Bootstrapping distributional feature vector quality. *Computational linguistics*, 35(3):435–461.

Automatic Discovery of Adposition Typology

Rishiraj Saha Roy
IIT Kharagpur
Kharagpur, India – 721302.
rishiraj@cse.iitkgp.ernet.in

Rahul Katare*
IIT Kharagpur
Kharagpur, India – 721302.
rah.ykg@gmail.com

Niloy Ganguly
IIT Kharagpur
Kharagpur, India – 721302.
niloy@cse.iitkgp.ernet.in

Monojit Choudhury
Microsoft Research India
Bangalore, India – 560001.
monojitc@microsoft.com

Abstract

Natural languages (NL) can be classified as prepositional or postpositional based on the order of the noun phrase and the adposition. Categorizing a language by its adposition typology helps in addressing several challenges in linguistics and natural language processing (NLP). Understanding the adposition typologies for less-studied languages by manual analysis of large text corpora can be quite expensive, yet automatic discovery of the same has received very little attention till date. This research presents a simple unsupervised technique to automatically predict the adposition typology for a language. Most of the function words of a language are adpositions, and we show that function words can be effectively separated from content words by leveraging differences in their distributional properties in a corpus. Using this principle, we show that languages can be classified as prepositional or postpositional based on the rank correlations derived from entropies of word co-occurrence distributions. Our claims are substantiated through experiments on 23 languages from ten diverse families, 19 of which are correctly classified by our technique.

1 Introduction

Adpositions form a subcategory of *function words* that combine with noun phrases to denote their semantic or grammatical relationships with verbs, and sometimes other noun phrases. NLs can be neatly divided into a few basic typologies based on the order of the noun phrase and its adposition. If the adposition is placed *before* the noun phrase, it is called a *preposition*. *Postpositions* and *inpositions*, on the other hand, are adpositions that are placed after and inside noun phrases respectively. If prepositions are predominantly used in the language, for example in English, Bulgarian and Russian, then the language is said to be *prepositional*. Similarly, Japanese, Hindi and Turkish are some examples of *postpositional languages*, which predominantly use postpositions. These two are the most commonly found adposition typologies across the globe. Out of 1185 languages analyzed on the World Atlas of Language Structures (WALS)[1] (Dryer and Haspelmath, 2011), there are 577 postpositional, 512 prepositional and only 8 inpositional languages. There are a few (30 and 58 respectively) languages which use no or both kinds of adpositions. The order of adpositions is strongly correlated with many other word order typologies. For instance, postpositional languages usually have Object-Verb ordering, whereas prepositional languages have Verb-Object ordering (Greenberg, 1963). Daumé and Campbell (2007) present a statistical model for automatically discovering such implications from a large typological database and discuss many other typological implications involving adpositions.

Motivation. Knowledge of the typological characteristics of languages is not only of interest to linguists, but also very useful in NLP for two main reasons. First, typological information, if appropriately exploited while designing computational methods, can lead to very promising results in tasks like coreference resolution and machine translation (Haghighi and Klein, 2007; Moore and Quirk, 2007). Second, as Bender and Langendoen (2010) have pointed out, in order to claim that a computational technique

*This work was done during the author's internship at Microsoft Research India.

[1] http://wals.info/

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1037–1046, Dublin, Ireland, August 23-29 2014.

is truly language independent, one must show its usefulness for languages having diverse typological features. However, there is very little work on the automatic discovery of typological characteristics, primarily because it is assumed that such information is readily available. However, Hammarström et al. (2008) argue that documenting a language and its typological features is a time consuming process for the linguists and therefore, automatic methods for bootstrapping language description is a worthwhile effort towards language preservation. Lewis and Xia (2008) mine inter-linearized data from the Web and infer typological features for "low-density" languages, i.e. languages represented in scarce quantities on the Web. We argue that apart from documenting and understanding the typology of "low-density" languages, unsupervised discovery of adposition typology is also useful for analyzing undeciphered languages and scripts, such as the Indus valley script (Rao et al., 2009) and the Cypro-Minoan syllabary (Palaima, 1989), as well as newly emerging languages, such as the language of Web search queries (Saha Roy et al., 2012) or the Nicaraguan sign language (Meir et al., 2010). While the former cases are interesting from historical and language change perspectives, the latter cases are useful for more practical reasons (for example, improvement in query understanding leading to better Web search, and development of interactive systems for deaf children).

Approach. In this work, we show that some simple word co-occurrence statistics, that can easily be computed from any medium-sized text corpus, can be used as reliable predictors of adposition typology of a language. These statistics have been arrived at based on two fundamental assumptions: (a) adpositions constitute a large fraction of function words; and (b) the strict ordering between the adposition and the noun phrase leads to differential co-occurrence characteristics on the left and right sides of the adposition. Therefore, if the function words of a language are automatically detected and the co-occurrence statistics on the left and right of those words are appropriately analyzed, then it should be possible to tell the prepositional languages apart from the postpositional ones. Specifically, we measure counts and entropies of left, right and total (either side) co-occurrence distributions for each word. We show that left co-occurrence statistics are better indicators of function words for prepositional languages, while right co-occurrence statistics perform better for postpositional languages. Interestingly, the performance of total co-occurrence statistics lie in between the two for both types of languages. Thus, the nature of the difference in performances of left (or right) and total co-occurrences is likely to be indicative of the adposition typology of the language. We formalize this intuition to devise our test for adposition typology. We demonstrate our technique on 23 languages from ten language families, of which 14 are prepositional and 9 are postpositional. Our technique is able to consistently predict the correct adposition typology for 19 of these languages. The remaining four languages are highly *inflectional* and *agglutinating* in nature, and hence not amenable to the present technique.

Organization. The rest of this paper is organized as follows. In Sec. 2, we present our method for function word detection using word co-occurrence statistics, along with results showing the effectiveness of such an approach. In Sec. 3, we propose our test for discovering the adposition typology of a language based on correlations inferred from different co-occurrence statistics. Sec. 4 discusses experiments conducted on diverse languages and inferences drawn from the observations. Finally, Sec. 5 summarizes our contribution and indicates possible directions for future work.

2 Function Word Detection

Our method for the prediction of the adposition typology of a language relies on the facts that most adpositions are function words, and the distributional properties of function words are very different from those of content words. We exploit this difference to first formulate a method for extracting the function words of a language from a corpus. We then proceed to use the same underlying principle to automatically discover the adposition typology for languages, where we do not assume that the true function word lists are available.

By function words, we refer to all the *closed-class* lexical items in a language, e.g., pronouns, determiners, prepositions, conjunctions, interjections and other particles (as opposed to open-class items, e.g., nouns, verbs, adjectives and adverbs). For the function word detection experiments, we shall look at four languages from different families: English, Italian, Hindi and Bangla. English is a *Germanic*

Language	Corpus source	S	N	V	Function word list source	F
English	Leipzig Corpora[a]	$1M$	$19.8M$	342157	Sequence Publishing[b]	229
Italian	*-do-*	$1M$	$20M$	434680	*-do-*	257
Hindi	*-do-*	$0.3M$	$5.5M$	127428	Manually constructed by linguists and augmented by extracting pronouns, determiners, prepositions, conjunctions and interjections from POS-tagged corpora available at LDC[c]	481
Bangla	Crawl of *Anandabazar Patrika*[d]	$0.05M$	$16.2M$	411878	*-do-*	510

[a] `http://corpora.informatik.uni-leipzig.de/download.html`
[b] `http://www.sequencepublishing.com/academic.html\#function-words`
[c] `http://www.ldc.upenn.edu` (Catalog Nos. LDC2010T24 and LDC2010T16 for Hindi and Bangla respectively)
[d] `http://www.anandabazar.com/`

Table 1: Details of NL corpora for function word detection experiments.

language, Italian is a *Romanic* language, and Hindi and Bangla belong to the *Indo-Aryan* family. English and Italian are prepositional languages with *subject-verb-object* word order, while Hindi and Bangla are postpositional, relatively free word order with preference for *subject-object-verb*. Therefore, any function word characterization strategy that works across these languages is expected to work for a large variety of languages.

The details of the corpora used for these four languages are summarized in Table 1. M in the value columns denotes million. S, N, V and F denote the *numbers* of all sentences, all words, unique words (vocabulary size) and function words, respectively. We note that the Indian languages have almost twice as many function words as compared to the European ones. This is due to morphological richness and the existence of large numbers of modal and vector verbs.

Frequency is often used as an indicator for detecting function words, but the following factors affect its robustness. If the corpus size is not large, many function words will not occur a sufficient number of times. For example, even though `the` and `in` will be very frequent in most English corpora, `meanwhile` and `off` may not be so. As a result, if frequency is used as a function word detector with small datasets, we will have a problem of low recall. In our experiments, we measure corpus size, N, as the total number of words present. If our language corpus is restricted, or sampled only from specific domains, words specific to those domains will have high frequencies and will get detected as function words. For example, the word `government` will be much more frequent in political news corpora than `although`. The number of unique words in a corpus, or the vocabulary size, V, is a good indicator of its diversity. For restricted domain corpora, V grows much more slowly with N than in a general domain corpus.

We now introduce other properties of function words that may help in more robust detection. We observe the following interesting characteristics about the syntactic distributions of function and content words in NL, which can be summarized by the following two postulates.

Postulate I. Function words, in general, tend to co-occur with a larger number of distinct words than content words. What can occur to the immediate left or right of a content word is much more restricted than that in the case of function words. We hypothesize that even if a content word, e.g., *government*, might have high frequency owing to the nature of the domain, there will only be a relatively fewer number of words that can co-occur immediately after or before it. Therefore, the co-occurrence count may be a more robust indicator of function words.

Postulate II. The co-occurrence patterns of function words are less likely to show bias towards specific words than those for content words. For example, `and` will occur beside several other words like `school`, `elephant` and `pipe` with more or less equally distributed co-occurrence counts with all of these words. In contrast, the co-occurrence distribution of `school` will be skewed, with more bias towards `to, high` and `bus` than `over, through` and `coast`, with the list of words occurring beside `school` also being much smaller than that for `and`.

In order to test Postulate I, we measure the number of distinct words that occur to the immediate left,

right and either side of each unique word in the sub-sampled corpora. We shall refer to these statistics as *left*, *right* and *total co-occurrence counts* (LCC, RCC and TCC) respectively. To test Postulate II, we compute the *entropy* of the co-occurrence distributions of the words occurring to the *left*, *right* and either side (i.e., *total*) contexts of a word w:

$$\text{Entropy}(w) = -\sum_{t_i \in \text{context}(w)} p_{t_i|w} \log_2(p_{t_i|w}) \tag{1}$$

where, $\text{context}(w)$ is the set of all words co-occurring with w either in the left, the right or the total contexts, and $p(t_i|w)$ is the probability of observing word t_i in that specific context.

Context. In this paper, the left, right and total *contexts* of a word w respectively denote the immediately preceding (one) word, immediately succeeding (one) word and both the immediately preceding and the immediately succeeding words for w respectively, in sentences of the corpus. The definition of context (i.e., whether it includes the preceding or the succeeding one or two or three words) will change the absolute values of our results, but all the trends are expected to remain the same.

We shall refer to the co-occurrence entropies as *left, right* and *total Co-occurrence Entropies* (LCE, RCE and TCE respectively). Due to their pivotal role in syntactically connecting the different words or parts of a sentence to each other, we would expect LCC, RCC or TCC of function words to be higher than that of content words due to *Postulate I*; similarly, due to *Postulate II* we can expect the LCE, RCE or TCE to be higher for function words than for content words. If the LCE or LCC of a word w is high, it means that a large number of distinct words can *precede* w in the language (additionally, almost with equal probabilities for high LCE). Thus, predicting the *previous* word of w is difficult. Similarly, if RCE or RCC of w is high, it means that a large number of words can *follow* w in the language (additionally, almost with equal probabilities for high RCE). Thus, predicting the *next* word of w is difficult. A high TCE for a word implies that the word can be preceded and followed by a large number of words, making the prediction of either the next or the previous word (or both) for w difficult.

2.1 Experiments and Results

In our approach, the output is a ranked list of words sorted in descending order of the corresponding property. Here we adopt a popular metric, *Average Precision* (AP), used in Information Retrieval (IR) for the evaluation of ranked lists. More specifically, let $w_1, w_2, \ldots, w_n$ be a ranked list of words sorted according to some corpus statistic, say, frequency. Thus, if $i < j$, then frequency of w_i is greater than the frequency of w_j. *Precision at rank k*, denoted by P@k, is defined as

$$\text{P@}k = \frac{1}{k}\sum_{i=1}^{k} f(w_i) \tag{2}$$

where, $f(w_i)$ is one if w_i is a function word, and is zero otherwise. This function can be computed based on the gold standard lists of function words. Subsequently, *average precision at rank n*, denoted by AP@n, is defined as

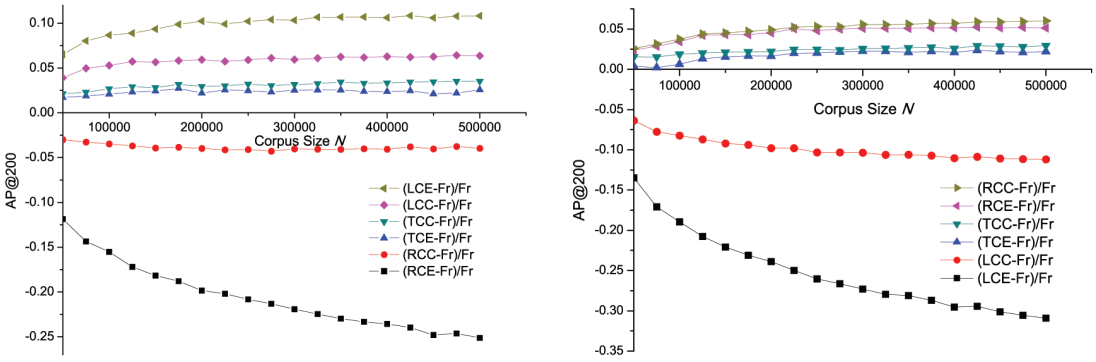
$$\text{AP@}n = \frac{1}{n}\sum_{k=1}^{n} \text{P@}k \tag{3}$$

AP@n is a better metric than P@k because P@k is insensitive to the rank at which function words occur in the list. In our experiments, we compute AP@n averaged over $\mathcal{N}$ corpus sub-samples, which is given by $\frac{1}{\mathcal{N}}\sum_{r=1}^{\mathcal{N}}(\text{AP@}n)_r$ where $(\text{AP@}n)_r$ is the AP@n for the r^{th} sub-sample. We note that there are other metrics popularly used in IR, e.g. the Normalized Discounted Cumulative Gain (nDCG). However, these are more sensitive to the correctness of the top few items in the list and hence, are not suitable for us. Knowing that the number of function words in a popular NL is at least 200 (Table 1), we compute AP@200 with respect to the gold standard lists of function words for all our experiments.

Language	Typology	Fr	LCC	LCE	TCC	TCE	RCC	RCE
English	Prepositional	0.663	**0.702**†	**0.729**†	**0.684**†	**0.679**†	0.637	0.527
Italian	Prepositional	0.611	**0.639**†	**0.645**†	**0.636**†	**0.620**	0.606	0.601
Hindi	Postpositional	0.682	0.614	0.510	**0.698**†	**0.694**†	**0.716**†	**0.713**†
Bangla	Postpositional	0.648	0.684†	0.691†	**0.730**†	**0.763**†	**0.741**†	**0.757**†

The four highest values in a row are marked in **boldface**. Statistically significant improvement over frequency is marked by †. The paired t-test was performed and the null hypothesis was rejected if p-value < 0.05.

Table 2: AP@200 for all indicators, averaged over 200 (N, V) pairs for each language.



(a) (b)

Figure 1: (Colour online) Performance of co-occurrence statistics for (a) English, and (b) Hindi, with respect to frequency for AP@200 with variation in N.

We now sort the list of all words in descending order of each of the seven indicators. We then compute AP@200 for these seven lists. To bring out the performance difference of each of the six co-occurrence features with respect to frequency, we plot (in Figs. 1 and 2) the following measure against N:

$$\text{Value plotted} = \frac{\text{Metric for indicator} - \text{Metric for Fr}}{\text{Metric for Fr}} \tag{4}$$

The x-axis can now be thought of as representing the performance of frequency. In Fig. 1, for a particular N, the data points were averaged over all (N, V) pairs (we had 20 (N, V) pairs for each N). For Fig. 2, V was binned into five zones, and for each zone, the AP was averaged over all corresponding (N, V) pairs. The observations (both N and V variation) for French and Italian were similar to that of English, while those for Hindi and Bangla were similar to each other. Table 2 reports AP values for all statistics for the four languages. From Table 2, we see that for all the languages, AP for some of the co-occurrence statistics are higher than AP obtained using frequency.

Regular improvements over frequency. From the plots and Table 2, it is evident that some of the co-occurrence statistics consistently beat frequency as indicators. In fact, as evident from Figs. 1 and 2, use of co-occurrence statistics results in systematic improvement over frequency with variations in N and V, and hence, are very robust indicators. Among the co-occurrence statistics, both entropies and counts are observed to have comparable performance.

3 Detection of Adposition Typology

From the results presented above, we observe that the best function word indicator depends upon language typology. Interestingly, while LCE and LCC are the best indicators of function words for the two prepositional languages of English and Italian, RCE and RCC perform better for Hindi and Bangla, the postpositional languages. This observation can be explained as follows. For a prepositional language, the function words, which are often the adpositions, precede the content word it is linked to. Therefore, the words following an adposition (or a function word) mark the beginnings of syntactic units such as

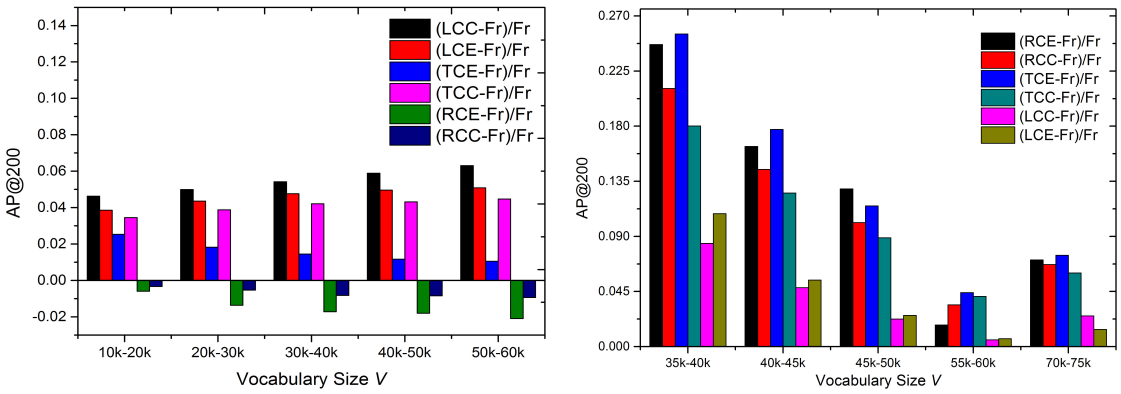


Figure 2: (Colour online) Performance of co-occurrence statistics for (a) Italian, and (b) Bangla, with respect to frequency for AP@200 with variation in V.

noun phrases and are typically restricted to certain syntactic categories. However, the words that precede the adpositions have no or much weaker syntactic restrictions. Hence, the LCE and LCC are higher and consequently better and more robust indicators of function words for prepositional languages. For very similar reasons, the RCE and RCC are better indicators of function words for postpositional languages. Importantly, we observe that TCE and TCC seem to be reasonably good predictors of function words irrespective of the typology, with performances lying in between the poorest indicators (RCE and RCC for prepositional languages and LCE and LCC for postpositional languages) and the best indicators (LCE and LCC for prepositional languages and RCE and RCC for postpositional languages) for all the four languages. This makes them safe indicators to rely on when not much is known about the language syntax. In fact, the philosophy of this research is to be of assistance in these less-known cases. Thus, co-occurrence statistics have potential in predicting the adposition typology of a new language, which we leverage in this research.

We now describe our intuition and method behind our tests for automatically detecting the adposition typology of a language. In this context, we *do not know* the actual function words or adpositions of the language under consideration. Let us take the three lists of the top 200 words from a language corpus, sorted according to the statistics TCE, LCE and RCE. For a prepositional language, we can expect to see the highest number of function words towards the top of the list when sorted according to LCE, followed by the number of function words towards the top of the TCE list. The RCE list would be expected to be the poorest in this regard. Thus, we expect a higher overlap between the top 200 word lists for TCE and LCE, than for TCE and RCE. The reverse is expected to be true for postpositional languages. Similar arguments can be presented for LCC, RCC and TCC as well. We quantify this correlation between the lists using two different statistics – the *Pearson's correlation coefficient* (r) and *Spearman's Rank Correlation Coefficient* (ρ).

For computing Pearson's coefficients, we use the actual values of the distributional statistics, while for Spearman's rank coeffcients, we use the ranks of the words. Let $r(\mathrm{TL})$ and $\rho(\mathrm{TL})$ respectively denote the Pearson's and Spearman's Rank correlation coefficients of the lists sorted by TCE and LCE (or TCC and LCC), and similarly, let $r(\mathrm{TR})$ and $\rho(\mathrm{TR})$ denote the respective coefficients for the lists sorted by TCE and RCE (or TCC and RCC).

Postulate. For a prepositional language, the top-200 words by LCE will have a higher correlation with the top-200 words by TCE than the corresponding correlation of RCE with TCE. For a postpositional language, the top-200 words by RCE will have a higher correlation with the top-200 words by TCE. Formally, for *prepositional languages*, $r(\mathrm{TL}) > r(\mathrm{TR})$, and $\rho(\mathrm{TL}) > \rho(\mathrm{TR})$, while for *postpositional languages* $r(\mathrm{TL}) < r(\mathrm{TR})$ and $\rho(\mathrm{TL}) < \rho(\mathrm{TR})$.

Language	Family	ρ(TL)	ρ(TR)	ρ(Diff.)	Predicted	True
Bulgarian	Slavic (Indo-European)	0.726	0.518	0.208	Pre-	Pre- (Scatton, 1984)
Danish	Germanic (Indo-European)	0.621	0.495	0.126	Pre-	Pre- (Allan et al., 1995)
Dutch	Germanic (Indo-European)	0.662	0.204	0.458	Pre-	Pre- (Shetter, 1958)
English	Germanic (Indo-European)	0.461	0.436	0.025	Pre-	Pre- (Selkirk, 1996)
German	Germanic (Indo-European)	0.563	0.517	0.046	Pre-	Pre- (Lederer, 1969)
Italian	Romance (Indo-European)	0.730	0.456	0.274	Pre-	Pre- (Sauer, 1891)
Macedonian	Slavic (Indo-European)	0.692	0.488	0.205	Pre-	Pre- (Friedman, 1993)
Norwegian	Germanic (Indo-European)	0.619	0.600	0.019	Pre-	Pre- (Olson, 1901)
Polish	Slavic (Indo-European)	0.798	0.554	0.243	Pre-	Pre- (Bielec, 1998)
Russian	Slavic (Indo-European)	0.743	0.652	0.091	Pre-	Pre- (Borras and Christian, 1959)
Slovenian	Slavic (Indo-European)	0.701	0.668	0.032	Pre-	Pre- (Priestly, 1993)
Swedish	Germanic (Indo-European)	0.663	0.525	0.138	Pre-	Pre- (Holmes and Hinchliffe, 1994)
Ukrainian	Slavic (Indo-European)	0.785	0.714	0.070	Pre-	Pre- (Stechishin, 1958)
Gujarati	Indic (Indo-European)	0.540	0.581	−0.041	Post-	Post- (Cardona, 1965)
Hindi	Indic (Indo-European)	0.529	0.731	−0.202	Post-	Post- (McGregor, 1977)
Japanese	Japanese (Japanese)	0.429	0.626	−0.197	Post-	Post- (Hinds, 1986)
Nepali	Indic (Indo-European)	0.495	0.719	−0.224	Post-	Post- (Bandhu, 1973)
Tamil	Southern Dravidian (Dravidian)	0.748	0.805	−0.057	Post-	Post- (Asher, 1982)
Turkish	Turkic (Altaic)	0.531	0.769	−0.238	Post-	Post- (Underhill, 1976)
Estonian	Finnic(Uralic)	0.790	0.733	0.057	Pre-	Post- (Tauli, 1983)
Finnish	Finnic (Uralic)	0.671	0.656	0.015	Pre-	Post- (Sulkala and Karjalainen, 1992)
Hungarian	Ugric (Uralic)	0.457	0.329	0.128	Pre-	Post- (Kenesei et al., 1998)
Lithuanian	Baltic (Indo-European)	0.715	0.724	−0.009	Post-	Pre- (Dambriunas et al., 1966)

Misclassified languages are marked in gray.

Table 3: Detecting adposition typology using Spearman's rank correlation coefficients on entropy lists.

4 Experimental Results and Observations

In this section, we first present our datasets, followed by detailed experiments on adposition typology detection and inferences drawn from the observations.

4.1 Datasets

For all our typology detection experiments, we use datasets from the publicly available Leipzig Corpora[2]. We selected 23 languages from various families that are typologically diverse. A $(300,000)$-sentence corpora was used for all the languages so as to ensure similar-sized corpora for all the languages (many languages do not have a larger corpus). All languages examined have been listed in Table 3, along with their families and true adposition typologies (accompanied by appropriate references).

4.2 Experiments and Results

We extracted the top-200 words by TCE, LCE and RCE, and TCC, LCC and RCC from the $300k$-sentence corpora. We then computed $r(\mathrm{TL})$, $\rho(\mathrm{TL})$, $r(\mathrm{TR})$ and $\rho(\mathrm{TR})$, for both entropies and counts. As per our postulate, if $\rho(\mathrm{TL}) - \rho(\mathrm{TR})$ (= $\rho(\mathrm{Difference})$) is positive, the language is prepositional; if it is negative, the language is postpositional. The same can be expected for $r(\mathrm{Difference})$.

The performance of ρ as a predictor was found to be better than r. Results when the entropy lists are used are presented in Table 3. For only 4 out of 23 languages, the typology predictions are incorrect. We observe that three of these misclassified languages are from the Uralic family that are *synthetic* in nature characterized by extensive regular *agglutination* of modifiers to verbs, nouns, adjectives and numerals.

[2] http://corpora.informatik.uni-leipzig.de/download.html

Corpus Size	Entropy lists (r)	Entropy lists (ρ)	Count lists (r)	Count lists (ρ)
$10k$	$17/23$	$\mathbf{21/23}$	$17/23$	$13/23$
$100k$	$17/23$	$\mathbf{19/23}$	$18/23$	$13/23$
$300k$	$16/23$	$\mathbf{19/23}$	$16/23$	$13/23$

The highest value in a row is marked in **boldface**.

Table 4: Correct predictions by strategy with varying factors.

The average number of characters in words of these languages were found to be in the relatively higher range of nine to eleven. Thus, function words, especially the adpositions, seldom occur as free words in these languages and hence our method cannot capture the distributional characteristics of the adpositions. It is worthwhile to note that the method can predict the correct typology for other languages that employ agglutination to a lesser degree (Bulgarian, Dutch, German, Tamil and Turkish). Lithuanian, though not synthetic, is a highly inflectional language and therefore, instead of adpositions it makes extensive use of case-markers. With ρ(Difference) very close to zero, our prediction for Lithuanian is inconclusive.

A note on synthetic languages: For synthetic languages, the difference between the two rank correlation coefficients are close to zero, which provides us with a direct way to identify them. One could also employ unsupervised morphological analysis to automatically identify and segment affixes, which will provide deeper insight into the morpho-syntactic properties of the language. Nevertheless, affixes (like infixes in Arabic or case-marking suffixes in Bangla) are technically not considered as adpositions, and therefore, they do not really determine the adposition typology. Languages are divided into four classes according to their adposition usage: prepositional, postpositional, ambi-positional (use both types) and adposition-less (use none). Thus, as far as adposition typology is concerned, it suffices to identify whether a language is primarily adposition-less, which our technique is potentially capable of doing (we demonstrate it for four languages, but we believe more experimentation is needed to establish this claim). Note that a language may use case-marking affixes along with adpositions. In such cases our method is able to correctly determine the typology, as demonstrated for Bangla.

4.3 Experimental variations

We repeated the above experiments with lists of TCC, LCC and RCC instead of the co-occurrence entropies. The performance was found to be poorer than the entropy lists, with nine classification errors instead of the earlier four. Performance of these lists by co-occurrence counts was found to be poorer in other cases as well (Table 4). We systematically experimented with r instead of ρ. To test the performance of our method with even smaller corpora, we sub-sampled 3 and 30 corpora containing $100k$ and $10k$ sentences respectively from the $300k$ corpus. We computed the correlation between the original top 200 words obtained using TCE (or TCC) from the $300k$ corpus and the corresponding LCE and RCE (or LCC and RCC) lists obtained from the smaller corpora. For a given language, the mean of ρ(Difference) and r(Difference) were used to predict the typology (observed standard deviations were very low, of the order of 10^{-3}). The results of these experiments are summarized in Table 4. Out of 23 languages, 21 and 19 were correctly classified by ρ for corpora of $10k$ and $100k$ sentences. The corresponding number for r are 18 for both $10k$ and $100k$, and 17 for $300k$ corpora. Thus, the sensitivity of the method improves with slightly smaller corpora, provided that the TCE list, which is being used as a proxy for the gold standard function word list, is computed from a slightly larger corpus. Finally, we note that using Spearman's rank correlation coefficient with lists constructed by co-occurrence entropy consistently produces the best results.

5 Conclusions and Future Work

Knowing the adposition typology of a natural language can be useful in several NLP tasks, and can be especially useful in understanding new or undeciphered languages. In this research, we have taken one

of the first steps towards automatic discovery of adposition typology. First, we have shown, through experiments on two prepositional and two postpositional languages, that function words can be effectively extracted from medium-sized corpora using word co-occurrence statistics, and such measures usually outperform simple frequency when used for the same task. Next, difference in behavior of various co-occurrence statistics for prepositional and postpositional languages has been exploited to devise a simple strategy for predicting the adposition typology of a language. Simple differences of rank correlation coefficients among total, left and right word co-occurrence entropies have been shown to be potent signals towards automatic discovery of adposition and noun phrase typology in a language. Results show sufficient promise through an extensive evaluation over 23 languages.

We ventured into this study while solving a very practical and important problem: query understanding through analysis of the structure of Web search queries. While queries seem to have an emergent syntax, it is unclear whether they have function words, and if so what role they play in determining the query grammar. To this end, we conducted the current study. Thus, we envisage that this technique will be applicable for any such emergent linguistic system, such as pidgins, creoles, and computer mediated communications (CMCs) like SMS and chats, where there is a large amount of text data available but the grammar is emerging or yet to be analyzed. Other examples are that of undeciphered languages, e.g., Indus valley language or script. In fact, our method can be applied to any system of symbols, be it linguistic or non-linguistic, such as musical note sequences.

As future work, it is important to improve our prediction accuracy further, while including more languages in the experimental setup. Combining clues from other sources to resolve uncertain cases and devising better ways of choosing corpus size and significance thresholds are some of the avenues in which effort may be channelized. Extending our approach to a morpheme-level analysis would also be beneficial in dealing with highly agglutinative and inflectional languages.

Acknowledgements

The first author was supported by Microsoft Corporation and Microsoft Research India under the Microsoft Research India PhD Fellowship Award. We would like to thank Amritayan Nayak, Walmart eCommerce (who was then a student of IIT Kharagpur working as an intern at Microsoft Research India) for contributing to some of the early experiments related to this study.

References

Robin Allan, Philip Holmes, and Tom Lundskær-Nielsen. 1995. *Danish: A Comprehensive Grammar*. Routledge, London.

R. E. Asher. 1982. *Tamil*, volume 7 of *Lingua Descriptive Studies*. North-Holland, Amsterdam.

Churamani Bandhu. 1973. Clause patterns in nepali. In Austin Hale, editor, *Clause, sentence, and discourse patterns in selected languages of Nepal 2*, volume 40.2 of *Summer Institute of Linguistics Publications in Linguistics and Related Fields*, pages 1–79. Summer Institute of Linguistics of the University of Oklahoma, Norman.

Emily M. Bender and D Terence Langendoen. 2010. Computational linguistics in support of linguistic theory. *Linguistic Issues in Language Technology*, 3(1).

Dana Bielec. 1998. *Polish: An Essential Grammar*. Routledge, London.

F. M. Borras and R. F. Christian. 1959. *Russian Syntax: Aspects of Modern Russian Syntax and Vocabulary*. Clarendon Press, Oxford.

George Cardona. 1965. *A Gujarati Reference Grammar*. The University of Pennsylvania Press, Philadelphia.

Leonardas Dambriunas, Antanas Klimas, and William R. Schmalstieg. 1966. *Introduction to Modern Lithuanian*. Franciscan Fathers Press, Brooklyn.

Hal Daumé and Lyle Campbell. 2007. A bayesian model for discovering typological implications. In *Annual Meeting of the Association for Computational Linguistics*, pages 65–72.

Matthew S. Dryer and Martin Haspelmath, editors. 2011. *The World Atlas of Language Structures Online*. Max Planck Digital Library, Munich, 2011 edition.

Victor A. Friedman. 1993. Macedonian. In Bernard Comrie and Greville G. Corbett, editors, *The Slavonic Languages*, pages 249–305. Routledge, London / New York.

Joseph H. Greenberg. 1963. Some universals of grammar with particular reference to the order of meaningful elements. *Universals of language*, 2:73–113.

Aria Haghighi and Dan Klein. 2007. Unsupervised coreference resolution in a nonparametric bayesian model. In *Annual meeting-Association for Computational Linguistics*, pages 848–855.

Harald Hammarström, Christina Thornell, Malin Petzell, and Torbjörn Westerlund. 2008. Bootstrapping language description: The case of mpiemo (bantu a, central african republic). In *Proceedings of the Sixth international conference on Language Resources and Evaluation*, LREC '08.

John Hinds. 1986. *Japanese*, volume 4 of *Croom Helm Descriptive Grammars*. Croom Helm, Routledge, London.

Philip Holmes and Ian Hinchliffe. 1994. *Swedish: A Comprehensive Grammar*. Routledge, London.

Istvn Kenesei, Robert M. Vago, and Anna Fenyvesi. 1998. *Hungarian*. Descriptive Grammars. Routledge, London / New York.

Herbert Lederer. 1969. *Reference Grammar of the German Language*. Charles Scribner's Sons, New York. Based on Grammatik der Deutschen Sprache, by Doras Schulz and Heinz Griesbach.

W. Lewis and F. Xia. 2008. Automatically identifying computationally relevant typological features. In *Proc. of the Third International Joint Conference on Natural Language Processing (IJCNLP-2008)*.

R. S. McGregor. 1977. *Outline of Hindi Grammar*. Oxford University Press, Delhi. 2nd edition.

Irit Meir, Wendy Sandler, Carol Padden, and Mark Aronoff. 2010. Emerging sign languages. *Oxford handbook of deaf studies, language, and education*, 2:267–280.

R. C. Moore and C. Quirk. 2007. An iteratively-trained segmentation-free phrase translation model for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 112–119. Association for Computational Linguistics.

Julius E. Olson. 1901. *Norwegian Grammar and Reader*. Scott, Foresman and Co, Chicago.

Thomas G. Palaima. 1989. Cypro-minoan scripts: Problems of historical context in problems in decipherment. *Bibliothèque des Cahiers de l'Institut de Linguistique de Louvain*, 49:121–187.

T. M. S. Priestly. 1993. Slovene. In Bernard Comrie and Greville G. Corbett, editors, *The Slavonic Languages*, pages 388–451. Routledge, London.

R.P.N. Rao, N. Yadav, M.N. Vahia, H. Joglekar, R. Adhikari, and I. Mahadevan. 2009. Entropic evidence for linguistic structure in the Indus script. *Science*, 324(5931):1165–1165.

Rishiraj Saha Roy, Monojit Choudhury, and Kalika Bali. 2012. Are web search queries an evolving protolanguage? In *Proceedings of the 9th International Conference on the Evolution of Language*, Evolang 9, pages 304–311, Singapore. World Scientific Publishing Co.

Charles Marquard Sauer. 1891. *Italian Conversational Grammar*. Julius Gross, Heidelberg.

Ernest A. Scatton. 1984. *A Reference Grammar of Modern Bulgarian*. Slavica Publishers, Columbus, Ohio.

Elizabeth Selkirk. 1996. The Prosodic Structure of Function Words. In James L. Morgan and Katherine Demuth, editors, *Signal to Syntax: Bootstrapping from Speech to Grammar in Early Acquisition*. Routledge.

William Z. Shetter. 1958. *Introduction to Dutch*. Martinus Nijhoff, The Hague.

J. W. Stechishin. 1958. *Ukrainian Grammar*. Trident Press, Winnipeg.

Helena Sulkala and Merja Karjalainen. 1992. *Finnish*. Descriptive Grammar Series. Routledge, London.

Valter Tauli. 1983. *Standard Estonian Grammar. Volume 2: Syntax*, volume 14 of *Studia Uralica et Altaica Upsaliensia*. Almqvist and Wiksell, Uppsala.

Robert Underhill. 1976. *Turkish Grammar*. Massachusetts Institute of Technology (MIT) Press, Cambridge.

What good are 'Nominalkomposita' for 'noun compounds': Multilingual Extraction and Structure Analysis of Nominal Compositions using Linguistic Restrictors

Patrick Ziering **Lonneke van der Plas**
Institute for Natural Language Processing
University of Stuttgart, Germany
{Patrick.Ziering, Lonneke.vanderPlas}@ims.uni-stuttgart.de

Abstract

Finding a definition of compoundhood that is cross-lingually valid is a non-trivial task as shown by linguistic literature. We present an iterative method for defining and extracting English noun compounds in a multilingual setting. We show how linguistic criteria can be used to extract compounds automatically and vice versa how the results of this extraction can shed new lights on linguistic theories about compounding. The extracted compound nouns and their multilingual contexts are a rich source that serves several purposes. In an additional case study we show how the database serves to predict the internal structure of tripartite noun compounds using spelling variations across languages, which leads to a precision of over 91%.

1 Introduction

Compounding is a phenomenon that is studied extensively in linguistic literature. Also in computational linguistics, compounds are enjoying more and more attention (Ó Séaghdha, 2008; Hendrickx et al., 2013). Compounding is a very productive word formation. Already 2-year-olds are able to form new words by using compounds consisting of two morphemes (Clark, 1981). As a consequence, compounds are a very common word type but many occur with a very low token count. In an analysis of the German APA corpus, Baroni et al. (2002) found that almost half (47%) of the word types were compounds. At the same time, the compounds accounted for a small portion of the overall token count (7%), which suggests that many of them are rare (83% of the compounds had a corpus frequency of 5 or lower). For English, more than half of the two-noun compounds (e.g., *car park*) in the BNC occur exactly once (Kim and Baldwin, 2006). The high productivity of compounds makes compositional approaches to automatic processing indispensable: listing all possible compounds in a dictionary would be as infeasible as listing all possible adjective-noun combinations. Even for compound nouns that occur 10 times or more in the BNC, static English dictionaries provide only 27% coverage (Tanaka and Baldwin, 2003).

Being abundant as a phenomenon but scarce in terms of individual examples (the combination of high type frequency and low token frequency) makes the analysis of these compound nouns particularly problematic for statistical techniques that need high token frequencies to make accurate predictions. Data sparsity is expected to lead to low performance. However, the correct analysis of compound nouns is important for a number of NLP tasks, for example in machine translation (Bouillon et al., 1992; Rackow et al., 1992; Johnston and Busa, 1999; Navigli et al., 2003). The accurate translation of compounds is non-trivial, because we find a large amount of variation in the way languages deal with compounding. Some languages such as German use closed compounding (i.e., they create one-word compounds, e.g., *Todesstrafe* (*death penalty*)) whereas others do not. In Romance languages, such as French, compounds are not as productive, instead postmodifying prepositional phrases (e.g., *peine de mort*) and adjectives (*peine capitale*) are used to construct complex nominals.

Another challenge in compound translation is due to the fact that the amount of underspecification in compound surface structure varies between languages. For example, whereas English leaves the compound relation (i.e., the semantic relation between two components, e.g., N_2 *made of* N_1 as in *iron door*)

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1047–1058, Dublin, Ireland, August 23-29 2014.

covert, in French we find prepositions that correlate with the relation type (Girju, 2007; Celli and Nissim, 2009). *Chocolate cake*, cake **made of** chocolate, is translated with *gateau **au** chocolat*, whereas *wedding cake*, cake **made for** a wedding, is *gateau **de** marriage*.

The first aim of this study is to extract a large database of compounds and their translations in context from a parallel corpus. This database will serve multiple purposes. For example, it will be used to study compounding across different languages, and we will exploit the cross-lingual variation for compound processing. In the second part of this paper, we will show a case study of how the extracted database can be used for analysing the structure of noun phrases, more specifically, we exploit spelling variations across languages for bracketing three-noun compounds (3NCs) such as *air traffic control*, which could be indicated as LEFT bracketing using the German phrase *Kontrolle des Luftverkehrs* (*control of air traffic*).

Compounding is an important subject of study in theoretical linguistics, because it constitutes a continuum from fully compositional to idiosyncratic word formation and is found at the boundary between words and phrases. However, there is virtually no reliable and universally accepted criterion for distinguishing compounds from phrases or other types of word formations, as stated by Lieber and Stekauer (2009). They discuss reasons for the complexity that arises when defining noun compounds, that we will review in the next section. They do, however, also describe a number of linguistic tests, each with their own advantages and drawbacks.

This brings us to the second aim of this paper. We propose an iterative method that, in absence of a clear definition, validates several linguistic tests on corpus data and continuously refines the definition. We show how we use linguistic tests to extract compounds automatically and vice versa how the results of this extraction can shed new lights on linguistic theory about compoundhood. The multilingual nature of our data (we work on parallel corpora) has the additional advantage that a cross-lingual definition can be sought by studying compounds in context and their translation across several languages.

In Section 2, we discuss the problem of defining noun compounds (NCs) as described by Lieber and Stekauer (2009) and present an iterative method for defining and extracting English NCs starting with an initial definition based on some linguistic tests. In Section 3, we present our method for extracting English NCs and their translations to several languages from a parallel corpus using a set of extraction rules. An experimental setup and results are presented in Section 4. In Section 5 we show in a case study of bracketing three-noun compounds how our database serves for exploiting multilingual spelling variations. Section 6 describes related work and finally Section 7 concludes.

2 Iterative method for the definition and extraction of noun compounds

In this section, we outline the controversy of defining compoundhood as described in linguistic literature. We present several linguistic tests for distinguishing compounds and show how we implement some linguistic criteria that can be used for identification and extraction of noun compounds and how these constitute the initial definition.

2.1 Definition of compounds and linguistic criteria

When we seek to find a working definition of noun compounds (NCs), we have to keep in mind that not only the definition but also the existence of such an NC is controversial. Lieber and Stekauer (2009) present a discussion about this controversy sketched below. While Bauer (2003) defines a compound as a "formation of a new lexeme by adjoining two or more lexemes", Marchand (1967) argues that there is no compounding word formation at all. Instead, he uses the word formation EXPANSION, which combines prefixed words like *reheat* with such as *steamboat* using the criterion of a free head. Lieber and Stekauer (2009) highlight two reasons for the complexity that arises when defining noun compounds. Firstly, in some languages, constituents are not free but stems or roots. For example, the Slovak term *rýchlovlak* (express train) starts with the stem of the adjective *rýchly* (as in the phrase *rýchly vlak* (fast train)). The lack of inflection in English makes compositional and phrasal structures (i.e., *fast train* as phrase or as compound (*express train*)) collapse. Secondly, sometimes phrases and derivations cannot be distinguished from compounds. While *blackboard* (in opposition to a *black board*) can be classified as compound without dissent, a *tomato bowl* that just happens to hold tomatoes might not be regarded as a

single lexeme (conforming Bauer's (2003) definition).

So, the only way for getting a suitable definition is to find solid criterions. Although Lieber and Stekauer (2009) come to the conclusion that there is almost no reliable and universally accepted criterion, they mention several plausible tests. Compounds can be identified by prosody. While in the phrase *black bird*, the head (*bird*) is stressed, in the compound *blackbird* the stress is on the first syllable (*black*). A syntactic test mentioned by Lieber and Stekauer (2009) is inseparability, i.e., there must not be any element intervening a compound's components. While *black bird* can be understood as compound, *black ugly bird* is a phrase. Another promising syntactic criterion is the inability to modify the first element (i.e., the modifier) of a compound. In a phrase like *social person*, the first element (*social*) can be modified (i.e., *very social person*). This is not possible for compounds (e.g., very *social policy*). A last syntactic criterion, the inability to replace the second noun of a nominal compound with a proform such as *one* (e.g., *black bird* vs. *black one*), would need human support. A morphological criterion states that in compounds only the head is inflected. Although this assumption does not always hold (as shown in examples like *overseas investor* or *girls club*), this seems to be a promising criterion when investigating inflectional behaviour in aligned languages that show strong morphology, e.g., French. Conversely, determining compoundhood on the basis of spelling is discarded by Lieber and Stekauer (2009). English orthography is highly inconsistent: some compounds usually occur as a closed compound (e.g., *football*), some occur hyphenated and some occur as an open compound (e.g., *waiting room* or *rule of law*). For some compounds, several spellings are possible (e.g., *flowerpot*, *flower-pot*, *flower pot* or *pot of flowers*).

In our study, we focus on written language as given in a parallel corpus. Since we do not have any speech data, we cannot use any phonological features such as stress for the extraction of noun compounds. For the inability to replace the second noun of a nominal compound with a proform, we cannot assess if the meaning of a sentence would have changed (e.g., *We see blackbirds* vs. *We see black ones*).

In this paper, we focus on criteria that are most suitable with the current data. Although Lieber and Stekauer (2009) exclude spelling as a reliable criterion of compoundhood, we take it as starting point. The parallel corpus we use for the extraction includes several languages. Spelling variations between languages can be exploited to find compounds (e.g., *social policy* can be written as one word in German (*Sozialpolitik*)). We account for the English spelling variations by defining part-of-speech (PoS) patterns that cover most plausible spellings. These PoS patterns treat each noun or adjective as a compound's component and thus, this way of extraction inherently implements the criterion of inseparability. We exploit multilingual evidence in terms of cross-lingual differences in spelling to extract compounds. Diverse language families have different declinations of forming a closed compound. While languages like Danish and German prefer closed compounding, English and Romance languages like Spanish use open compounds. It is this spelling variation that we base our first set of extraction rules on with the aim of having a set of English NCs and their translations in up to 9 European languages. We will show that cross-lingual closed compounding is a promising feature for extracting English NCs.

The inability to modify the first element of a compound seems to be a promising test. Since there are many linguistic factors that have to be taken into account (e.g., morphological agreement in gender, number or case), we plan to include this criterion for several languages and any combination of contextual modifier and potential noun compound. We will implement this and further morphological criteria in future work.

2.2 Initial definition for compound extraction

With a focus on multilingual validity, we adapt the definition of Bauer (2003) to our multilingual setting. Inspired by Behagel's (1909) First Law ("Elements that belong close together intellectually will also be placed close together"), we associate a closed compounding language realising an English word sequence as a closed NC with an indicator for compoundhood:

Initial definition: *A noun compound is a nominal composition of several lexemes that are represented as a one-word expression in some of the languages studied.*

This definition covers both target single words (e.g., *blackbird* translates to German as *Amsel*) or target

closed compounds (e.g., *football match* translates to Dutch as *voetbalwedstrijd*).

We are aware of the fact that this definition leads to some controversial cases for English word sequences including pre-nominal adjectives. While some of them are commonly accepted such as *social policy* (German: *Sozialpolitik*), others are less accepted such as *strong wind* (German: *Starkwind*) or *small car* (German: *Kleinwagen*). This is not an unwanted side-effect. On the contrary, these controversial cases are an essential part of the iterative process we described, as they will foster linguistic discussions. Although the German *Starkwind* can be regarded as partly compositional, it is frequently used with a concrete definition (in contrast to the phrase *starker Wind*) and cannot occur in a context violating this definition, as shown in the table below.

1 a)	Als Starkwind wird meist eine Windstärke zwischen 6 und 7 Beaufort bezeichnet.
1 b)	A {strong wind} usually refers to a wind force of 6-7 Beaufort.
2 a)	Am Samstag weht ein starker Wind mit Windstärke 8 von Westen.
2 b)	On Saturday, a strong wind with wind force 8 will blow from the west.
3 a)	*Am Samstag weht ein Starkwind mit Windstärke 8 von Westen.
3 b)	On Saturday, a *{strong wind} with wind force 8 will blow from the west.

3 Multilingual extraction of NCs

This method is based on the initial definition for compound extraction described in Section 2.2 and can be adapted in succeeding iterations. English NCs are extracted from a parallel corpus that includes English and some closed compounding languages (e.g., German).

3.1 Preprocessing the parallel data

In Section 4.1, we describe the tokenization, sentence alignment, word alignment and PoS tagging we apply to the parallel data in more detail. In addition, we perform a binary compound splitter on each word that is tagged as a noun by following a variant of the methods of Stymne et al., (2013). This unsupervised splitter checks each noun for all possible segmentations into at most two components with at least two characters. All possible segmentations are scored with the geometric mean of the components' frequencies in the parallel corpus. The highest-scored segmentation (possibly with no split point) is used.

3.2 Preselection of English noun compounds using PoS patterns

As a basis for the extraction of English NCs, we use a set of possible English PoS sequences that can constitute an NC. These PoS patterns account for the various ways of composing English NCs and for the inseparability property as described in Section 2.1. Table 1 lists all plausible PoS patterns for bipartite and tripartite NCs with some examples (cf. the Penn Treebank tag set (Marcus et al., 1993)). For all examples in Table 1, we found translations to a closed compound in German, which satisfies our initial definition described in Section 2.2, e.g., *overall recovery rate* has been translated to *Gesamtrückforderungsquote*. Although the larger the number of components, the sparser the number of (correct) extractions, we create a regular expression for PoS patterns that cover English NCs with n components (where $2 \leq n \leq 10$). This regular expression combines all possible combinations of observed NC types. In the next step, we will filter noise, that occurs mostly in longer word sequences.

3.3 Noise filters

The selection of English NCs and their translations is based on automatic preprocessing, which leads to some noise due to false PoS tags or flaws in word alignment. With increasing word sequence length, the amount of noise increases. We apply several filters on each preselected NC and on their alignments to all other languages in the corpus and keep only those that pass all filters.

3.3.1 PoS filters

1. Two filters are applied to all languages: we disqualify word sequences including nouns or adjectives that (1) consist of only one character or (2) are contained in a stop list[1].

[1] ranks.nl/stopwords

PoS pattern	Example
Bipartite noun compounds	
NN	marketplace
NN NN	death penalty
JJ NN	structural policy
NN POS NN	children's development
NN IN NN	fall in population
NN IN DT NN	concussion of the brain
Tripartite noun compounds	
NN NN NN	energy security goal
JJ NN NN	overall recovery rate
NN IN NN IN NN	income per head of population
Regular expression for 2–10 components	
NN ((IN (DT)?|POS))? NN){1,9}	greenhouse gas emission allowance trading scheme
JJ NN ((IN (DT)?|POS))? (JJ)? NN){1,8}	internal energy market package

Table 1: English PoS sequences for noun compounds

2. Then, to account for PoS tagging errors in English, we collect all words and their PoS tags in the parallel corpus. For each word, we compute the probability of being tagged as a noun or adjective as given in (1).

$$P(noun/adj \mid word) = \frac{f((noun \cup adj) \cap word)}{f(word)} \quad (1)$$

We disqualify English word sequences, if they contain a noun or adjective w with $P(noun/adj \mid w) < \theta$. After testing several values for θ, we have decided to choose $\theta = 0.15$ because it has turned out to be a promising trade-off between coverage and precision (e.g., accepting words like *human* but rejecting words like *anywhere*).

3.3.2 Word alignment filter

Shortcomings in word alignment quality are remedied with three word alignment filters.

1. We truncate extraneous words (i.e., determiners, prepositions and (ad)verbs) from the border of the word sequence (adjectives are removed from the right border for Germanic languages and from the left border for Romance languages).

2. We disqualify the word sequence as being phrasal if it contains two consecutive nouns with verbs or adjectives in between or if the nouns are more than ϕ tokens apart from each other. When analysing many instances of Romance phrases aligned to an English noun compound, we observed that $\phi = 3$ is the maximum token distance two nominal components can be apart (usually separated by preposition or preposition+determiner). If the word sequence is qualified as phrasal, we add determiners and prepositions that occur in the context between the nouns, otherwise the word sequence remains unchanged.

3. We remove the word sequence if it does not contain at least one noun.

The resulting set of English word sequences that conform to the regular expression in Table 1 and their aligned and filtered word sequences are stored as a set of m-tuples of word sequences. Subsequently, we will refer to this set as the *basic set*. The basic set still contains English word sequences that do not comply with our initial definition for compound extraction (Section 2.2), i.e., that are not aligned to a closed compound. In the next step, we apply a restrictor to all NCs in the basic set and keep only those instances that pass the restrictor.

3.4 Closed compound restrictor

An English word sequence is considered to be an NC if it is represented as a one-word expression in some of the closed compounding languages (e.g., Dutch, German, Swedish, ...). Given a parallel corpus with $n > 1$ closed compounding languages, this definition leaves space for investigating the degree of cross-lingual closed compounding (deg_{closed}) which is necessary for optimal extraction quality (i.e., optimal precision and recall). Because the rules described in Section 3.3.2 still leave some word alignment errors (i.e., English word sequences that are aligned to only a part of the true translation), a single compounding language realising the English word sequence as one word (i.e., $deg_{closed} = 1$) might not be restrictive enough.

The closed compound restrictor with $deg_{closed} \geq i$ retains only English word sequences that are aligned to at least i one-word expressions in the aligned closed compounding languages. We will refer to this restrictor as $CCR(i)$ and to the resulting data set as $closed\ compound(i)$.

4 Experiments for NC Extraction

4.1 Setup

Data and preprocessing. We use the 7th release of the Europarl corpus[2]. Although the Europarl corpus comprises 21 European languages, the amount of common data they cover is rather small. This means, the more languages we use, the smaller the amount of common data. In order to get a good trade-off between cross-lingual coverage and language variation exploitation, we decided on a set of 10 languages: English, the closed compounding languages Danish, Dutch, German and Swedish, as well as Greek and the Romance languages French, Italian, Portuguese and Spanish. Instead of preprocessing the parallel corpus on our own, we exploit the already preprocessed Europarl resource of OPUS[3] (Tiedemann, 2012). This preprocessed resource is PoS tagged using TreeTagger (Schmid, 1995) for English, Dutch, German, French, Italian and Spanish and the Hunpos[4] tagger for Danish, Portuguese and Swedish. We additionally tagged the Greek data using the MATE[5] tagger. The sentence alignment provided by OPUS is restricted to language pairs. As we need a sentence representation that is parallel in all 10 languages, we apply the OPUS sentence aligner (with English as pivot) on our language set and extract a total of 884,164 parallel sentence representations. The word alignment information provided by OPUS was also based on language pairs. This means, the sentence-wise token indices has to be adapted to our updated sentence representation (which is different due to a larger language set). In OPUS, the word alignment tool GIZA++ (Och and Ney, 2003) has been used with the symmetrisation heuristics (grow-diag-final-and (Koehn et al., 2007)).

4.2 Evaluation procedure and scoring

In order to compare the added value in terms of recall and precision of each closed compound restrictor (i.e., $CCR(1)$ to $CCR(4)$), we randomly select 50 accepted and 50 rejected English word sequences for each restrictor. We rate the correctness of acceptance and rejection and compute precision and recall as given in (2) and (3). F-Score is defined as harmonic mean of precision and recall.

$$Precision = \frac{accepted \cap correct}{accepted} \tag{2}$$

$$Recall = \frac{accepted \cap correct}{(accepted \cap correct) \cup (rejected \cap incorrect)} \tag{3}$$

The precision of the basic set is measured as the accuracy of a 50 sample subset. We do not compute recall and F-Score for the basic set.

[2] statmt.org/europarl

[3] opus.lingfil.uu.se

[4] code.google.com/p/hunpos/downloads/list

[5] code.google.com/p/mate-tools

We measure the amount of closed NCs in a given closed compounding language (ccl) and for a given set of NCs (Set) by using the frequency of closed NCs relative to the number of all word sequences ($N_{Set,cll}$) (word sequences removed in Section 3.3 are excluded). Since the alignment to single words is still somewhat noisy (i.e., our compound splitter does not work error-free and there are still deficiencies in the word alignment), we select a set of 50 closed noun compound samples and rate the accuracy. The final amount of closed NCs is the product of relative frequency and accuracy, as given in (4).

$$p_{ccl}(Set) = \frac{f_{Set}(closed\ NC)}{N_{Set,cll}} \cdot acc_{ccl}(Set) \tag{4}$$

4.3 Results

Set	Size	Precision	Recall	F-Score	p_{en}
Basic set	3,178,661	38.0%	—	—	1.5%
closed compound (1)	795,518	84.0%	71.2%	77.1%	4.7%
closed compound (2)	495,837	92.0%	74.2%	82.1%	6.6%
closed compound (3)	316,330	98.0%	65.3%	78.4%	9.2%
closed compound (4)	143,121	98.0%	63.6%	77.2%	10.4%

Table 2: Extraction quality of the basic set after restrictor application

Table 2 shows the results when applying the four different degrees of the closed compound restrictor to the basic set. The first result is that using only a PoS-based method leads to a very poor extraction accuracy (38%). For the applications of the closed compound restrictors, the result is that increasing deg_{closed} means increasing precision but decreasing recall in NC extraction. The reason for this is that an aligned closed NC is generally a sufficient condition for an English NC (except for controversial cases such as *strong wind*) but not a necessary condition (i.e., a true English NC may be aligned to only periphrastic constructions). The highest F-Score (82.1%) is achieved using $CCR(2)$. We can conclude that the closed compound restrictor is a reliable method for extracting English NCs. In future work, we will use a large set of human annotators with different backgrounds in order to get a widely distributed sense of compoundhood. Moreover, instead of a binary rating, we will consider compoundhood as a continuum and compare rating scores with the amount of aligned closed compounding languages realising a closed compound in a larger parallel corpus.

The last column in Table 2 shows the amount of closed English NCs in each respective set. Since deg_{closed} correlates with the amount of closed English NCs, we can conclude that, despite the cross-lingual differences in spelling conventions attested in linguistic literature, there is a bias for a universal consensus in closed compounding.

Language	p_{ccl}
German	71.2%
Danish	63.3%
Swedish	62.2%
Dutch	58.7%

Table 3: The amounts of closed noun compounds

Table 3 shows the amounts of closed noun compounds in the closed compounding languages Danish, Dutch, German and Swedish, extracted from the closed compound (1) set. Our result shows that German is the most productive language in closed compounding (71.2%), while the other languages have a similar productivity (58-63%).

The result of our extraction method is a database of English NCs and their translations in up to 9 European languages. As described in the introduction, this database will serve several purposes. One is to study cross-lingual variation. Table 4 shows some examples of multilingual noun compound extractions from closed compound (2).

English	German	Dutch	French	Italian
automotive sector	Automobilmarkt	automobielsector	secteur automobile	mercato dell' automobile
fishing techniques	Fischfangtechniken	visserijmethoden	techniques de pêche	tecniche di pesca
timetable	Zeitplan	tijdschema	calendrier	calendario
highways	Autobahnen	snelwegen	autoroutes	autostrade
trading system	Handelssystem	handelsbestel	système commercial	sistema di scambi

Table 4: Examples of multilingual noun compounds

The examples show that English noun compounds have various realisations in European languages. Although French and Italian are open compounding languages, we do find closed compounding (e.g., *autoroutes*). Compounds such as *timetable* can also be aligned to single nouns such as *calendrier* (calendar). We found three common word formation types in Romance languages for bipartite noun compounds: (1) two nouns and a preposition in between, (2) one noun and a post-nominal adjective and (3) a single (possibly compounding) noun. Although Romance languages usually agree with respect to the word formation type, they may disagree as is the case for French and Italian for the example concerning *trading system*. One interesting observation is that while the head of *highways* (*ways*) is translated fairly literally, the modifier (*high*) is replaced by alternative aspects. On highways, cars (***Auto**bahnen* (*car-ways*)) usually drive fast (***snel**wegen* (*fast-ways*)). In future work, we will use this database for researching the nature of compoundhood in a cross-lingual perspective. The resource is publicly available for future research[6].

5 Bracketing three-noun compounds

In this section, we show a case study of how our extracted database can be used to predict the structure of NPs, more specifically to bracket tripartite noun compounds (3NCs), i.e., a composition of three bare nouns that function as one unit. Given a 3NC, we can either have RIGHT bracketing, as in *baby [bicycle seat]*, or LEFT bracketing, as in *[human rights] abuses*.

5.1 The cross-lingual bracketing method

We first start with six phrase patterns that correspond to foreign phrases that are aligned to an English 3NC, as shown in Table 5, where SN refers to a single (non-compounding) noun, FC refers to a functional context (i.e., a sequence of functional words), ADJ refers to an adjective and CNC refers to a closed (bipartite) NC (based on the splitter described in Section 3.1). Each phrase pattern contains a complex unit that is separated from the rest, e.g., a closed NC or a combination of adjective and single noun. For each pattern, we know what is the head and what is the modifier: the first phrase pattern contains only one nominal component, that can be identified as head. For the other patterns, the order is: head, FC, modifier. Based on the assumption that the aligned head corresponds to the English head, we can infer the English bracketing from the complexity of the aligned head. If the aligned head is the complex unit, the English bracketing label is RIGHT, otherwise LEFT. The third column in Table 5 shows the inferred labels for the English 3NC based on the foreign phrase pattern. For an English 3NC, we check all aligned languages for a matching phrase pattern and collect, in the case of a match, the inferred label. The majority label determines the final bracketing label.

The examples below illustrate instances for each phrase pattern, where the indices correspond to those in Table 5.

[6] www.ims.uni-stuttgart.de

	Phrase pattern in foreign language	Label for English 3NC
(1)	ADJ CNC	RIGHT
(2)	CNC FC SN	RIGHT
(3)	SN FC CNC	LEFT
(4)	SN FC ADJ SN	LEFT
(5)	ADJ SN FC SN	RIGHT
(6)	SN ADJ FC SN	RIGHT

Table 5: Phrase pattern and inferred label

(1) de: *staatliche Steueraufsichtsbehörden*
state {tax inspectorates}
"state tax inspectorates"

(2) de: *Absatzmarkt für Fahrzeuge*
{sales market} for vehicles
"car sales market"

(3) nl: *methode voor geboortebeperking*
method for {birth control}
"birth control method"

(4) sv: *brottet mot mänskliga rättigheterna*
abuses of {human rights}
"human rights abuses"

(5) da: *gennemsnitlige overførsel af data*
{average transfer} of data
"data transfer rate"

(6) es: *consumo final de energía*
{consumption final} of energy
"energy end consumption"

We observed that the initial assumption (saying that the aligned head corresponds to the English head) is not always true. Sometimes the English head and modifier are swapped in aligned languages, as illustrated in example (7).

(7) nl: *stabiele wisselkoersen*
stable {exchange rate}
"exchange rate stability"

To solve this problem, we inspect the word alignment from the phrase pattern of language l_j to the English nouns N_1, N_2 and N_3 in a 3NC. If the complex unit is aligned to $\{N_2, N_3\}$ or to $\{N_1, N_3\}$, l_j provides the label RIGHT. If the complex unit is aligned to $\{N_1, N_2\}$, l_j votes for LEFT. If the complex unit is aligned to all three nouns, this is an indicator for a word alignment error. In this case, l_j will not perform any prediction. In all other cases, the inferred label from the phrase pattern is used.

5.2 Evaluation for cross-lingual bracketing

As there are only two possible structures for 3NCs, namely LEFT or RIGHT branching, we regard this task as a binary classification and score the accuracy of class agreement. As basis, we use the basic set created in Section 3, because alignments to closed compounds are not of interest for the bracketing task. Two trained human annotators (of which one is one of the authors) individually bracket a sample of 100 randomly selected 3NCs in context. Contextual cues can help the annotator to disambiguate the structure of the English NC, so the accompanying sentences are shown to the annotator. The annotators are no domain experts and since terms in Europarl can be quite domain specific, they are allowed to look up the meaning of the constituents in a dictionary or check Google. Annotators are asked to label 3NCs as LEFT or RIGHT, or UNDECIDED if they are unclear. Furthermore, the annotators are asked to mark extraction errors. When inspecting the inter-annotator agreement for the bracketing classes (LEFT/RIGHT; i.e., 76 of 100 samples), we achieved an agreement rate of 89% and $\kappa = 0.693$ (Cohen, 1960), which means substantial agreement (Landis and Koch, 1977). Afterwards, the annotators discuss disagreements and revise their annotations. This has led to a perfect agreement in our setting. The 8 UNDECIDED labellings show that in some cases the bracketing remains ambiguous even in context. In future work, we would like to investigate if larger contexts or domain knowledge is necessary for the disambiguation process or if the NCs are inherently flat (i.e., if LEFT or RIGHT bracketing does not make any difference in meaning). We evaluate our cross-lingual bracketing system for (1) inferred label of a phrase pattern and (2) word

alignment information for phrase pattern with inferred label as back-off. We compare the bracketing performance against the LEFT class baseline.

5.3 Results

Method	Accuracy
LEFT baseline	71.1 %
Inferred phrase pattern labels	89.0† %
Word alignment for phrase patterns	91.6† %

Table 6: Bracketing performance; † indicates significantly higher than the LEFT baseline

Table 6 shows the results of our system compared to the LEFT class baseline. The first result is that both inferred label and word alignment information for phrase pattern outperform the LEFT class baseline significantly[7]. Bracketing with word alignment information for phrase pattern outperforms bracketing based on the inferred labels.

6 Related Work

Our methods for extracting and structuring English NCs rely on the spelling of various aligned languages. Previous work on multilingual extraction include Morin and Daille (2010) and Weller and Heid (2012). These type-based approaches focus on bilingual terminology extraction using comparable corpora. Our token-based extraction method includes 10 languages and we extract both the NCs and their context. While the aforementioned work serves as resource for improving machine translation (MT) systems, we focus on NC research and how multilingual evidence can help analysing and interpreting English NCs.

This multilingual perspective on a considerable number of languages has been adopted as well by Macherey et al., (2011), who present a multilingual language-independent approach to compound splitting. Moreover, they learned morphological operations on compounding automatically. Here, Macherey et al., (2011) extract training instances using a method related to Garera and Yarowsky (2008): select a single word f in a language l translated to several English words e_i . If there is a translation for each e_i to a word g_i that shows a (partial) substring match with f, $(f; e_1, \ldots, e_n; g_1, \ldots, g_n)$ is extracted. While Macherey et al., (2011) extract training instances type-based in a bilingual setting, we directly extract NC instances with a set of four closed compounding languages. This token-based perspective has the advantage that we can process English NCs for which there is no literal translation to the target language (e.g., *health insurance* aligned to *Krankenversicherung* (lit. invalid insurance)).

In cross-lingual annotation transfer (Yarowsky and Ngai, 2001; Padó, 2007; Van der Plas et al., 2011) human annotations are transferred from one language to the other in parallel data. In this paper, we use the structural differences between languages as found in parallel corpora to generate annotations on the target language and do not rely on annotations on the source language.

Bracketing methods for both three-noun compounds and complete base NPs have been designed both supervised and unsupervised. Vadas and Curran (2007) used a supervised bracketing method on manually annotated data. Pitler et al. (2010) used the data from Vadas and Curran (2007) for a parser applicable on base NPs of any length including coordinations. Their supervised classifier exploited web-scale N-grams. Although supervised methods outperform unsupervised methods by far, the need for annotated data is a drawback of supervised approaches. Bergsma et al. (2011) used crosslingual data as additional supervision to make the need for manual annotations less pressing. Unsupervised methods use N-gram statistics (Marcus, 1980; Lauer, 1995; Nakov and Hearst, 2005) or semantic information (Kim and Baldwin, 2013).

7 Conclusion

In this paper, we discussed the complexity related to the definition of compoundhood and presented an iterative method that tries to refine existing definitions by tentatively demonstrating the efficacy of

[7] Approximate randomization test (Yeh, 2000), $p < 5\%$

linguistic criteria on corpus data. The initial implementation of two linguistic criteria, based on cross-lingual spelling conventions and the inseparability of a compound's components, achieved an F-Score of 82.1% on the task of extracting English compounds.

The extracted multilingual database of compounds in contexts serves multiple purposes. For example, it can be used to study cross-lingual variations in compounding. We showed in an additional experiment how the cross-lingual evidence found in the multilingual database can be used to bracket English three-noun compounds using cross-lingual spelling variation with a set of six phrase patterns. We achieved a bracketing accuracy of 91.6% that is very close to human performance.

In future work, we plan to continue refining the definition of compoundhood in a cross-lingual setting. We will experiment with additional linguistic criteria defined over multiple languages. This way, we hope to improve the quality of the multilingual database that we will further explore for compound analysis and translation.

Acknowledgements

This research was funded and supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) as part of the SFB 732. We thank the anonymous reviewers for their comments. We also thank Gianina Iordachioaia for her helpful input and interesting discussion.

References

Marco Baroni, Johannes Matiasek, and Harald Trost. 2002. Predicting the components of German nominal compounds. In *Proceedings of ECAI*, pages 470–474, Lyon. IOS Press.

L. Bauer. 2003. *Introducing Linguistic Morphology*. Introducing Linguistic Morphology. Edinburgh University Press.

Otto Behaghel. 1909. Beziehungen zwischen umfang und reihenfolge von satzgliedern. *Indogermanische Forschungen*, page 110142.

Shane Bergsma, David Yarowsky, and Kenneth Church. 2011. Using large monolingual and bilingual corpora to improve coordination disambiguation. In *ACL-HLT 2011*, pages 1346–1355.

Pierrette Bouillon, Katharina Boesefeldt, and Graham Russell. 1992. Compound Nouns in a Unification-Based MT System. In *ANLP 1992*, pages 209–215, Trento.

Fabio Celli and Malvina Nissim. 2009. Automatic identification of semantic relations in Italian complex nominals. In *IWCS 2009*, pages 45–60.

Eve V. Clark. 1981. Lexical innovations: How children learn to create new words. In Werner Deutsch, editor, *The Child's Construction of Language*, pages 299–328. Academic Press, New York.

J. Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1).

Nikesh Garera and David Yarowsky. 2008. Translating compounds by learning component gloss translation models via multiple languages. In *IJCNLP*, pages 403–410.

Roxana Girju. 2007. Improving the Interpretation of Noun Phrases with Cross-linguistic Information. In *ACL 2007*, pages 568–575.

Iris Hendrickx, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale. 2013. Semeval-2013 task 4: Free paraphrases of noun compounds. In *Workshop on Semantic Evaluation (SemEval 2013)*, pages 138–143.

Michael Johnston and Frederica Busa. 1999. Qualia structure and the compositional interpretation of compounds. In *E. Viegas (ed.), Breadth and depth of semantics lexicons*, pages 167–187. Dordrecht: Kluwer Academic.

Su Nam Kim and Timothy Baldwin. 2006. Interpreting semantic relations in noun compounds via verb semantics. In *ACL 2006*, pages 491–498.

Su Nam Kim and Timothy Baldwin. 2013. A lexical semantic approach to interpreting and bracketing english noun compounds. *Natural Language Engineering*, 19(3):385–407.

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

J. R. Landis and G. G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174.

Mark Lauer. 1995. *Designing Statistical Language Learners: Experiments on Noun Compounds*. Ph.D. thesis, Macquarie University.

R. Lieber and P. Stekauer. 2009. *The Oxford Handbook of Compounding*. Oxford Handbooks in Linguistics. OUP Oxford.

Klaus Macherey, Andrew M. Dai, David Talbot, Ashok C. Popat, and Franz Och. 2011. Language-independent compound splitting with morphological operations. In *ACL-HLT 2011*.

Hans Marchand. 1967. Expansion, transposition, and derivation. *La Linguistique*, pages 13–26.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, June.

Mitchell Marcus. 1980. *A Theory of Syntactic Recognition for Natural Language*. MIT Press.

Emmanuel Morin and Batrice Daille. 2010. Compositionality and lexical alignment of multi-word terms. *Language Resources and Evaluation*, 44:79–95.

Preslav Nakov and Marti Hearst. 2005. Search engine statistics beyond the n-gram: Application to noun compound bracketing. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CoNLL 2005, pages 17–24.

Roberto Navigli, Paola Velardi, and Aldo Gangemi. 2003. Ontology learning and its application to automated terminology translation. *IEEE Intelligent Systems*, 18(1):22–31.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*

Diarmuid Ó Séaghdha. 2008. *Learning compound noun semantics*. Ph.D. thesis, Computer Laboratory, University of Cambridge.

S. Padó. 2007. *Cross-lingual Annotation Projection Models for Role-Semantic Information*. Ph.D. thesis, Saarland University.

Emily Pitler, Shane Bergsma, Dekang Lin, and Kenneth Ward Church. 2010. Using web-scale n-grams to improve base np parsing performance. In *COLING 2010*, pages 886–894.

Ulrike Rackow, Ido Dagan, and Ulrike Schwall. 1992. Automatic translation of noun compounds. In *COLING 1992*, pages 1249–1253.

Helmut Schmid. 1995. Improvements in part-of-speech tagging with an application to german. In *ACL SIGDAT-Workshop 1995*, pages 47–50.

Takaaki Tanaka and Timothy Baldwin. 2003. Noun-noun compound machine translation: A feasibility study on shallow processing. In *ACL-2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, page 1724.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *LREC 2012*.

David Vadas and James R. Curran. 2007. Large-scale supervised models for noun phrase bracketing. In *PACLING 2007*, pages 104–112.

L. Van der Plas, P. Merlo, and J. Henderson. 2011. Scaling up cross-lingual semantic annotation transfer. In *ACL-HLT 2011*.

Marion Weller and Ulrich Heid. 2012. Analyzing and aligning german compound nouns. In *LREC 2012*, Istanbul, Turkey.

D. Yarowsky and G. Ngai. 2001. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *NAACL 2001*, pages 1–8.

A. Yeh. 2000. More accurate tests for the statistical significance of result differences. In *COLING 2000*.

Automatic Classification of Communicative Functions of Definiteness

Archna Bhatia[*,‡] **Chu-Cheng Lin**[*] **Nathan Schneider**[*] **Yulia Tsvetkov**[*]
Fatima Talib Al-Raisi[*] **Laleh Roostapour**[*] **Jordan Bender**[†] **Abhimanu Kumar**[*]
Lori Levin[*] **Mandy Simons**[*] **Chris Dyer**[*]

[*]Carnegie Mellon University [†]University of Pittsburgh
Pittsburgh, PA 15213 Pittsburgh, PA 15260
[‡]archnab@cs.cmu.edu

Abstract

Definiteness expresses a constellation of semantic, pragmatic, and discourse properties—the communicative functions—of an NP. We present a supervised classifier for English NPs that uses lexical, morphological, and syntactic features to predict an NP's communicative function in terms of a language-universal classification scheme. Our classifiers establish strong baselines for future work in this neglected area of computational semantic analysis. In addition, analysis of the features and learned parameters in the model provides insight into the grammaticalization of definiteness in English, not all of which is obvious a priori.

1 Introduction

Definiteness is a morphosyntactic property of noun phrases (NPs) associated with semantic and pragmatic characteristics of entities and their discourse status. Lyons (1999), for example, argues that definite markers prototypically reflect *identifiability* (whether a referent for the NP can be identified by the discourse participants or not); other aspects identified in the literature include *uniqueness* of the entity in the world and whether the hearer is already *familiar* with the entity given the context and preceding discourse (Roberts, 2003; Abbott, 2006). While some morphosyntactic forms of definiteness are employed by all languages—namely, demonstratives, personal pronouns, and possessives—languages display a vast range of variation with respect to the form and meaning of definiteness. For example, while languages like English make use of definite and indefinite articles to distinguish between the discourse status of various entities (***the*** *car* vs. ***a*** *car* vs. *cars*), many other languages—including Czech, Indonesian, and Russian—do not have articles (although they do have demonstrative determiners). Sometimes definiteness is marked with affixes or clitics, as in Arabic. Sometimes it is expressed with other constructions, as in Chinese (a language without articles), where the existential construction can be used to express indefinite subjects and the *ba-* construction can be used to express definite direct objects (Chen, 2004).

Aside from this variation in the form of (in)definite NPs within and across languages, there is also variability in the mapping between semantic, pragmatic, and discourse functions of NPs and the (in)definites expressing these functions. We refer to these as *communicative functions* of definiteness, following Bhatia et al. (2014). Croft (2003, pp. 6–7) shows that even when two languages have access to the same morphosyntactic forms of definiteness, the conditions under which an NP is marked as definite or indefinite (or not at all) are language-specific. He illustrates this by contrasting English and French translations (both languages use definite as well as indefinite articles) such as:

(1) He showed **extreme care**. (unmarked)
Il montra **<u>un</u> soin extrême**. (indef.)

(2) I love **artichokes** and asparagus. (unmarked)
J'aime **<u>les</u> artichauts** et les asperges. (def.)

(3) His brother became **<u>a</u> soldier**. (indef.)
Son frère est devenu **soldat**. (unmarked)

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1059–1070, Dublin, Ireland, August 23-29 2014.

- **NONANAPHORA** [$-A,-B$] **999**
  - **UNIQUE** [$+U$] **287**
    - **UNIQUE_HEARER_OLD** [$+F,-G,+S$] **251**
      - UNIQUE_PHYSICAL_COPRESENCE [$+R$] 13
      - UNIQUE_LARGER_SITUATION [$+R$] 237
      - UNIQUE_PREDICATIVE_IDENTITY [$+P$] 1
    - UNIQUE_HEARER_NEW [$-F$] 36
  - **NONUNIQUE** [$-U$] **581**
    - **NONUNIQUE_HEARER_OLD** [$+F$] **169**
      - NONUNIQUE_PHYSICAL_COPRESENCE [$-G,+R,+S$] 39
      - NONUNIQUE_LARGER_SITUATION [$-G,+R,+S$] 117
      - NONUNIQUE_PREDICATIVE_IDENTITY [$+P$] 13
    - NONUNIQUE_HEARER_NEW_SPEC [$-F,-G,+R,+S$] 231
    - NONUNIQUE_NONSPEC [$-G,-S$] 181
  - **GENERIC** [$+G,-R$] **131**
    - GENERIC_KIND_LEVEL 0
    - GENERIC_INDIVIDUAL_LEVEL 131
- **ANAPHORA** [$+A$] **1574**
  - **BASIC_ANAPHORA** [$-B,+F$] **795**
    - SAME_HEAD 556
    - DIFFERENT_HEAD 329
  - **EXTENDED_ANAPHORA** [$+B$] **779**
    - BRIDGING_NOMINAL [$-G,+R,+S$] 43
    - BRIDGING_EVENT [$+R,+S$] 10
    - BRIDGING_RESTRICTIVE_MODIFIER [$-G,+S$] 614
    - BRIDGING_SUBTYPE_INSTANCE [$-G$] 0
    - BRIDGING_OTHER_CONTEXT [$+F$] 112
- **MISCELLANEOUS** [$-R$] **732**
  - PLEONASTIC [$-B,-P$] 53
  - QUANTIFIED 248
  - PREDICATIVE_EQUATIVE_ROLE [$-B,+P$] 58
  - PART_OF_NONCOMPOSITIONAL_MWE 100
  - MEASURE_NONREFERENTIAL 125
  - OTHER_NONREFERENTIAL 148

	+	−	0		+	−	0		+	−	0		+	−	0
Anaphoric	1574	999	732	Generic	131	1476	1698	Predicative	72	53	3180	Specific	1305	181	1819
Bridging	779	1905	621	Familiar	1327	267	1711	Referential	690	863	1752	Unique	287	581	2437

Figure 1: CFD (Communicative Functions of Definiteness) annotation scheme, with frequencies in the corpus. Internal (non-leaf) labels are in bold; these are not annotated or predicted. +/− values are shown for ternary attributes Anaphoric, Bridging, Familiar, Generic, Predicative, Referential, Specific, and Unique; these are inherited from supercategories, but otherwise default to 0. Thus, for example, the full attribute specification for UNIQUE_PHYSICAL_COPRESENCE is $[-A,-B,+F,-G,0P,+R,+S,+U]$. Counts for these attributes are shown in the table at bottom.

A cross-linguistic classification of communicative functions should be able to characterize the aspects of meaning that account for the different patterns of definiteness marking exhibited in (1–3): e.g., that (2) concerns a generic class of entities while (3) concerns a role filled by an individual. For more on communicative functions, see §2.

This paper develops supervised classifiers to predict communicative function labels for English NPs using lexical, morphological, and syntactic features. The contribution of our work is in both the output of the classifiers and the models themselves (features and weights). Each classifier predicts communicative function labels that capture aspects of discourse-newness, uniqueness, specificity, and so forth. Such functions are useful in a variety of language processing applications. For example, they should usually be preserved in translation, even when the grammatical mechanisms for expressing them are different. The communicative function labels also represent the discourse status of entities, making them relevant for entity tracking, knowledge base construction, and information extraction.

Our log-linear model is a form-meaning mapping that relates syntactic, lexical, and morphological features to properties of communicative functions. The learned weights of this model can, e.g., generate plausible hypotheses regarding the form-meaning relationship which can then be tested rigorously through controlled experiments. This hypothesis generation is linguistically significant as it indicates new grammatical mechanisms beyond the obvious *a* and *the* articles that are used for expressing definiteness in English.

To build our models, we leverage a cross-lingual definiteness annotation scheme (§2) and annotated English corpus (§3) developed in prior work (Bhatia et al., 2014). The classifiers, §4, are supervised models with features that combine lexical and morphosyntactic information and the prespecified attributes or groupings of the communicative function labels (such as Anaphoric, Bridging, Specific in fig. 1) to predict leaf labels (the non-bold faced labels in fig. 1); the evaluation measures (§5) include one that exploits these label groupings to award partial credit according to relatedness. §6 presents experiments comparing several models and discussing their strengths and weaknesses; computational work and applications related to definiteness are addressed in §7.

2 Annotation scheme

The literature on definiteness describes functions such as uniqueness, familiarity, identifiability, anaphoricity, specificity, and referentiality (Birner and Ward, 1994; Condoravdi, 1992; Evans, 1977, 1980; Gundel et al., 1988, 1993; Heim, 1990; Kadmon, 1987, 1990; Lyons, 1999; Prince, 1992; Roberts, 2003; Russell, 1905, *inter alia*) as being related to definiteness. Reductionist approaches to definiteness try to define it in terms of one or two of the aforementioned communicative functions. For example, Roberts (2003) proposes that the combination of uniqueness and a presupposition of familiarity underlie all definite descriptions. However, possessive definite descriptions (*John's daughter*) and the weak definites (*the son of Queen Juliana of the Netherlands*) are neither unique nor necessarily familiar to the listener before they are spoken. In contrast to the reductionist approaches are approaches to grammaticalization (Hopper and Traugott, 2003) in which grammar develops over time in such a way that each grammatical construction has some prototypical communicative functions, but may also have many non-prototypical communicative functions. The scheme we are adopting for this work—the annotation scheme for Communicative Functions of Definiteness (CFD) as described in Bhatia et al. (2014)—assumes that there may be multiple functions to definiteness. CFD is based on a combination of these functions and is summarized in fig. 1. It was developed by annotating texts in two languages (English and Hindi) for four different genres—namely TED talks, a presidential inaugural speech, news articles, and fictional narratives—keeping in mind the communicative functions that have been associated with definiteness in the linguistic literature.

CFD is hierarchically organized. This hierarchical organization serves to reduce the number of decisions that an annotator needs to make for speed and consistency. We now highlight some of the major distinctions in the hierarchy.

At the highest level, the distinction is made between **Anaphora**, **Nonanaphora**, and **Miscellaneous** functions of an NP (the annotatable unit). **Anaphora** and **Nonanaphora** respectively describe whether an entity is old or new in the discourse; the **Miscellaneous** function is mainly assigned to various kinds of nonreferential NPs.

The **Anaphora** category has two subcategories: **Basic_Anaphora** and **Extended_Anaphora**. **Basic_Anaphora** applies to NPs referring to entities that have been mentioned before. **Extended_Anaphora** applies to any NP whose referent has not been mentioned itself, but is evoked by a previously mentioned entity. For example, after mentioning a wedding, *the bride*, *the groom*, and *the cake* are considered to be **Extended_Anaphora**.

Within the **Nonanaphora** category, a first distinction is made between **Unique**, **Nonunique**, and **Generic**. The **Unique** function applies to NPs whose referent becomes unique in a context for any of several reasons. For example, *Obama* can safely be considered unique in contemporary political discourse in the United States. The function **Nonunique** applies to NPs that start out with multiple possible referents and that may or may not become identifiable in a speech situation. For example, *a little riding hood of red velvet* in fig. 2 could be annotated with the label **Nonunique**. Finally, **Generic** NPs refer to classes or types of entities rather than specific entities. For example, *Dinosaurs* in *Dinosaurs are extinct.* is a **Generic** NP.

Another important distinction CFD makes is between **Hearer_Old** for references to entities that are familiar to the hearer (e.g., if they are physically present in the speech situation), versus **Hearer_New** for nonfamiliar references. This distinction cuts across the two subparts of the hierarchy, **Anaphora** and **Nonanaphora**; thus, labels marking **Hearer_Old** or **Hearer_New** also encode other distinctions (e.g., **Unique_Hearer_Old**, **Unique_Hearer_New**, **Nonunique_Hearer_Old**). For further details on the annotation scheme, see fig. 1 and Bhatia et al. (2014).

Because the ordering of distinctions determines the tree structure of the hierarchy, the same communicative functions could have been organized in a superficially different way. In fact, Komen (2013) has proposed a hierarchy with similar leaf nodes, but different internal structure. Since it is possible that some natural groupings of labels are not reflected in the hierarchy we used, we also decompose each label into fundamental communicative functions, which we call *attributes*. Each label type is associated with values for attributes Anaphoric, Bridging, Familiar, Generic, Predicative, Referential, Specific, and Unique. These attributes can have values of +, −, or 0, as shown in fig. 1. For instance, with the Anaphoric

Once upon a time there was a dear little girl who was loved by everyone who looked at her, but most of all by her grandmother, and there was nothing that she would not have given to the child.

Once she (SAME_HEAD) gave her (DIFFERENT_HEAD) a little riding hood of red velvet (OTHER_NONREFERENTIAL), (NONUNIQUE_HEARER_NEW_SPEC) which suited her (SAME_HEAD) so well that she (SAME_HEAD) would never wear anything else (QUANTIFIED); so she (SAME_HEAD) was always called 'Little Red Riding Hood.' (UNIQUE_HEARER_NEW)

Figure 2: An annotated sentence from "Little Red Riding Hood." The previous sentence is shown for context.

attribute, a value of + applies to labels that can never mark NPs new to the discourse, − applies to labels that can *only* apply if the NP is new in the discourse, and 0 applies to labels such as **Pleonastic** (where anaphoricity is not applicable because there is no discourse referent).

3 Data

We use the English definiteness corpus of Bhatia et al. (2014), which consists of texts from multiple genres annotated with the scheme described in §2.[1] The 17 documents consist of prepared speeches (TED talks and a presidential address), published news articles, and fictional narratives. The TED data predominates (75% of the corpus);[2] the presidential speech represents about 16%, fictional narratives 5%, and news articles 4%. All told, the corpus contains 13,860 words (868 sentences), with 3,422 NPs (the annotatable units). Bhatia et al. (2014) report high inter-annotator agreement, estimating Cohen's $\kappa = 0.89$ within the TED genre as well as for all genres.

Figure 2 is an excerpt from the "Little Red Riding Hood" annotated with the CFD scheme.

4 Classification framework

To model the relationship between the grammar of definiteness and its communicative functions in a data-driven fashion, we work within the supervised framework of feature-rich discriminative classification, treating the functional categories from §2 as output labels y and various lexical, morphological, and syntactic characteristics of the language as features of the input x. Specifically, we learn two kinds of probabilistic models. The first is a log-linear model similar to multiclass logistic regression, but deviating in that logistic regression treats each output label (response) as atomic, whereas we decompose each into *attributes* based on their linguistic definitions, enabling commonalities between related labels to be recognized. Each weight in the model corresponds to a feature that mediates between *percepts* (characteristics of the input NP) and attributes (characteristics of the label). This is aimed at attaining better predictive accuracy as well as feature weights that better describe the form–function interactions we are interested in recovering. We also train a random forest model on the hypothesis that it would allow us to sacrifice interpretability of the learned parameters for predictive accuracy.

Our setup is formalized below, where we discuss the mathematical models and linguistically motivated features.

4.1 Models

We experiment with two classification methods: a log-linear model and a nonlinear tree-based ensemble model. Due to their consistency and interpretability, linear models are a valuable tool for quantifying and analyzing the effects of individual features. Non-linear models, while less interpretable, often outperform logistic regression (Perlich et al., 2003), and thus could be desirable when the predictions are needed for a downstream task.

[1] The data can be obtained from `http://www.cs.cmu.edu/~ytsvetko/definiteness_corpus`.

[2] The TED talks are from a large parallel corpus obtained from `http://www.ted.com/talks/`.

4.1.1 Log-linear model

At test time, we model the probability of communicative function label y conditional on an NP x as follows:

$$p_{\boldsymbol{\theta}}(y|x) = \log \frac{\exp \boldsymbol{\theta}^\top \mathbf{f}(x,y)}{\sum_{y' \in \mathcal{Y}} \exp \boldsymbol{\theta}^\top \mathbf{f}(x,y')} \tag{1}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is a vector of parameters (feature weights), and $\mathbf{f} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ is the feature function over input–label pairs. The feature function is defined as follows:

$$\mathbf{f}(x,y) = \boldsymbol{\phi}(x) \times \tilde{\boldsymbol{\omega}}(y) \tag{2}$$

where the percept function $\boldsymbol{\phi} : \mathcal{X} \to \mathbb{R}^c$ produces a vector of real-valued characteristics of the input, and the attribute function $\tilde{\boldsymbol{\omega}} : \mathcal{Y} \to \{0,1\}^a$ encodes characteristics of each label. There is a feature for every percept–attribute pairing: so $d = c \cdot a$ and $f_{(i-1)a+j}(x,y) = \phi_i(x)\tilde{\omega}_j(y), 1 \le i \le c, 1 \le j \le a$.[3] The contents of the percept and attribute functions are detailed in §4.2 and §4.3 respectively.

For prediction, having learned weights $\hat{\boldsymbol{\theta}}$ we use the Bayes-optimal decision rule for minimizing misclassification error, selecting the y that maximizes this probability:

$$\hat{y} \leftarrow \arg\max_{y \in \mathcal{Y}} p_{\hat{\boldsymbol{\theta}}}(y|x) \tag{3}$$

Training optimizes $\hat{\boldsymbol{\theta}}$ so as to maximize a convex L_2-regularized[4] learning objective over the training data $\mathcal{D}$:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} -\lambda ||\boldsymbol{\theta}||_2^2 + \sum_{\langle x,y \rangle \in \mathcal{D}} \log \frac{\exp \boldsymbol{\theta}^\top \mathbf{f}(x,y)}{\sum_{y' \in \mathcal{Y}} \exp \left(\boldsymbol{\theta}^\top \mathbf{f}(x,y') \right)} \tag{4}$$

With $\tilde{\boldsymbol{\omega}}(y) =$ *the identity of the label*, this reduces to standard logistic regression.

4.1.2 Non-linear model

We employ a random forest classifier (Breiman, 2001), an ensemble of decision tree classifiers learned from many independent subsamples of the training data. Given an input, each tree classifier assigns a probability to each label; those probabilities are averaged to compute the probability distribution across the ensemble.

An important property of the random forests, in addition to being an effective tool in prediction, is their immunity to overfitting: as the number of trees increases, they produce a limiting value of the generalization error.[5] Thus, no hyperparameter tuning is required. Random forests are known to be robust to sparse data and to label imbalance (Chen et al., 2004), both of which are challenges with the definiteness dataset.

4.2 Percepts

The characteristics of the input that are incorporated in the model, which we call *percepts* to distinguish them from model features linking inputs to outputs, see §4.1, are intended to capture the aspects of English morphosyntax that may be relevant to the communicative functions of definiteness.

After preprocessing the text with a dependency parser and coreference resolver, which is described in §6.1, we extract several kinds of percepts for each NP.

4.2.1 Basic

Words of interest. These are the *head* within the NP, all of its *dependents*, and its *governor* (external to the NP). We are also interested in the *attached verb*, which is the first verb one encounters when traversing the dependency path upward from the head. For each of these words, we have separate percepts capturing: the token, the part-of-speech (POS) tag, the lemma, the dependency relation, and (for the head only) a

[3] Chahuneau et al. (2013) use a similar parametrization for their model of morphological inflection.

[4] As is standard practice with these models, bias parameters (which capture the overall frequency of percepts/attributes) are excluded from regularization.

[5] See Theorem 1.2 in Breiman (2001) for details.

binary indicator of plurality (determined from the POS tag). As there may be multiple dependents, we have additional features specific to the first and the last one. Moreover, to better capture tense, aspect and modality, we collect the attached verb's *auxiliaries*. We also make note of the negative particle (with dependency label *neg*) if it is a dependent of the verb.

Structural. The structural percepts are: the *path length* from the head up to the root, and to the attached verb. We also have percepts for the number of dependents, and the number of dependency relations that link non-neighbors. Integer values were binarized with thresholding.

Positional. These percepts are the *token length* of the NP, the NP's *location* in the sentence (first or second half), and the *attached verb's position* relative to the head (left or right). 12 additional percept templates record the POS and lemma of the left and right neighbors of the head, governor, and attached verb.

4.2.2 Contextual NPs

When extracting features for a given NP (call it the "target"), we also consider NPs in the following relationship with the target NP: its *immediate parent*, which is the smallest NP whose span fully subsumes that of the target; the *immediate child*, which is the largest NP subsumed within the target; the *immediate precedent* and *immediate successor* within the sentence; and the *nearest preceding coreferent mention*.

For each of these related NPs, we include all of their basic percepts conjoined with the nature of the relation to the target.

4.3 Attributes

As noted above, though CFD labels are organized into a tree hierarchy, there are actually several dimensions of commonality that suggest different groupings. These attributes are encoded as ternary characteristics; for each label (including internal labels), every one of the 8 attributes is assigned a value of +, −, or 0 (refer to fig. 1). In light of sparse data, we design features to exploit these similarities via the attribute vector function

$$\boldsymbol{\omega}(y) = [y, A(y), B(y), F(y), G(y), P(y), R(y), S(y), U(y)]^\top \tag{5}$$

where $A : \mathcal{Y} \to \{+, -, 0\}$ returns the value for Anaphoric, $B(y)$ for Bridging, etc. The identity of the label is also included in the vector so that different labels are always recognized as different by the attribute function. The categorical components of this vector are then binarized to form $\tilde{\boldsymbol{\omega}}(y)$; however, instead of a binary component that fires for the 0 value of each ternary attribute, there is a component that fires for *any* value of the attribute—a sort of bias term. The weights assigned to features incorporating + or − attribute values, then, are easily interpreted as deviations relative to the bias.

5 Evaluation

The following measures are used to evaluate our predictor against the gold standard for the held-out evaluation (dev or test) set $\mathcal{E}$:

- **Exact Match:** This accuracy measure gives credit only where the predicted and gold labels are identical.
- **By leaf label:** We also compute precision and recall of each leaf label to determine which categories are reliably predicted.
- **Soft Match:** This accuracy measure gives partial credit where the predicted and gold labels are related. It is computed as the proportion of attributes-plus-full-label whose (categorical) values match: $|\boldsymbol{\omega}(y) \cap \boldsymbol{\omega}(y')|/9$.

6 Experiments

6.1 Experimental Setup

Data splits. The annotated corpus of Bhatia et al. (2014) (§3) contains 17 documents in 3 genres: 13 prepared speeches (mostly TED talks),[6] 2 newspaper articles, and 2 fictional narratives. We arbitrarily choose some documents to hold out from each genre; the resulting test set consists of 2 TED talks

[6]We have combined the TED talks and presidential speech genres since both involved prepared speeches.

Condition	$\|\boldsymbol{\theta}\|$	λ	Exact Match Acc.	Soft Match Acc.
Majority baseline	—	—	12.1	47.8
Log-linear classifier, attributes only	473,064	100	38.7	77.1
Log-linear classifier, labels only	413,931	100	40.8	73.6
Full log-linear classifier (labels + attributes)	926,417	100	43.7	**78.2**
Random forest classifier	20,363	—	**49.7**	77.5

Table 1: Classifiers and baseline, as measured on the test set. The first two columns give the number of parameters and the tuned regularization hyperparameter, respectively; the third and fourth columns give accuracies as percentages. The best in each column is bolded.

("Alisa_News", "RobertHammond_park"), 1 newspaper article ("crime1_iPad_E"), and 1 narrative ("Little Red Riding Hood"). The test set then contains 19,28 tokens (111 sentences), in which there are 511 annotated NPs; while the training set contains 2,911 NPs among 11,932 tokens (757 sentences).

Preprocessing. Automatic dependency parses and coreference information were obtained with the parser and coreference resolution system in Stanford CoreNLP v. 3.3.0 (Socher et al., 2013; Recasens et al., 2013) for use in features (§4.2). Syntactic features were extracted from the Basic dependencies output by the parser. To evaluate the performance of Stanford system on our data, we manually inspected the dependencies and coreference information for a subset of sentences from our corpus (using texts from TED talks and fictional narratives genres) and recorded the errors. We found that about 70% of the sentences had all correct dependencies, and only about 0.04% of the total dependencies were incorrect for our data. However, only 62.5% of the coreference links were correctly identified by the coreference resolver. The rest of them were either missing or incorrectly identified. We believe this may have caused a portion of the classifier errors while predicting the Ananphoic labels.

Throughout our experiments (training as well as testing), we use the gold NP boundaries identified by the human annotators. The automatic dependency parses are used to extract percepts for each gold NP. If there is a conflict between the gold NP boundaries and the parsed NP boundaries, to avoid extracting misleading percepts, we assign a default value.

Learning. The log-linear model variants are trained with an in-house implementation of supervised learning with L_2-regularized AdaGrad (Duchi et al., 2011). Hyperparameters are tuned on a development set formed by holding out every tenth instance from the training set (test set experiments use the full training set): the power of 10 giving the highest Soft Match accuracy was chosen for λ.[7] The Python `scikit-learn` toolkit (Pedregosa et al., 2011) was used for the random forest classifier.[8]

6.2 Results

Measurements of overall classification performance appear in table 1. While far from perfect, our classifiers achieve promising accuracy levels given the small size of the training data and the number of labels in the annotation scheme. The random forest classifier is the most accurate in Exact Match, likely due to the robustness of that technique under conditions where the data are small and the frequencies of individual labels are imbalanced. By the Soft Match measure, our attribute-aware log-linear models perform very well. The most successful of the log-linear models is the richest model, which combines the fine-grained communicative function labels with higher-level attributes of those labels. But notably the attribute-only model, which decomposes the semantic labels into attributes without directly considering the full label, performs almost as well as the random forest classifier in Soft Match. This is encouraging because it suggests that the model has correctly exploited known linguistic generalizations to account for the grammaticalization of definiteness in English.

Table 2 reports the precision and recall of each leaf label predicted. Certain leaf labels are found to be easier for the classifier to predict: e.g., the communicative function label **Pleonastic** has a high F_1 score. This is expected as the **Ploenastic** CFD for English is quite regular and captured by the `EX`

[7] Preliminary experiments with cross-validation on the training data showed that the value of λ was stable across folds.

[8] Because it is a randomized algorithm, the results may vary slightly between runs; however, a cross-validation experiment on the training data found very little variance in accuracy.

Leaf label	N	P	R	F_1	Leaf label	N	P	R	F_1
Pleonastic	44	100	78	88	Part_of_Noncompositional_MWE	88	20	17	18
Bridging_Restrictive_Modifier	552	58	84	68	Bridging_Nominal	33	33	10	15
Quantified	213	57	57	57	Generic_Individual_Level	113	14	11	13
Unique_Larger_Situation	97	52	58	55	Nonunique_Nonspec	173	9	25	13
Same_Head	452	41	41	41	Bridging_Other_Context	96	33	6	11
Measure_Nonreferential	98	88	26	40	Bridging_Event	9	—	0	—
Nonunique_Hearer_New_Spec	190	36	46	40	Nonunique_Physical_Copresence	36	0	0	—
Other_Nonreferential	134	39	36	37	Nonunique_Predicative_Identity	10	—	0	—
Different_Head	271	32	33	32	Predicative_Nonidentity	57	0	0	—
Nonunique_Larger_Situation	97	29	25	27	Unique_Hearer_New	26	—	0	—

Table 2: Number of training set instances and precision, recall, and F_1 percentages for leaf labels.

part-of-speech tag. The classifier finds predictions of certain CFD labels, such as **Bridging_Event**, **Bridging_Nominal** and **Nonunique_Nonspecific**, to be more difficult due to data sparseness: it appears that there were not enough training instances for the classifier to learn the generalizations corresponding to these CFDs. **Bridging_Other_Context** was hard to predict as this was a category which referred not to the entities previously mentioned but to the whole speech event from the past. There seem to be no clear morphosyntactic cues associated with this CFD, so to train a classifier to predict this category label, we would need to model more complex semantic and discourse information. This also applies to the classifier confusion between the **Same_Head** and **Different_Head**, since both of these labels share all the semantic attributes used in this study.

An advantage of log-linear models is that inspecting the learned feature weights can provide useful insights into the model's behavior. Figure 3 lists 10 features that received the highest positive weights in the full model for the + and − values of the Specific attribute. These confirm some known properties of English definites and indefinites. The definite article, possessives (`PRP$`), proper nouns (`NNP`), and the second person pronoun are all associated with specific NPs, while the indefinite article is associated with nonspecific NPs. The model also seems to have picked up on the less obvious but well-attested tendency of objects to be nonspecific (Aissen, 2003).

In addition to confirming known grammaticalization patterns of definiteness, we can mine the highly-weighted features for new hypotheses: e.g., in figs. 3 and 4, the model thinks that objects of "from" are especially likely to be Specific, and that NPs with comparative adjectives (`JJR`) are especially likely to be nonspecific (fig. 3). From fig. 3, we also know that **Num. of dependents, dependent's POS: `1,PRP$`** has a higher weight than, say, **Num. of dependents, dependent's POS: `2,PRP$`**. This observation suggests a hypothesis that in English the NPs which have possessive pronouns immediately preceding the head are more likely to be specific than the NPs which have intervening words between the possessive pronoun and the head. Similarly, looking at another example in fig. 4, the following two percepts get high weights for the NP *the United States of America* to be Specific: **last dependent's POS: `NNP`** and **first dependent's lemma: `the`**. Since frequency and other factors affect the feature weights learned by the classifier, these differences in weights may or may not reflect an inherent association with Specificity. Whether these are general trends, or just an artifact of the sentences that happened to be in the training data and our statistical learning procedure, will require further investigation, ideally with additional datasets and more rigorous hypothesis testing.

Finally, we can remove features to test their impact on predictive performance. Notably, in experiments ablating features indicating articles—the most obvious exponents of definiteness in English—we see a decrease in performance, but not a drastic one. This suggests that the expression of communicative functions of definiteness is in fact much richer than morphological definiteness.

Errors. Several labels are unattested or virtually unattested in the training data, so the models unsurprisingly fail to predict them correctly at test time. **Same_Head** and **Different_Head**, though both common, are confused quite frequently. Whether the previous coreferent mention has the same or different head is a simple distinction for humans; low model accuracy is likely due to errors propagated from coreference resolution. This problem is so frequent that merging these two categories and retraining the random forest model improves Exact Match accuracy by 8% absolute and Soft Match accuracy by 5% absolute.

Percepts			
+Specific		−Specific	
First dependent's POS	PRP$	**First dependent's lemma**	a
Head's left neighbor's POS	PRP$	**Last dependent's lemma**	a
Last dependent's lemma	you	**Num. of dependents, dependent's lemma**	1,a
Num. of dependents, dependent's lemma	1,you	**Head's left neighbor's POS**	JJR
Num. of dependents, dependent's POS	1,PRP$	**Last dependent's POS**	JJR
Governor's right neighbor's POS	PRP$	**Num. of dependents, dependent's lemma**	2,a
Last dependent's POS	NNP	**First dependent's lemma**	new
Last dependent's POS	PRP$	**Last dependent's lemma**	new
First dependent's lemma	the	**Num. of dependents, dependent's POS**	2,JJR
Governor's lemma	from	**Governor's left neighbor's POS**	VB

Figure 3: Percepts receiving highest positive weights in association with values of the Specific attribute.

Example	Relevant percepts from fig. 3	CFD annotation
This is just for *the United States of America*.	**Last dependent's POS: NNP** **First dependent's lemma: the**	Unique_Larger_Situation
We were driving from *our home in Nashville* to a little farm we have 50 miles east of Nashville — driving ourselves.	**First dependent's POS: PRP$** **Head's left neighbor's POS: PRP$** **Governor's right neighbor's POS: PRP$** **Governor's lemma: from**	Bridging_Restrictive_Modifier

Figure 4: Sentences from our corpus illustrating percepts fired for gold NPs and their CFD annotations.

Another common confusion is between the highly frequent category **Unique_Larger_Situation** and the rarer category **Unique_Hearer_New**; the latter is supposed to occur only for the first occurrence of a proper name referring to a entity that is not already part of the knowledge of the larger community. In other words, this distinction requires world knowledge about well-known entities, which could perhaps be mined from the Web or other sources.

7 Related Work

Because semantic/pragmatic analysis of referring expressions is important for many NLP tasks, a computational model of the communicative functions of definiteness has the potential to leverage diverse lexical and grammatical cues to facilitate deeper inferences about the meaning of linguistic input. We have used a coreference resolution system to extract features for modeling definiteness, but an alternative would be to predict definiteness functions as input to (or jointly with) the coreference task. Applications such as information extraction and dialogue processing could be expected to benefit not only from coreference information, but also from some of the semantic distinctions made in our framework, including specificity and genericity.

Better computational processing of definiteness in different languages stands to help machine translation systems. It has been noted that machine translation systems face problems when the source and the target language use different grammatical strategies to express the same information (Stymne, 2009; Tsvetkov et al., 2013). Previous work on machine translation has attempted to deal with this in terms of either (a) preprocessing the source language to make it look more like the target language (Collins et al., 2005; Habash, 2007; Nießen and Ney, 2000; Stymne, 2009, *inter alia*); or (b) post-processing the machine translation output to match the target language, (e.g., Popović et al., 2006). Attempts have also been made to use syntax on the source and/or the target sides to capture the syntactic differences between languages (Liu et al., 2006; Yamada and Knight, 2002; Zhang et al., 2007). Automated prediction of (in)definite articles has been found beneficial in a variety of applications, including postediting of MT output (Knight and Chander, 1994), text generation (Elhadad, 1993; Minnen et al., 2000), and identification and correction of ESL errors (Han et al., 2006; Rozovskaya and Roth, 2010). More recently, Tsvetkov et al. (2013) trained a classifier to predict where English articles might plausibly be added or removed in a phrase, and used this classifier to improve the quality of statistical machine translation.

While definiteness morpheme prediction has been thoroughly studied in computational linguistics,

studies on additional, more complex aspects of definiteness are limited. Reiter and Frank (2010) exploit linguistically-motivated features in a supervised approach to distinguish between generic and specific NPs. Hendrickx et al. (2011) investigated the extent to which a coreference resolution system can resolve the bridging relations. Also in the context of coreference resolution, Ng and Cardie (2002) and Kong et al. (2010) have examined anaphoricity detection. To the best of our knowledge, no studies have been conducted on automatic prediction of semantic and pragmatic communicative functions of definiteness more broadly.

Our work is related to research in linguistics on the modeling of syntactic constructions such as dative shift and the expression of possession with "of" or "'s". Bresnan and Ford (2010) used logistic regression with semantic features to predict syntactic constructions. Although we are doing the opposite (using syntactic features to predict semantic categories), we share the assumption that reductionist approaches (as mentioned earlier) are not able to capture all the nuances of a linguistic phenomenon. Following Hopper and Traugott (2003) we observe that grammaticalization is accompanied by *function drift*, resulting in multiple communicative functions for each grammatical construction. Other attempts have also been made to capture, using classifiers, (propositional as well as non propositional) aspects of meaning that have been grammaticalized: see, for instance, Reichart and Rappoport (2010) for tense sense disambiguation, Prabhakaran et al. (2012) for modality tagging, and Srikumar and Roth (2013) for semantics expressed by prepositions.

8 Conclusion

We have presented a data-driven approach to modeling the relationship between universal communicative functions associated with (in)definiteness and their lexical/grammatical realization in a particular language. Our feature-rich classifiers can give insights into this relationship as well as predict communicative functions for the benefit of NLP systems. Exploiting the higher-level semantic attributes, our log-linear classifier compares favorably to the random forest classifier in Soft Match accuracy. Further improvements to the classifier may come from additional features or better preprocessing. This work has focused on English, but in future work we plan to build similar models for other languages—including languages without articles, under the hypothesis that such languages will rely on other, subtler devices to encode many of the functions of definiteness.

Acknowledgments

This work was sponsored by the U. S. Army Research Laboratory and the U. S. Army Research Office under contract/grant number W911NF-10-1-0533. We thank the reviewers for their useful comments.

References

Barbara Abbott. 2006. Definite and indefinite. In Keith Brown, editor, *Encyclopedia of Language and Linguistics*, pages 3–392. Elsevier.

Judith Aissen. 2003. Differential object marking: iconicity vs. economy. *Natural Language & Linguistic Theory*, 21(3):435–483.

Archna Bhatia, Mandy Simons, Lori Levin, Yulia Tsvetkov, Chris Dyer, and Jordan Bender. 2014. A unified annotation scheme for the semantic/pragmatic components of definiteness. In *Proc. of LREC*. Reykjavík, Iceland.

Betty Birner and Gregory Ward. 1994. Uniqueness, familiarity and the definite article in English. In *Proc. of the Twentieth Annual Meeting of the Berkeley Linguistics Society*, pages 93–102.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Joan Bresnan and Marilyn Ford. 2010. Predicting syntax: Processing dative constructions in American and Australian varieties of English. *Language*, 86(1):168–213.

Victor Chahuneau, Eva Schlinger, Noah A. Smith, and Chris Dyer. 2013. Translating into morphologically rich languages with synthetic phrases. In *Proc. of EMNLP*, pages 1677–1687. Seattle, Washington, USA.

Chao Chen, Andy Liaw, and Leo Breiman. 2004. Using random forest to learn imbalanced data. *University of California, Berkeley*.

Ping Chen. 2004. Identifiability and definiteness in Chinese. *Linguistics*, 42:1129–1184.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proc. of ACL*, pages 531–540. Ann Arbor, Michigan.

Cleo Condoravdi. 1992. Strong and weak novelty and familiarity. In *Proc. of SALT II*, pages 17–37.

William Croft. 2003. *Typology and Universals*. Cambridge University Press.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Michael Elhadad. 1993. Generating argumentative judgment determiners. In *Proc. of AAAI*, pages 344–349.

Gareth Evans. 1977. Pronouns, quantifiers and relative clauses. *Canadian Journal of Philosophy*, 7(3):46.

Gareth Evans. 1980. Pronouns. *Linguistic Inquiry*, 11.

Jeanette K. Gundel, Nancy Hedberg, and Ron Zacharski. 1988. The generation and interpretation of demonstrative expressions. In *Proc. of XIIth International Conference on Computational Linguistics*, pages 216–221.

Jeanette K. Gundel, Nancy Hedberg, and Ron Zacharski. 1993. Cognitive status and the form of referring expressions in discourse. *Language*, 69:274–307.

Nizar Habash. 2007. Syntactic preprocessing for statistical machine translation. In *MT Summit XI*, pages 215–222. Copenhagen.

Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in english article usage by non-native speakers. *Natural Language Engineering*, 12:115–129.

Irene Heim. 1990. E-type pronouns and donkey anaphora. *Linguistics and Philosophy*, 13:137–177.

Iris Hendrickx, Orphée De Clercq, and Véronique Hoste. 2011. Analysis and reference resolution of bridge anaphora across different text genres. In Iris Hendrickx, Sobha Lalitha Devi, Antonio Horta Branco, and Ruslan Mitkov, editors, *DAARC*, volume 7099 of *Lecture Notes in Computer Science*, pages 1–11. Springer.

Paul J. Hopper and Elizabeth Closs Traugott. 2003. *Grammaticalization*. Cambridge University Press.

Nirit Kadmon. 1987. *On unique and non-unique reference and asymmetric quantification*. Ph.D. thesis, University of Massachusetts.

Nirit Kadmon. 1990. Uniqueness. *Linguistics and Philosophy*, 13:273–324.

Kevin Knight and Ishwar Chander. 1994. Automated postediting of documents. In *Proc. of the National Conference on Artificial Intelligence*, pages 779–779. Seattle, WA.

Erwin Ronald Komen. 2013. *Finding focus: a study of the historical development of focus in English*. LOT, Utrecht.

Fang Kong, Guodong Zhou, Longhua Qian, and Qiaoming Zhu. 2010. Dependency-driven anaphoricity determination for coreference resolution. In *Proc. of COLING*, pages 599–607. Beijing, China.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. of COLING/ACL*, pages 609–616. Sydney, Australia.

Christopher Lyons. 1999. *Definiteness*. Cambridge University Press.

Guido Minnen, Francis Bond, and Ann Copestake. 2000. Memory-based learning for article generation. In *Proc. of*

the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning, pages 43–48.

Vincent Ng and Claire Cardie. 2002. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *Proc. of COLING*. Taipei, Taiwan.

Sonja Nießen and Hermann Ney. 2000. Improving SMT quality with morpho-syntactic analysis. In *Proc. of COLING*, pages 1081–1085.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, M. Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Claudia Perlich, Foster Provost, and Jeffrey S. Simonoff. 2003. Tree induction vs. logistic regression: a learning-curve analysis. *Journal of Machine Learning Research*, 4:211–255.

Maja Popović, Daniel Stein, and Hermann Ney. 2006. Statistical machine translation of German compound words. In *Advances in Natural Language Processing*, pages 616–624. Springer.

Vinodkumar Prabhakaran, Michael Bloodgood, Mona Diab, Bonnie Dorr, Lori Levin, Christine D. Piatko, Owen Rambow, and Benjamin Van Durme. 2012. Statistical modality tagging from rule-based annotations and crowdsourcing. In *Proc. of the Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics*, ExProM '12, pages 57–64.

Ellen F. Prince. 1992. The ZPG letter: Subjects, definiteness and information status. In S. Thompson and W. Mann, editors, *Discourse description: diverse analyses of a fund raising text*, pages 295–325. John Benjamins.

Marta Recasens, Marie-Catherine de Marneffe, and Christopher Potts. 2013. The life and death of discourse entities: identifying singleton mentions. In *Proc. of NAACL-HLT*, pages 627–633. Atlanta, Georgia, USA.

Roi Reichart and Ari Rappoport. 2010. Tense sense disambiguation: A new syntactic polysemy task. In *Proc. of EMNLP*, EMNLP '10, pages 325–334.

Nils Reiter and Anette Frank. 2010. Identifying generic noun phrases. In *Proc. of ACL*, pages 40–49. Uppsala, Sweden.

Craig Roberts. 2003. Uniqueness in definite noun phrases. *Linguistics and Philosophy*, 26:287–350.

Alla Rozovskaya and Dan Roth. 2010. Training paradigms for correcting errors in grammar and usage. In *Proc. of NAACL-HLT*, pages 154–162.

Bertrand Russell. 1905. On denoting. *Mind, New Series*, 14:479–493.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proc. of ACL*, pages 455–465. Sofia, Bulgaria.

Vivek Srikumar and Dan Roth. 2013. An inventory of preposition relations. *CoRR*, abs/1305.5785.

Sara Stymne. 2009. Definite noun phrases in statistical machine translation into Danish. In *Proc. of Workshop on Extracting and Using Constructions in NLP*, pages 4–9.

Yulia Tsvetkov, Chris Dyer, Lori Levi, and Archna Bhatia. 2013. Generating English determiners in phrase-based translation with synthetic translation options. In *Proc. of WMT*.

Kenji Yamada and Kevin Knight. 2002. A decoder for syntax-based statistical MT. In *Proc. of ACL*, pages 303–310. Philadelphia, Pennsylvania, USA.

Yuqi Zhang, Richard Zens, and Hermann Ney. 2007. Improved chunk-level reordering for statistical machine translation. In *IWSLT 2007: International Workshop on Spoken Language Translation*, pages 21–28.

Argument structure of adverbial derivatives in Russian

Igor Boguslavsky
Institute for Information Transmission Problems, Russian Academy of Sciences /
19, B.Karetnyj, GSP-4, Moscow, Russia
Universidad Politécnica de Madrid / Facultad de Informática, Campus de Montegancedo,
28660 Boadilla del Monte, Madrid, Spain

Abstract

Adverbial derivatives (AdvD) of nouns of the type *v jarosti* 'in a rage'*, s naslaždeniem* 'with pleasure'*, pod predlogom* 'under the pretext of' etc. often inherit the arguments (actants) of the noun they are derived from. However, as a rule, in case of AdvDs these arguments are realized in a way very different from the nouns. The main linguistic findings of the paper consist in the set of positions the arguments may take with respect to AdvD. In a general case, a actant slot of an AdvD can be either (a) blocked, or (b) filled by a dependent of the AdvD itself (e.g. *pod predlogom bolezni* 'under the pretext of illness', *v dokazatel'stvo svoej nevinovnosti* 'as a proof of his innocence'), or (c) filled by the dominating verb (*po privyčke prosnulsja rano* 'woke up early out of habit', *slushal pesnju s naslaždeniem* 'listened to the song with relish'), or (d) filled somewhere within the clause organized by the dominating verb; in this case the AdvD argument may be identified based on (d1) its syntactic position (*po privyčke* 'by habit'), or (d2) its semantic role with respect to its mother element (*v podarok* 'as a present'), or (d3) its communicative function (*v bol'šinstve* 'mostly'). A notation is proposed that permits to present the argument structure of AdvDs in a compact way.

1 Introduction

This paper is not about computation, it is about linguistics. It does not describe any electronic resource. It is not inspired by weaknesses of NLP applications that need to be fixed. We investigate certain heavily understudied and even largely unnoticed linguistic phenomena that deserve scientific study independently of whether their neglect causes serious errors in today's NLP applications or not. However, on the other hand, taking these phenomena into account is definitely useful for applications, such as semantic parsing, question answering, recognizing textual entailments, information extraction (e.g. Meyers et al. 1998), machine reading, machine translation, etc. Indeed, semantic parsers should represent the content of the text by means of elementary propositions independently of the syntactic status of the main predicate in these propositions, be it a verb, a noun, or an adverbial. They should be able to understand that such expressions as *I believe (that) he is wrong – My opinion is (that) he is wrong) – In my opinion <to my mind>, he is wrong* are different NL realizations of the same proposition. Question answering systems should be able to obtain an answer to the question *What habits does John have?* from the sentence *John woke up early out of habit*, although the argument frame of the noun *habit* does not cover this type of construction (it is the argument frame of the adverbial derivative *out of habit* that does). Similarly, textual entailment recognition systems should understand that *John woke up early out of habit* entails *John has a habit of waking up early*, which again requires correlating argument frames of three different expressions: the noun *habit,* the "support verb + noun" combination *to have a habit* and the adverbial *out of habit* .

Syntactic derivation is one of the most direct manifestations of the systemic character of the lexicon. As is well-known, language is capable of representing the same meaning (or several very close meanings) by means of words belonging to different grammatical classes. It is often possible to replace words of a certain grammatical category with those of another grammatical category without significant modification of their lexical meaning. For example, the concept 'believe' can be realised by means of a verb (*to believe*) or a noun (*opinion*) or an adverbial phrase (*in my opinion, to my mind*). This is one of the important ideas of Éléments de syntaxe structurale de Tesnière (1959). According to Tesnière, the ability to transfer one category to another at will in fluid speech is the primary tool that

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1071–1080, Dublin, Ireland, August 23-29 2014.

makes truly productive speech possible. This mechanism is an integral part of the linguistic capacity of humans and deserves in-depth study.

Lexical resources available to date are not sufficient for that. First, resources such as WordNet do not establish synonymy relations across category boundaries, and will not recognize these expressions as synonymous. Second, the task does not boil down to relating such expressions to the same concept. To reconstruct the proposition, one also needs to find the arguments of all the predicates and identify their roles. The latter task, also known as Semantic Role Labeling (SRL), is fairly well studied for the arguments of the verbs (cf. CoNLL-2004 and CoNLL-2005 shared tasks on semantic role labeling, Computational Linguistics Special Issue on Semantic Role Labeling, 2008). Much less is done in SRL of nouns and adjectives (Gerber 2011, Macleod 1997, 1998). Sometimes, adjectives and prepositions are included in (verbal and nominal) frames in FrameNet. However, we are not aware of any attempt to investigate arguments of adverbials. This category of words is largely understudied. It is not even represented in WordNet. In the introductory paper to the Special Issue on Semantic Role Labeling, the SRL task for adverbials is not even mentioned (Marquez et al. 2008).

Yet, adverbial derivatives are no less entitled to have arguments than the predicates they are derived from. If we want to find and identify the arguments of the verb *to cause* in (1), we would want to do the same in (2), where this concept is represented by means of the adverbial *due to:*

(1) *The minister's interview caused a dramatic fall of the market.*

(2) *The market fell dramatically due to the minister's interview.*

However, the problem is that, in a general case, it is more difficult to find these arguments in the sentence than it is for prototypical verbal or nominal predicates. The positions of these arguments in may differ greatly from the positions of «classical» arguments.

The goal of this research is to investigate these non-classical arguments with a view to their adequate representation in the dictionary and their automatic detection in the text. We intend to show (a) that the arguments of adverbials need to be found and identified, however non-trivial this task may be, (b) what their different types are and (c) how the argument structure of adverbial derivatives can be represented in the dictionary.

In this study, we will restrict ourselves to adverbial syntactic derivatives (AdvD) of Russian nouns and verbs. We will call a syntactic derivative of word *L* such a word, or phrase, *L´* that has the same (or very close) meaning as *L*, but belongs to a different syntactic category and hence displays a different behavior. We will denote the word *L* as the basic word, or keyword, of the derivation. *L´* may be a nominal derivative, or nominalization (*to construct - construction, to believe – opinion*), a verbal derivative (*revolution - revolutionize*), an adjectival derivative (*government – governmental*), or an adverbial derivative (*speed – at the speed of, cause – due to/because of*).

The plan of our presentation will be as follows. In Section 2 we will characterize briefly some properties of AdvD in Russian, then in Section 3 we will review related work on adverbial derivatives within the framework of the Meaning – Text theory. Section 4 will present different types of argument structures of adverbial derivatives. How these structures can be represented in the dictionary will be shown in Section 5. We will conclude in Section 6.

2 Adverbial derivatives in Russian

We will discuss two properties of AdvDs – their grammatical status and their semantic load. From the point of view of the grammatical status, there are two types of AdvD in Russian – grammatical and lexical ones. They will be explained in Sections 2.1 and 2.2. The semantic load of AdvD will be commented upon in Section 2.3.

2.1 Grammatical AdvDs (verbal adverbs)

Russian has a regular morphological way of constructing AdvDs of verbs – verbal adverbs (deeprichastija) that can be derived of virtually any verb. They serve to express a secondary predication attached to the main one.

(3) *On sprosil eto, gljadja ej v glaza.*

'He asked that, looking into her eyes'

Russian verbal adverbs are similar to participial constructions in adverbial usage (or gerunds) existing in a variety of languages. However, they also exhibit significant differences. An important peculiarity of the argument behaviour of Russian verbal adverbs is that their subject cannot be

expressed in the surface structure and should be co-referential with the subject of the main verb. Their other arguments do not have any characteristic properties and are attached to them just as they are attached to the finite form of the verb:
(4a) *Petr pokupaet odeždu v modnyx magazinax za ogromnye den'gi* 'Peter buys clothes in fashionable shops for a lot of money'.
(4b) *Pokupaja odeždu v modnyx magazinax za ogromnye den'gi, Petr večno po uši v dolgax* 'buying clothes in fashionable shops for a lot of money, Peter is always in debt up to his neck'.

Many languages have absolute constructions, absent in Russian, which allow the subject to be attached to the participle and to be non-coreferential with the subject of the main verb:
(5) *His wife buying clothes in fashionable shops, Peter is always in debt up to his neck.*
(6) Spanish: *Habiendo pasado más de una hora, las piernas comenzaron a flaquear* 'more than an hour having passed, his legs began to fail'.

Verbal adverbs may be active, as in (3), or passive, as in (7):
(7) *Buduči sorvannym, muxomor prodolžaet rasti* 'being plucked, the amanita continues to grow'.

It is to be noted that the implicit subject of the passive verbal adverb *buduči sorvannym* 'being plucked' is the second argument of the active form *sorvat'* 'pluck' and, according to the general rule, is co-referential with the subject of the main verb *muxomor* "amanita".

2.2 Lexical AdvDs

Besides verbal adverbs derived by means of inflection, there is a large number of AdvDs that are expressed by adverbs (*good – well, systematic – systematically, can - possibly*), prepositions (*cause – due to/because of*) or prepositional phrases (*love – with love*). The latter case is the most important, since a large number of AdvDs is formed in this way. It is to be noted that different AdvDs are formed with different prepositions. Semantically, lexical AdvDs are in many cases equivalent to verbal adverbs. Some examples: *otčajanie* 'dispair' – *v otčajanii* '(being) in dispair', *interes* 'interest' – *s interesom* 'with interest, feeling interest', *odežda* 'clothes' – *v odežde* 'being dressed', *bodrstvovat'* 'be awake' – *najavu* '(being) awake', *obed* 'dinner' – *za obedom* 'at dinner', *zaščita* 'protection' – *pod zaščitoj* 'under protection, being protected', *pomošč* 'help' – *s pomoščju* or *pri pomošči* 'with the help of, being helped by'.

2.3 Pure AdvDs vs. semantically loaded AdvDs

One has to distinguish between two types of AdvD: "pure" derivatives, which do not contain any additional meaning components absent in the meaning of the basic predicate, and semantically enriched derivatives, for which the reverse is true. As an example of the latter, let us consider the phrase *pod imenem X* 'under the name of X' as represented in
(8) *Napoleon exal pod imenem gerzoga Vičentskogo, to est' Kolenkura* 'Napoleon was travelling under the name of duke of Vicenza, that is of Colencour'.

The meaning of this sentence contains a component of replacing the true name with another one. *Pod imenem X* 'under the name of X' does not mean that the person in question has the name of X, but rather that this person or somebody else wants other people to refer to him/her as X while the speaker knows, believes or admits that this is not the true name. Obviously, the noun *imja* 'name' has no such component (as opposed to *pseudonym* or *nickname*). It cannot even be ascribed to the preposition *pod* 'under', either, since in (8) this preposition has obviously the same meaning as in phrases *pod nazvaniem <zagolovkom, rubrikoj>* 'under the title <heading>' to which the component of concealment is completely alien. Phrases like *pod imenem* (or its English counterpart *under the name*) that are to a certain extent idiomatic are lexical units in their own right and have their own entries in the dictionary.

As for "pure", non-idiomatic PP AdvDs, they hardly qualify for separate lexical units. However, irrespective of whether an AdvD is idiomatic or not, it should be supplied with the information about its arguments: if a sentence contains the phrase *at request*, e.g. *I called Mary at the request of my father,* we should be able to answer the question "Who asked whom to do what?" In this example, we are entitled to infer that speaker's father asked him/her to call Mary.

Note: strictly speaking, the content of father's request need not necessarily be "Make a call to Mary". He could have asked his son/daughter to invite Mary for dinner. However, the phrase *I called Mary at the request of my father* is still appropriate provided the act of inviting Mary contains calling

her as its essential part.

In order to be able to make such an inference, one needs to represent the argument structure of AdvDs fully and unambiguously and relate it to the argument structure of the basic word.

As an example of how the correlation between argument structures of different words can be established, we can recall the description of converse terms in the theory of Lexical Functions within the Meaning – Text approach (Melčuk et al. 1984a, 1984b, 1988, 1992). Conversives (the input and the output of the Lexical Function Conv) are a pair of words that denote the same situation but differ in the way their arguments are ordered, e.g. *buy – sell*. Like any verb, *buy* and *sell* are supplied with subcategorization frames (aka government patterns in the Meaning – Text approach) that list all their arguments and their means of expression. On the other hand, their being conversives implies that their lexical functional description should indicate the correlation between their argument structures. Namely, the first argument of *sell* ("who sells?") corresponds to the third argument of *buy* ("from whom buys?"), the third argument of *sell* ("to whom?") – to the first argument of *buy* ("who buys?"), while the second and the fourth arguments ("what?" and "for how much?") occupy the same positions within government patterns of both verbs. This correlation is rendered by the numerical index attached to the Conv symbol: $Conv_{3214}$(*sell*) = *buy*: the *j*-th position in the index is occupied by *i* if the *j*-th argument of the output corresponds to the *i*-th actant of the input.

For AdvDs the problem of the correlation of their argument structure with that of the basic predicate is particularly acute. While *sell* and *buy* are rightful lexical units entitled to have their own government patterns, adverbial collocations of the type *v jarosti* 'in anger', *po privyčke* 'by habit', *s akcentom* 'with an accent' or *pod predlogom* 'under the pretext of' are not usually treated as separate lexical units. It is assumed that all necessary information about their meaning and use should be formulated in the lexical entry of the noun. To what extent does an AdvD inherit the argument structure of the noun? If not in full, how should its argument structure be described in the dictionary?

Before answering this question, we will recall how syntactic derivatives, and AdvD in particular, are treated in the theory of lexical functions.

3 Syntactic derivatives in the theory of lexical functions

Two types of syntactic derivation are distinguished: "zero" and "actant" derivation. Zero syntactic derivatives (S_0, V_0, A_0 и Adv_0) have the same meaning as the keyword but belong to a different part of speech: S_0(*investigate*)=*investigation,* V_0(*investigation*)=*investigate,* A_0 (*government*)=*governmental,* Adv_0 (*good*)=*well*. Actant derivatives (S_i, A_i and Adv_i) are oriented towards one of the actants of the keyword in the following sense.

S_i is a standard name of the *i*-th actant of the keyword (S_1(*teach*) = *teacher*, S_2(*teach*) = (*subject*) *matter* [in high school], S_3(*teach*) = *pupil*).

A_i also has a bearing to the *i*-th actant, but in the adjectival syntactic status. This means that its typical syntactic function is to modify a noun that fills the *i*-th valence slot of the keyword. A grammatical way of expressing A_i is participles. A_1 is equivalent to an active participle, and A_2, to a passive participle. For example, adjectival derivatives of the verb *to control* are either an active participle *controlling* (A_1) or a prepositional phrase *under control* (A_2); cf. *controlling organizations* ('organizations that control something') – *operations under control* ('operations that are being controlled by someone').

Things are more complicated with adverbial actant derivatives (Adv_i). This function is defined as follows:

«Adv_i – determining property of the action by the *i*-th … actant of *L* according to its role in the situation denoted by *L*. Adv_1 is roughly equivalent to an active verbal adverb ('while L-ing') and Adv_2, to a passive verbal adverb ('while being L-ed').

Adv_2(*bombard*) = *under bombardment*

Adv_1(*speed*) = *at [a speed of...]*» (Mel'čuk 1996: 55).

As this definition shows, the only link between the actantial structures of the keyword and the adverbial derivative of the Adv_i type is the *i*-th actant of the keyword. Although it is not stated explicitly, one can presume that the *i*-th actant's position in the sentence is the position of the subject of the verb to which Adv_i is attached. In (9) the first actant of *anger* is obviously *Mary,* the first actant of the verb *reject,* and not *John* or anybody else.

(9) *Mary rejected John's proposal with anger.*

This is understandable, since lexical function Adv_i is intended to model the behaviour of verbal adverbs and, as mentioned above, they normally correlate strongly with the subject (first actant) of the main predicate (except for the absolute constructions). However, lexical function Adv_i provides no information as to the position of other actants of the keyword. The next section will show that this information is essential for text understanding and that different AdvDs significantly differ in this respect.

4 Argument structure of adverbial derivatives

If we compare adverbial derivation with other types of syntactic derivation, we will encounter an important difference. Argument structure of such derivatives as $Conv_{ij}$ or S_i can be easily characterized in terms of the argument structure of the keyword. When we pass from the keyword to such a derivative, we may find that an actant either stays in its initial position (*teach mathematics – teacher of mathematics*), or changes its number (*the verb dominates the preposition – the preposition depends on the verb*), or gets blocked altogether (*drive home - *driver home*). However, the syntactic position of the actant can only change in a very limited way. If a valence slot of the keyword is expressible in the sentence with its $Conv_{ij}$ or S_i at all, the actant should either be attached to the derivative directly, or through a copula or other lexical functional verb (*Peter teaches mathematics – Peter is a teacher of mathematics*) or by means of the apposition (*Peter, a teacher of mathematics*).

The matters stand differently with AdvDs. Their actant properties are much more diverse than those of $Conv_{ij}$ or S_i, or even of verbal adverbs. In some cases, the position of their actants in the sentence cannot be characterized in purely syntactic terms. In a general case, a valence slot of an AdvD can be either blocked, or:

- filled by a dependent of the AdvD itself;
- filled by the dominating verb;
- filled somewhere within the clause organized by the dominating verb; in this case the AdvD actant may be identified based on:
    - its syntactic position;
    - its semantic role;
    - its communicative function.

We will illustrate all these possibilities below.

4.1 Valence slots filled by a dependent of AdvD

In the canonical case, AdvD inherits most of the governing properties of the keyword.
(10a) *skorost' 800 km/čas* 'a speed of 800 km per hour',
(10b) *Samolet letel so skorostju 800 km/čas* 'the aircraft was flying at a speed of 800 km per hour',
(11a) *sovet Ivana* 'Ivan's advice',
(11b) *po sovetu Ivana* 'at Ivan's advice',
(12a) *Eto podarok ot Viktora* lit. 'this is a present from Victor',
(12b) *Ja polučil eto v podarok ot brata* lit. 'I got it as a present from my brother'.

In some cases, governing properties of AdvD are different from those of the keyword. Let us consider the pair *predlog* 'pretext' – *pod predlogom* 'on/under the pretext of' that manifests an interesting correlation of actant properties. The noun *predlog* 'pretext' has three valence slots: *P is a pretext for X for doing Q* = 'wishing to do Q, which violates norms of ethics, or politeness, X uses situation P to do Q; he thinks that P justifies Q' (Boguslavskaya 2003). When *predlog* is used without the preposition *pod,* it can attach actant Q (= the action carried out) but not P (= false motive). The latter can only be expressed outside the phrase containing *predlog*:
(13a) *Golovnaja bol'* [P] *– xorošij predlog, čtoby ostat'sja doma* [Q] 'headache [P] is a good pretext for staying at home [Q]'.
(13b) **predlog golovnoj boli* [P] 'the pretext of the headache [P]'

AdvD *pod predlogom* has opposite governing properties. Actant P (= false motive) can now be a dependent of AdvD while actant Q (= the action) loses this property and moves to the position of the dominating word:
(13c) *Ona ostalas' doma* [Q] *pod predlogom golovnoj boli* [P] 'she stayed at home [Q] on the pretext of the headache [P]'.

4.2 Valence slots filled by the dominating verb

Adverbial derivatives of many predicates which have a propositional valence slot fill it by means of the dominating verb. One example is (13c) above. In the following examples, the actant at issue is underlined in both the sentence with the basic predicate, and in the sentence with the AdvD.
(14a) *Ljusja dokazala polnuju sdaču svoix pozicij tem, čto pocelovala Marata v nos* 'Ljusja proved complete surrender by kissing Marat on the nose'.
(14b) *V konce koncov sama Ljusja priznala grubost' svoego zamečanija i v dokazatel'stvo polnoj sdači svoix pozicij pocelovala Marata v nos* 'after all, Ljusja herself acknowledged that her remark had been rude, and as a proof of complete surrender kissed Marat on the nose' (AdvD *v dokazatel'stvo* 'as a proof').
(15a) *On otvetil mnogoznačitel'nym myčaniem* 'he responded with a significant mumble'.
(15b) *V otvet on čto-to mnogoznačitel'no promyčal* 'in response he mumbled something in a significant manner' (AdvD *v otvet* 'in response').
(16a) *Ja sčitaju, čto ždat' bol'še nečego* 'I think there is nothing more to wait for'.
(16b) *Po-moemu, ždat' bol'še nečego* 'in my opinion, there is nothing more to wait for' (AdvD *po-moemy* 'in my opinion').

4.3 Valency slots filled by dependents of the dominating verb

If a valency slot of an AdvD is filled by a dependent of the dominating verb, the question arises as to how to specify its position among other dependents of the verb. We will show that this position can be identified based on the syntactic function (4.3.1), semantic role (4.3.2) or communicative function (4.3.3).

4.3.1 Syntactic function

As mentioned in section 2, in the prototypical case of adverbial derivation, that of verbal adverbs, one of the actants of the keyword (the first or the second) is necessarily co-referential with the first actant (subject) of the dominating verb. Since this actant is not expressible as a dependent of the AdvD, the subject of the dominating verb is its only manifestation in the sentence. In this sense, we can say that the valence slot is filled by the subject of the dominating verb. If it is the first actant of the keyword that is co-referential with the subject, we are dealing with the active verbal adverb (Adv_1, in Mel'čuk's terminology). If it is the second actant, the verbal adverb (Adv_2) is passive. If the co-reference requirement is not met, sentences with verbal adverbs are usually ungrammatical in Russian. Cf. a textbook example of a wrong use of a verbal adverb **Podjezžaja k stancii, u menja sletela šljapa* 'when approaching the station, my hat fell down'.

As for non-verbal AdvDs, this requirement holds for some of them and not for others. Let us discuss one example: the verb *privyknut'* 'have a habit of'. It has two valencies – 'who has the habit?' and 'what does the habit consist in?'. Its AdvD is *po privyčke* 'by habit'. Although it does not take any syntactic dependents, sentences with this AdvD provide unambiguous information on who has a habit and what it consists in/ hence, both valencies are filled:
(17) *Ivan po privyčke ostavil dver' otkrytoj* 'by habit, Ivan left the door open'
The identity of the first actant of *po privyčke* and the subject of the main verb can be easily demonstrated. Let's take the verbs *zanimat'* 'to borrow' and *odalživat'* 'to lend'. Being conversives, they denote the same situation and sentences (18a) and (18b) are synonymous:
(18a) *Ivan zanjal u soseda 1000 rublej* 'Ivan borrowed 1000 roubles from the neighbour'
(18b) *Sosed odolžil Ivanu 1000 rublej* 'the neighbour lent Ivan 1000 roubles'.

If AdvD *po privyčke* 'by habit' is introduced in (18a) and (18b) in the same position, the sentences will no longer be synonymous. (19a) refers to the habit of Ivan while (19b) – to the habit of the neighbour.
(19a) *Po privyčke Ivan zanjal u soseda 1000 rublej* 'by habit Ivan borrowed 1000 roubles from the neighbour'
(19b) *Po privyčke sosed odolžil Ivanu 1000 rublej* 'by habit the neighbour lent Ivan 1000 roubles'.

4.3.2 Semantic role

Another type of constraint is manifested by AdvDs *v podarok* 'as a present', *v dar* 'as a gift', *v nagradu* 'in reward'. Nouns of the *present / gift / reward* type have three valence slots: the agent of

presenting something (X), the theme (Y) and the recipient (Z). The AdvDs co-occur with a large set of verbs concentrated around the meaning of 'transfer': *polučat'* 'receive', *prinimat'* 'accept', *trebovat'* 'demand', *prosit'* 'request'; *prinosit'* 'bring (on foot)', *privozit'* 'bring (by transport)', *dostavljat'* 'deliver', *posylat'* 'send', *otpravljat'* 'dispatch', *prednaznačat'* 'intend for', *žalovat'* 'grant', *podnosit'* 'offer', *predlagat'* 'offer', *peredavat'* 'pass (to)', *vručat'* 'hand over', *davat'* 'give', *otdavat'* 'give back', etc. It is impossible to associate the subject of the main verb with any single actant of AdvD, since each of the three actants can perform the role of the subject:
(20a) *Otec* (X) *privez dočeri v podarok ožerelje* 'Father (X) brought a necklace as a present to his daughter'.
(20b) *Maria* (Z) *prinjala ožerelje v podarok* 'Mary (Z) accepted the necklace as a present'.
(20c) *Ožerelje* (Y) *dostalos' ej v podarok ot babuški* 'the necklace (Y) came to her as a present of her grandmother'.

It is not syntactic constraints that regulate the position of the actants of these AdvD with respect to the main verb but semantic ones. The correlation between the valence slots of AdvD and the main verb can be formulated IN TERMS OF SEMANTIC ROLES as follows: if a valence slot of AdvD which corresponds to semantic role R (Agent, Theme, Recipient) is instantiated, it is either filled by an AdvD dependent (as in *v podarok dočeri* 'as a present to one's daughter', *v podarok ot otca* 'as a present from one's father'), or by a dependent of the main verb which performs the role R with respect to the predicate of transfer within the meaning of the main verb. For example, in (20a-c) the subjects *otec* 'father', *Maria* 'Maria' and *ožerelje* 'necklace' all play different semantic roles with respect to the main verb: the father is the Agent of bringing, Maria is the Recipient of giving ('accept' ≈ 'agree to be given'), and the necklace is the Theme of coming. Accordingly, these words are the Agent, Recipient and Theme of the present, respectively.

It should be stressed that the semantic role of a noun phrase with respect to the dominating verb may be different from its semantic role with respect to an inner predicate of this verb. For example, in
(21a) *Ivan potreboval poltsarstva* 'Ivan demanded half of the kingdom' [= 'demanded that he were given half of the kingdom']
Ivan is the Agent of demanding and at the same time the Recipient of giving. What is important for AdvD of the *v podarok* type is the role of the actant with respect to giving. Therefore, in (21b) Ivan is the Recipient and not the Agent of reward:
(21b) *Ivan potreboval sebe v nagradu poltsarstva* 'Ivan demanded half of the kingdom as a reward'.

4.3.3 Communicative function

Boguslavsky (2005) discussed the argument frames of noun *bol'šinstvo* 'majority, most of' and *men'šinstvo* 'minority'. It was shown that these words have three arguments: the whole, a part of the whole and the property of the part that is shared by most of the elements of the whole. Here we will only be interested in one of these arguments – that of the whole, expressed prototypically by preposition *iz* 'of' as represented in phrases *the majority of cases, most of the students.* In sentences with AdvD *v bol'šinstve* 'mostly, for the most part' this valence slot is filled, as a rule, by the subject of the dominating verb:
(22) *Oni byli arestovany i podverglis' v bol'šinstve svoem ssylke v Gvianu i na Sejšel'skie ostrova* 'they were arrested and mostly exiled to Guiana or Seychelles' [= 'most of them were exiled…']

However, this is not the only possible syntactic role for this actant. In (23) it is the direct object *inostrannye knigi* 'foreign books' that fills the valence slot of the whole:
(23) *Russkie knigi byli sobrany pokojnym mužem knjagini…, inostrannye že – v bol'šinstve vyvezla sama Anna Arkadjevna iz Pariža* lit. 'Russian books were collected by the late husband of the princess…, while foreign books (dir. object) mostly Anna Arkadjevna brought from Paris herself' [= 'most of the foreign books'].

And even this is not all. What is essential here is not the syntactic role of the actant but the communicative organization of the clause. The valence slot of the whole should be filled by the Topic. Since the position of the Topic is most often held by the subject, it is clear why it is the subject that for the most part fills this valence slot. The claim that the valency of the whole of *v bol'šinstve* 'mostly' is Topic-oriented can be confirmed by a minimal pair of sentences below.

Due to the relatively free word order in Russian, the Topic-Focus distinction is rarely marked syntactically or lexically. The same syntactic structure may correspond to different Topic-Focus

articulations. In most cases it is the clause-initial phrase that is the Topic of the sentence[1]. In (24a) and (24b) the syntactic structures are the same but the word order and the Topic-Focus articulation s are different. Therefore, the valency slot of the whole is filled differently:
(24a) *Ženščiny* (Topic) *v bol'šinstve svoem sideli v zale*
lit. 'the women (Topic) in majority were sitting in the hall'
'most of the women were in the hall'
(24b) *V zale sideli* (Topic) *v bol'šinstve svoem ženščiny*
lit. 'in the hall were sitting (Topic) in majority the women'
'most of those in the hall were women'

5 Representation of adverbial derivatives in the lexicon

From the viewpoint of the theory of phraseology developed in Mel'čuk 1995, AdvDs belong to the class of collocations and should be represented in the dictionary within the entries of their nominal component – the keyword of the derivation (Mel'čuk 1995: 184). The entries of the keywords contain all the information on their argument frames. Based on this information, one can represent the argument frame of AdvD in a compact way.

AdvD are to be described in the dictionary entry of the keyword, as a value of the Adv Lexical Function. The argument structure of the derivative is described by means of an index attached to the Adv symbol. We showed above (in 2.3) how the correlation between the arguments of the conversives can be stated by means of the numerical index attached to the symbol of the Conv Lexical Function. We are going to describe AdvDs along similar lines, but the index should be somewhat more elaborated. Namely, the argument index of the Lexical Function Adv is constructed as follows:

- it consists of *n* positions, according to the number of valency slots of the keyword; the 1st position corresponds to the 1st slot of the keyword, the 2nd position corresponds to the 2nd slot etc.
- each position contains information on whether the corresponding valency can be filled if the keyword is represented by its adverbial derivative and, if so, how it should be filled. This information is one of the following:
  - 0, if the slot cannot be filled,
  - i, if the slot is filled as the *i*-th slot of the keyword,
  - G, if the slot is filled by the syntactic governor of the AdvD,
  - i(G), if the slot is filled by a phrase that is the *i*-th actant of the syntactic governor of AdvD or has semantic role *i* with respect to this governor,
  - Topic, if the slot is filled by the Topic of the clause to which belongs AdvD.

Let us show how the properties of AdvDs of different types can be represented using this notation. Each illustration consists of three parts: (a) the keyword and its argument frame, (b) an example containing AdvD, (c) representation of the argument frame of AdvD with a short comment.
(25a) *skorost'* 'speed' («what has the speed?», «the value of the speed»)
(25b) *Avtomobil' mčalsja so skorostju 200 km/čas* 'the car moved at the speed of 200 km/hour'
(25c) *so skorostju* 'at the speed of' = $Adv_{G,2}$ [the 1st argument is the syntactic governor of AdvD ('moved'), and the 2nd is the 2nd argument of the keyword]
(26a) *jarost'* 'rage' («who is in the state of rage?», «what was the cause of this state?»)
(26b) *On v jarosti razorval pis'mo na kločki* 'in a rage, he tore the letter to pieces'.
(26c) *v jarosti* 'in a rage' = $Adv_{1(G),0}$ [the 1st argument is the 1st argument ('he') of the syntactic governor ('tore'), the 2nd argument cannot be realized with AdvD]
(27a) *naslaždenie* 'relish, enjoyment'(«who enjoys? », «what does one enjoy?»)
(27b) *On s naslaždeniem vykuril sigaru* 'he smoked a cigar with relish'.
(27c) *s naslaždeniem* 'with relish' = $Adv_{1(G),G}$ [the 1st argument is the 1st argument ('he') of the syntactic governor ('smoked'), the 2nd argument is the syntactic governor itself. Note the important difference between (26c) and (27c): in case of 'with relish' the main predicate refers to the source of the emotional state: smoking a cigar is what made him feel relish; in (26b) the reason of feeling rage is not specified. This difference is reflected in different indices]
(28a) *somnenije* 'doubt' («who doubts?», «what does one doubt?»)

[1] Of course, this is a simplification, the reality is more complicated, but this is a general rule.

(28b) *Ona vrjad li pridet* ‘she will hardly come’
(28c) *vrjad li* ‘hardly, unlikely’ = $Adv_{0,G}$ [the 1st argument cannot be realized with AdvD, the 2nd is expressed by the syntactic governor]
(29a) *podarok* ‘a present’ (“who gives?”, “what is given?”, “to whom?”)
(29b) *Otec privjoz Marii v podarok ožerelje* ‘Father brought Maria a necklace as a present’.
(29c) *v podarok* ‘as a present’ = $Adv_{G(Agent),G(Theme),G(Recipient)}$ [in (29b) all the three argument slots of AdvD are filled by the corresponding arguments of the predicate of transfer – *prvjoz* ‘brought’]
(30a) *bol'šinstvo* ‘majority’ (“what constitutes the majority?”, “what is the whole?”)
(30b) *V zale sideli v bol’šinstve svoem ženščiny* lit. ‘in the hall were sitting (Topic) the women’
‘most of those in the hall were women’
(30c) *v bol’šinstve* ‘mostly’ = $Adv_{0,Topic}$ [this AdvD is topic-sensitive; in (30b) the Topic is ‘those who were in the hall’, therefore it is this meaning that fill the valency of the whole].

6 Conclusion

The data presented above show that the argument frames of the adverbial derivatives of predicates are much more diverse than it was believed before. The number of the arguments and their roles are motivated by the semantics of the predicate they are derived from, but their syntactic realization is largely different. We showed a variety of syntactic, semantic and communicative positions the arguments of adverbial derivatives may take and how these positions can be described in the dictionary in a compact way. This information is needed in many semantics-related tasks but is not available in any of the existing lexicographic resources. We proposed a way to represent this information in the lexicon in a compact way. Supplied with this information, the lexicon will be able to support the extraction of propositions for a variety of applications[2].

References.

Olga Boguslavskaya. 2003. ПОВОД, ПРЕДЛОГ. In: Новый объяснительный словарь синонимов. Языки славянской культуры. Вып. 3.

Igor Boguslavsky. 2005. Валентности кванторных слов. In: Квантификативный аспект языка. Москва, стр. 139-165.

Computational Linguistics. Special Issue on Semantic Role Labeling, 2008.

Matthew Steven Gerber. 2011. Semantic Role Labeling of Implicit Arguments for Nominal Predicates. A dissertation submitted to Michigan State University n partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science.

Catherine Macleod, Adam Meyers, Ralph Grishman, Leslie Barrett, Ruth Reeves. 1997. *Designing a Dictionary of Derived Nominals. Proceedings of Recent Advances in Natural Language Processing*, Tzigov Chark, Bulgaria, September, 1997.

Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, Ruth Reeves. 1998. *NOMLEX: A Lexicon of Nominalizations*. Proceedings of EURALEX'98, Liege, Belgium, August 1998.

Lluis Marquez, Xavier Carreras, Kenneth C. Litkowski, Suzanne Stevenson. 2008. *Semantic Role Labeling: An Introduction to the Special Issue*. Computational Linguistics. Special Issue on Semantic Role Labeling, 2008.

Igor Mel’čuk et al. 1984a. Мельчук И. А., А. К. Жолковский, Ю. Д. Апресян, и др. 1984. *Толково-комбинаторный словарь современного русского языка. Опыты семантико-синтаксического описания русской лексики*. Wien.

Igor Mel’čuk et al. 1984b. DEC: Dictionnaire explicatif et combinatoire du français contemporain. Recherches lexico-sémantiques I. Les Presses de l’Université de Montréal, Montréal (Québec).

Igor Mel’čuk et al. 1988. DEC: Dictionnaire explicatif et combinatoire du français contemporain. Recherches lexico-sémantiques II. Les Presses de l’Université de Montréal, Montréal (Québec).

Igor Mel’čuk et al. 1992. DEC: Dictionnaire explicatif et combinatoire du français contemporain. Recherches lexico-sémantiques III. Les Presses de l’Université de Montréal, Montréal (Québec).

Igor Mel’čuk. 1995. Phrasemes in Language and Phraseology in Linguistics. In: M. Everaert, E.-J. van der Linden, A. Schenk and R. Schreuder (eds), Idioms. Structural and Psychological Perspectives, 1995, Hillsdale, N.J.—Hove: Lawrence Erlbaum Associates, 167-232.

[2] This work was partly supported by the RBRF grant 12-07-00663 and the RFH grant 13-04-00343, which is gratefully acknowledged.

Igor Mel'čuk. 1996. *Lexical Functions: A Tool for the Description of Lexical Relations in the Lexicon*. In: L. Wanner (ed.), *Lexical Functions in Lexicography and Natural Language Processing*, Amsterdam/Philadelphia: Benjamins, 37-102.

Adam Meyers, Catherine Macleod, Roman Yangarber, Ralph Grishman, Leslie Barrett, Ruth Reeves. 1998. *Using NOMLEX to Produce Nominalization Patterns for Information Extraction.* Coling-ACL98 workshop Proceedings: the Computational Treatment of Nominals Montreal, Canada, August, 1998.

Lucien Tesnière.1959. Éléments de syntaxe structurale, Klincksieck, Paris.

Active Learning in Noisy Conditions for Spoken Language Understanding

Hossein Hadian
Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
hadian@ce.sharif.edu

Hossein Sameti
Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
sameti@sharif.edu

Abstract

Active learning has proved effective in many fields of natural language processing. However, in the field of spoken language understanding which is always dealing with noise, no complete comparison between different active learning methods has been done. This paper compares the best known active learning methods in noisy conditions for spoken language understanding. Additionally a new method based on Fisher information named as Weighted Gradient Uncertainty (WGU) is proposed. Furthermore, Strict Local Density (SLD) method is proposed based on a new concept of *local density* and a new technique of utilizing information density measures. Results demonstrate that both proposed methods outperform the best performance of the previous methods in noisy and noise-free conditions with SLD being superior to WGU slightly.

1 Introduction

Spoken language understanding (SLU) is currently an emerging field in the intersection of speech processing and natural language processing (Tur and De Mori, 2011). The task of an SLU system is to extract meaning from speech utterances. Example real-world applications are AT&T's How May I Help You? and BBN's Call Director. In the field of SLU, as well as other fields of natural language processing, gathering data is fairly cheap but labeling is quite expensive and time-consuming. Thus, active learning methods apply very well and can greatly reduce costs. This article evaluates different techniques of active learning in the context of statistical SLU to reduce the labeling effort as much as possible. Also, SLU deals with the most amount of noise, in comparison with other fields of NLP, making robustness one of its most important issues (Tur and De Mori, 2011). Therefore, in this article noisy conditions of SLU are explored too. In this paper, we concentrate on statistical approaches for modeling the SLU system. Specifically conditional random fields (Lafferty et al., 2001) are used with a flat semantic frame to represent meaning and to model the SLU system.

While there have been a couple of studies on active learning in the context of SLU, they have mostly used only methods in the frameworks of uncertainty sampling (Tur et al., 2003; Jars and Panaget, 2008) and query-by-committee (Gotab et al., 2009). In addition, noisy conditions which are an important aspect of SLU have not been addressed thoroughly.

In this paper, performance of various known active learning methods namely uncertainty sampling, query-by-committee, Fisher information ratio (Settles and Craven, 2008) and instability sampling (Zhu

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1081–1090, Dublin, Ireland, August 23-29 2014.

and Ma, 2012) are examined and analyzed in noise-free and noisy conditions of SLU. Also a new method for measuring informativeness of instances based on the Fisher information framework is developed and evaluated along with other methods. Besides, to deal with noisy conditions, the new concept of *local density* and a new technique to utilize density measures are introduced and described.

The rest of this paper is organized as follows: Section 2 briefly describes CRFs, pool-based active learning framework, and selected active learning methods applicable to CRFs. Section 3 describes the first proposed method: Weighted Gradient Uncertainty. Section 4 introduces the local density concept, describes its motives and the proposed method of SLD is described. In Section 5, the noise model is described and experiments are performed in both noisy and noise-free conditions. Finally in Section 6 conclusions are derived.

2 Active Learning and CRFs

CRFs (Lafferty et al., 2001) are statistical graphical models which have demonstrated state-of-the-art accuracy in many fields as well as in SLU. A linear-chain CRF with parameter vector $\vec{\theta}$, defines the probability of $\vec{y}$ being the true label sequence for observation sequence $\vec{x}$ (with length T) as:

$$P(\vec{y}|\vec{x};\vec{\theta}) = \frac{1}{Z_{\vec{\theta}}(\vec{x})} \cdot \exp\left(\sum_{j=1}^{T}\sum_{i=1}^{K}\theta_i f_i(y_{j-1}, y_j, \vec{x}, j)\right). \quad (1)$$

$Z_{\vec{\theta}}(\vec{x})$ is the normalization factor and ensures that sum of $P(\vec{y}|\vec{x};\vec{\theta})$ over all possible labelings equals 1. There are K feature functions $f_k(y_{j-1}, y_j, \vec{x}, j)$ in a linear-chain CRF along with their weights θ_k. Each feature f_k, is a function of the whole observation sequence, the position of current observation and the current and previous labels. Training is the process of finding the optimum weight vector $\vec{\theta}$ to maximize the conditional log-likelihood of training instances in the labeled data set $\mathcal{L}$:

$$\ell(\mathcal{L};\vec{\theta}) = \sum_{(\vec{x},\vec{y})\epsilon\mathcal{L}} \log P(\vec{y}|\vec{x};\vec{\theta}) - \sum_{k=1}^{K}\frac{\theta_k^2}{\sigma^2}. \quad (2)$$

The second term is a regularization penalty to prevent over-fitting. After training, the labels can be predicted using the Viterbi algorithm.

Given: Labeled set $\mathcal{L}$, unlabeled pool $\mathcal{U}$, query startegy $\phi(\cdot)$, query batch size $\mathcal{B}$
repeat
  // learn a model using the current $\mathcal{L}$
  $\theta = train(\mathcal{L})$;
  for $b = 1$ **to** $\mathcal{B}$ **do**
    // query the most informative instance
    $\mathbf{x}_b^* = \underset{x\epsilon\mathcal{U}}{\operatorname{argmax}}\, \phi(\boldsymbol{x})$;
    // move the labeled query from $\mathcal{U}$ *to* $\mathcal{L}$
    $\mathcal{L} = \mathcal{L} \cup \langle \mathbf{x}_b^*, label(\mathbf{x}_b^*)\rangle$;
    $\mathcal{U} = \mathcal{U} - \mathbf{x}_b^*$;
  end
until *some stopping criterion;*

Figure 1. Pool-based active learning (Settles and Craven, 2008).

The focus of this paper is on pool-based active learning in which a learner should select most informative instances for labeling from a pool of unlabeled ones. We adopt the same notation used by Settles and Craven (2008) for the generic pool-based algorithm, sketched in Figure 1. Query strategy

$\phi(\cdot)$ is a function which evaluates how informative an unlabeled instance is. Most methods of active learning are a definition for this function. In the following subsections the best known active learning methods are briefly described.

2.1 Uncertainty Sampling

In this very common framework the learner queries the instance that it is most uncertain how to label. Two methods in this framework proved effective according to Settles and Craven (2008) which are presented here. First is the **least confident (LC)** method:

$$\phi^{LC}(\vec{x}) = 1 - P(\vec{y}^*|\vec{x};\theta), \quad (3)$$

where $\vec{y}^*$ is the most likely label sequence. Second query strategy is the **sequence entropy (SE)** method which measures informativeness of an instance based on entropy in different labelings:

$$\phi^{SE}(\vec{x}) = -\sum_{\vec{y}\epsilon\mathcal{Y}} P(\vec{y}|\vec{x};\theta)\log P(\vec{y}|\vec{x};\theta), \quad (4)$$

where $\mathcal{Y}$ is the set of all possible labelings for $\vec{x}$.

2.2 Query-By-Committee

Query-by-committee (QBC) is another well-studied and common framework for active learning. There are many approaches in this framework, but we use the approach suggested by Settles and Craven (2008) which has performed best with CRFs: in each round of active learning, $\mathcal{L}$ is sampled $|\mathcal{L}|$ times (with replacement) to create a unique modified labeled set $\mathcal{L}^{(c)}$. This is done C times to create C unique labeled sets. Then a committee of C models is trained: Each model $\theta^{(c)}$ is trained using its corresponding labeled set $\mathcal{L}^{(c)}$. Then the disagreement among the committee members about labeling an instance is measured as its informativeness:

$$\phi^{QBC}(\vec{x}) = -\sum_{\vec{y}\epsilon\mathcal{N}^C} P(\vec{y}|\vec{x};C)\, log\, P(\vec{y}|\vec{x};C). \quad (5)$$

In this equation, $\mathcal{N}^C$ is the union of N-best labelings of all models in the committee, and $P(\vec{y}|\vec{x};C) = \frac{1}{C}\sum_{c=1}^{C} P(\vec{y}|\vec{x};\theta)$ is the consensus posterior probability for some label sequence $\vec{y}$.

2.3 Representativeness

It is suggested that considering representativeness of instances can reduce the chance of selecting outliers in the process of active learning (Roy and McCallum, 2001). Representativeness can be measured by density of each instance, defined as the average similarity of an instance to other instances. Because the computation of density can be quite time-consuming in large-scale data sets, it is suggested to compute density in clusters (Tang et al., 2002; Shen et al., 2004) or in a k-Nearest-Neighbor manner (Zhu et al., 2008). Representativeness is applied by multiplying density to any arbitrary uncertainty measure to prevent outliers. Settles and Craven (2008) define a query strategy based on density:

$$\phi^{ID}(\vec{x}) = \phi^{LC}(\vec{x}) \times [ID(\vec{x})]^{\beta}, \quad (6)$$

$$ID(\vec{x}) = \frac{1}{U}\sum_{u=1}^{U} Sim(\vec{x}, \vec{x}^{(u)}). \quad (7)$$

Parameter β controls the relative effect of density $ID(\vec{x})$. This density uses a similarity measure $Sim(\cdot,\cdot)$ to compute the average similarity of an instance with all other unlabeled instances. The similarity measure used by Settles and Craven (2008) is a cosine similarity between two instances after being transformed to a vector of fixed length using this relation:

$$\vec{x} = \left[\sum_{t=1}^{T} f_1(x_t), \ldots, \sum_{t=1}^{T} f_J(x_t)\right], \quad (8)$$

where $f_j(x_t)$ is the value of feature f_j for token x_t, and J is the number of features in input representation. These features can be generated using CRF feature templates. Please refer to Settles and Craven (2008) for more details.

2.4 Fisher Information

We also evaluate the FIR (Fisher Information Ratio) method proposed by Settles and Craven (2008). Two vectors based on Fisher information are defined:

$$\mathcal{I}_{\vec{x}}(\theta) = \sum_{\hat{y}\epsilon\mathcal{N}} P(\hat{y}|\vec{x};\theta)\left[\left(\frac{\partial \log P(\hat{y}|\vec{x};\theta)}{\partial\theta_1}\right)^2 + \delta, \dots, \left(\frac{\partial \log P(\hat{y}|\vec{x};\theta)}{\partial\theta_K}\right)^2 + \delta\right], \quad (9)$$

$$\mathcal{I}_{\mathcal{U}}(\theta) = \frac{1}{|\mathcal{U}|}\sum_{u=1}^{|\mathcal{U}|}\mathcal{I}_{\vec{x}(u)}(\theta), \quad (10)$$

where $\mathcal{I}_{\vec{x}}(\theta)$ and $\mathcal{I}_{\mathcal{U}}(\theta)$ are the Fisher information matrices for sequence $\vec{x}$ and unlabeled pool $\mathcal{U}$ respectively. These matrices are estimated using their diagonal due to performance issues. Also K is the total number of CRF features, $\mathcal{N}$ is the set of N-best label sequences for input $\vec{x}$ and constant δ<<1 is added to prevent division by zero. Finally, FIR measures the informativeness of instances using:

$$\phi^{FIR}(\vec{x}) = -trace\left(\mathcal{I}_x(\theta)^{-1}\mathcal{I}_{\mathcal{U}}(\theta)\right). \quad (11)$$

2.5 Instability Sampling

(Zhu and Ma, 2012) suggest selecting instances which are most unstable. They propose two new methods to select most unstable instances based on recent active learning cycles: label-insensitive instability sampling (LIIS) and label-sensitive instability sampling (LSIS). Given an unlabeled instance $\vec{x}$ at i[th] learning cycle, its instability value in LIIS is estimated by:

$$\phi^{LIIS}(\vec{x}) = \phi_i^{SE}(\vec{x}) + \sum_{i-l<k\leq i}\left(\phi_k^{SE}(\vec{x}) - \phi_{k-1}^{SE}(\vec{x})\right), \quad (12)$$

where $\phi_i^{SE}(\vec{x})$ is $\phi^{SE}(\vec{x})$ at i[th] learning cycle and l is the number of cycles considered for instability estimation. Likewise, the instability value of $\vec{x}$ in LSIS is estimated by:

$$\phi^{LSIS}(\vec{x}) = \phi_i^{SE}(\vec{x}) + \sum_{i-l<k\leq i}\delta\left(\vec{y}^{(k)},\vec{y}^{(k-1)}\right)\left(\phi_k^{SE}(\vec{x}) - \phi_{k-1}^{SE}(\vec{x})\right), \quad (13)$$

where $\delta\left(\vec{y}^{(k)},\vec{y}^{(k-1)}\right)$ is 0 if the predicted label sequences $\vec{y}^{(k)}$ and $\vec{y}^{(k-1)}$ are the same and 1 otherwise. It's worthwhile to point that none of the instability sampling methods have been evaluated in the context of sequence labeling and they have only been evaluated in the context of classification.

3 The First Proposed Method: Weighted Gradient Uncertainty (WGU)

The new method to be introduced in this article is an improvement over the FIR method (subsection 2.4). According to evaluations by Settles and Craven (2008), the FIR method didn't perform well in practice despite its sound theory. In this section, first we investigate the essence of each component of $\mathcal{I}_{\vec{x}}(\theta)$:

$$\mathcal{I}_{\vec{x}}(\theta)_k = \sum_{\hat{y}\epsilon\mathcal{N}} P(\hat{y}|\vec{x};\theta)\left(\frac{\partial\ log\ P(\hat{y}|\vec{x};\theta)}{\partial\theta_k}\right)^2. \quad (14)$$

According to this relation, the k[th] component of Fisher vector $\mathcal{I}_{\vec{x}}(\theta)$ is the weighted sum of squared gradients of log-probabilities for the N best labelings for instance $\vec{x}$ in k[th] dimension of CRF features. It can be seen intuitively that each component of the Fisher vector increases when there is a kind of entropy between the N-best probabilities. That's because when for example the best label sequence has probability 1 then its gradient will be zero in all dimensions (complete fit) and hence all the components will be zero. On the other hand, if N best label sequences have equal probabilities, none of them will have a zero gradient since none is a complete fit and $\mathcal{I}_{\vec{x}}(\theta)$ will be maximized.

To show this fact more rigidly, assume the N best label sequences as $\mathcal{N} = \{\vec{y}^{(1)}, \vec{y}^{(2)}, \dots, \vec{y}^{(N)}\}$, and also for simplicity, define: $P_n = P\left(\vec{y}^{(n)}\middle|\vec{x};\theta\right)$. Then we will have:

$$logP_n = \sum_{j=1}^{T}\sum_{i=1}^{K}\theta_i f_i\left(y_{j-1}^{(n)}, y_j^{(n)}, \vec{x}, j\right) - logZ_{\vec{\theta}}(\vec{x}), \quad (15)$$

and so, its partial derivative in k[th] dimension will be (assuming $\mathcal{N}$ contains all possible label sequences):

$$\frac{\partial log P_n}{\partial \theta_k} = \overbrace{\sum_{j=1}^{T} f_k\left(y_{j-1}^{(n)}, y_j^{(n)}, \vec{x}, j\right)}^{F_n^{(k)}} - \frac{1}{Z_{\vec{\theta}}(\vec{x})} \frac{\partial Z_{\vec{\theta}}(\vec{x})}{\partial \theta_k} = F_n^{(k)} - \sum_{m=1}^{N} P_m F_m^{(k)}, \quad (16)$$

where $F_n^{(k)}$ is the result of applying feature function f_k (from CRF model) on nth best label sequence. Now using (16) we can rewrite (14) as:

$$\mathcal{I}_{\vec{x}}(\theta)_k = \sum_{n=1}^{N} P_n \left(F_n^{(k)} - \sum_{m=1}^{N} P_m F_m^{(k)} \right)^2. \quad (17)$$

To fully understand each component, we further factorized the above relation and proved it to be equal to (the proof is omitted here for brevity):

$$\mathcal{I}_{\vec{x}}(\theta)_k = \sum_{i=1}^{N} \sum_{j=i+1}^{N} P_i P_j \left(F_i^{(k)} - F_j^{(k)} \right)^2. \quad (18)$$

This relation explains the meaning of components of the Fisher vector completely. Each component is a summation over N best label sequences. The expression under summation consists of two parts: $P_i P_j$ and $\left(F_i^{(k)} - F_j^{(k)}\right)^2$. It can be shown using Lagrange multipliers that the first part is maximized (independently) when $P_i = \frac{1}{N}, \forall i$; which means this part is maximized when maximum entropy between N best probabilities occurs. The second part is the squared difference of kth feature function applied to two label sequences. So this part is maximized when the dissimilarities between every two label sequences in N-best list are maximum, which in turn means the model has maximum uncertainty in choosing the N-best label sequences for the input. Notice that in this interpretation we have assumed the two parts to be independent while they are not actually. However since the number of features of CRF (i.e. K) is too large, the dependency is negligible and can be ignored. So we conclude that each component of the Fisher vector $\mathcal{I}_{\vec{x}}(\theta)$ is a measure of uncertainty of the model about the sequence $\vec{x}$ in the corresponding dimension. Accordingly, each component of the total Fisher information vector $\mathcal{I}_{\mathcal{U}}(\theta)$ is the average uncertainty of the model in the corresponding dimension.

Knowing the precise identity of Fisher vector $\mathcal{I}_{\vec{x}}(\theta)$, we propose a natural measure which we call Weighted Gradient Uncertainty (WGU) based on the facts explained in the previous paragraph:

$$\phi^{WGU}(\vec{x}) = \sqrt{\sum_j \mathcal{I}_{\mathcal{U}}(\theta)_j \left(\mathcal{I}_{\vec{x}}(\theta)_j\right)^2}. \quad (19)$$

This measure is the weighted norm of $\mathcal{I}_{\vec{x}}(\theta)$ with the total Fisher information vector $\mathcal{I}_{\mathcal{U}}(\theta)$ as the weight vector. This query strategy favors instances with high uncertainty in each dimension of CRF feature space, especially the dimensions where the average uncertainty is higher. In other terms, the WGU measure maximizes the components of the Fisher vector, while the FIR method minimizes the inversed components of the Fisher vector; and since many components of the Fisher vector are zero or near-zero, their inversed values are very large and block out the other larger components (with very small inverse values) leading to a measure which effectively just counts the number of zero components and chooses the instance with the maximum number of zero components.

4 Using Local Density for Noisy Conditions

As described in Introduction, a great issue in SLU systems is the presence of noise in utterances. To address this problem, all the ATIS[1] instances were converted to vectors according to (8) and were reduced to 2 dimensions using Principle Component Analysis (PCA). Then the global density $ID(\vec{x})$ for

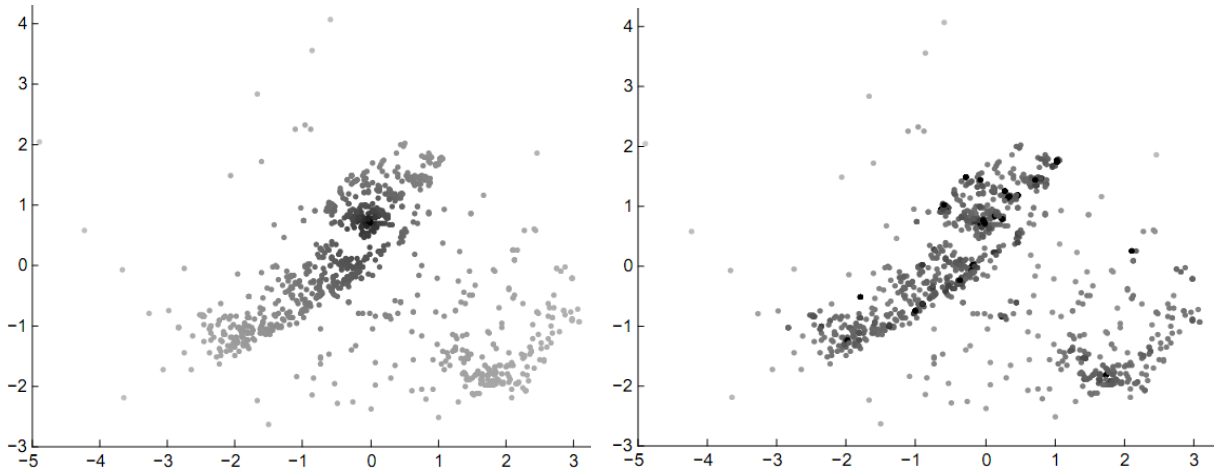
[1] ATIS is the dataset used in this article for evaluation; please read subsection 5.1.

each instance was computed using (7). Figure 2 shows the plot of all instances with darker points indicating instances with higher densities and lighter points showing the ones with lower densities.

As seen in Figure 2(a), the center of the distribution in terms of density is the darkest part. Also, the distribution of instances is not uniform at all, and excluding any part of the distribution especially parts further from the density center can lead to great decrease in performance of the model. The query strategy ϕ^{ID} (6) uses this density to reduce the chance of querying outliers. However, outliers as well as many other instances which are far from the density center are almost deprived of the chance of being selected. To address this problem and yet avoid outliers we choose to compute information density for each instance locally, i.e. using k nearest instances and not all instances. Thus, we define the local information density measure as follows:

$$LD(\vec{x}, k) = \frac{1}{k} \sum_{\vec{x}' \in \Gamma_k(\vec{x})} \text{Sim}(\vec{x}, \vec{x}'). \tag{20}$$

In which, $\Gamma_k(\vec{x})$ is the set of k most similar instances to $\vec{x}$, and k is the degree of locality.



(a) (b)

Figure 2. Plot of all ATIS instances. Darker points show higher densities and lighter ones show lower densities. (a) Using global density measure (b) Using local density measure (k=5).

The same procedure to plot Figure 2(a) is repeated again but with $LD(\vec{x}, k)$ computed as the density of each instance and the result is shown in Figure 2(b). The degree of locality is set to k=5. As seen in this plot, outliers are still completely grey which means they are avoided. Also, any small neighborhood with sufficient density is biased to black, which means the instances in the center of that neighborhood have almost the same chance of being queried as the instances in the center of global density (ID) in Figure 2(a).

Another advantage of local density is that it avoids noisy instances. Noisy instances in the SLU context are the utterances in which one or more words are erroneous due to ASR or user errors. Because of such errors, noisy instances take a small distance from their similar instances and reside alone in small neighborhoods.

Based on the LD measure (20), two active learning methods are considered: the first method applies local density measure to query strategy by multiplication (same as ϕ^{ID}):

$$\phi^{LD}(\vec{x}) = \phi^{LC}(\vec{x}) \times [LD(\vec{x}, k)]^{\beta}. \tag{21}$$

The second method which is proposed in this paper, strictly applies the local density measure by first filtering out instances with local densities lower than a threshold T, and then queries the most informative instance according to a certain query strategy (here we use ϕ^{LC}). This method is called Strict LD (SLD). We believe that this method of utilizing density measures is more effective than the traditional method (i.e. multiplying density measure by uncertainty measure (6)), since it does not affect all instances but

only very low-density ones. The threshold T is assumed to be in the form of $\alpha * \overline{LD}$, where $\overline{LD}$ is the average of local density over all unlabeled instances, and parameter α sets the intensity of filtering.

It is necessary to note that the k-Nearest-Neighbor density measure (Zhu et al., 2008) is identical to local density in definition but the motivation is different and in this article we look at the k-nearest-neighbor density from a completely different perspective: to avoid a shortcoming in the global density which is ignoring great parts of the input distribution and also to detect noisy instances.

5 Experiments

Experiments are all performed on the ATIS[1] data set (Hemphil et. al, 1990), both in noise-free and noisy conditions. In this section, the noise model used to generate noise is briefly described and then the evaluations are presented.

5.1 ATIS and Noise Model

ATIS is a relatively simple corpus which contains air travel information data. This corpus is the most commonly used data set for SLU research (Tur et. al, 2010). The data set contains questions (utterances) about flight, airport, and airline information. We specifically use the class-A (context independent) utterances from ATIS-3 corpus (Dahl et. al, 2004). These utterances are not semantically labelled, instead for each utterance there is an SQL command which queries the answer to the utterance from database. Thus a flat sematic representation was designed and semantic label sequences were generated semi-automatically from the SQL queries (as explained by He and Young (2006)). The flat semantic representation is listed in Table 1(a). A flat semantic representation is in fact a set of attributes (semantic labels) which are used to label an input utterance. Table 1(c) shows a typical utterance with sematic labels; note that IOB labeling scheme is used. Totally there are 1630 class-A instances (test + train) in ATIS-3 which are used in the experiments.

Attribute	Description	Attribute	Description
DCity	depart. city	ACity	arrival city
SCity	stop city	DAir	depart. airport
DDate	depart. date	ADate	arrival date
RDate	return date	AAir	arrival airport

(a)

Origin	Pair
ASR	via → fly at
Human	to Chicago → chica to Chicago
ASR	phoenix → t x

(b)

Show	flights	from	Denver	to	Washington	on	Sunday	arriving	before	noon
O	O	O	DCity	O	ACity	O	DDate	O	ADate-I	ADate-I

(c)

Table 1: (a) The flat semantic representation used to label utterances in the data set. (b) Some example pairs in noise model. Each pair is extracted from actual errors in ATIS-3 utterances. (c) A typical example from ATIS utterances.

Utterances in ATIS are de-noised by wizards[2]. There are two origins of noise: human (end-user) errors and ASR recognition errors. We design a simple noise-model based on actual errors and regenerate human and ASR errors. In ATIS-3, human errors are marked in SRO files and ASR errors are in N-best lists in log files. The noise model is a list of pairs of the form [correct-expression] → [erroneous-expression] which are applied to ATIS instances to add arbitrary percentage of noise. A few example pairs in the noise model are listed in Table 1(b). Each pair is extracted from an actual error; for example [phoenix] → [t x] is a result of an ASR error in ATIS-3 logs where "phoenix" in "Show me flights from phoenix …" was recognized as "t x" mistakenly. Obviously this pair is only applicable to an utterance which contains the word "phoenix".

[1] Air Travel Information System

[2] A wizard is a human expert who transcribes utterances or answers them (Hemphil et. al, 1990).

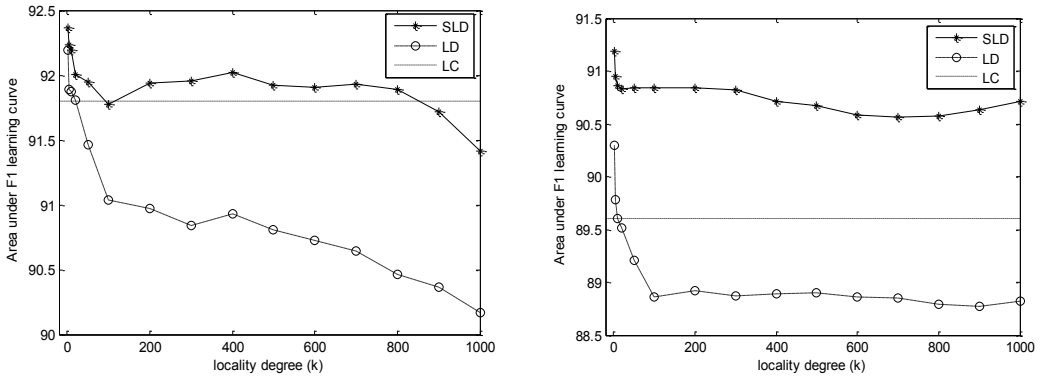
5.2 Parameter Settings

Using the noise model described, 3 levels of noise were generated: 7% of instances in level 1, 15% in level 2, and 25% in level 3 are noisy. In noisy conditions, when a noisy instance is selected by an active learning method, we assume that the instance is correctly detected as noisy by the annotator and is rejected (i.e. not added to $\mathcal{L}$); but the determination of an instance as noisy incurs a cost which we assume to be a quarter of cost of labeling one instance[1]. In all experiments, $\mathcal{L}$ is initialized with 5 random training instances. Batch size in all experiments is set to B=2 and new instances are added to $\mathcal{L}$ until the total labeling cost reaches 100. For query-by-committee method, we set C=4 and N=20 to balance between speed and accuracy. For LD and SLD, we set k=1 because it achieved best performance. For LIIS and LSIS, we set l=2 which achieved better results. Each method is evaluated as the average of 5 trials and each trial is performed using 5-fold cross validation. The reported performance for each method is the area under F1 learning curve (F1 score in SLU is computed as described by Tur and De Mori (2011)).

5.3 Effect of Locality

By initial evaluations, β=1 and α=0.6 were chosen for the LD and SLD method respectively. In Figure 3, the performances of LD and SLD for different degrees of locality (for k=1 to 1000) are shown. The performance of the LC method is also shown for comparison.

As seen in Figure 3, local density improves uncertainty measure (i.e. ϕ^{LC}, which is the base method in LD and SLD) and performs better than global density (i.e. local density with k=1000+). Note that LD has led to better performances than LC only for very local densities (i.e. k<5) while SLD has improved the performance of LC almost for all degrees of locality. It can also be seen that applying density strictly is more effective than the traditional way for all degrees of locality especially in noisy conditions.



(a) Noise-free condition (b) Noisy condition (average of all levels)

Figure 3. Effect of locality degree (in computation of information density) on performance of active learning methods. Plots (a) and (b) show the area under F1 learning curve for different values of k in LD and SLD methods, for noise-free and noisy conditions respectively. The area under F1 learning curve for LC is also shown for comparison.

5.4 Evaluations

The detailed results of the discussed active learning methods on different levels of noise are presented in Table 2. In each row, best performance is bolded and underlined, and second best performance is just bolded. Random refers to the random sampling of instances (passive learning). In noise-free condition, LD and SLD have improved a little over LC, but in average, SLD has performed remarkably better than LC, which shows the effectiveness of using local density to avoid noisy instances (note that LC is the base method used in LD and SLD). The instability sampling methods have improved over uncertainty

[1] The cost of labelling one instance is equal to 1 for any instance. In this paper, learning curves are depicted in terms of annotation cost which is equivalent to annotation time (please refer to Tomanek and Hahn (2010)).

sampling (i.e. SE) but not significantly. In the last row of Table 2 the running time of one cycle of active learning for each method is presented in seconds. QBC is the slowest method and LC is the fastest one. WGU is the second best in average performance but is rather slow in comparison to LC and this is a disadvantage of WGU. In fact all methods that iterate over best labelings are considerably slower that LC.

Learning curves cannot be shown for all active learning methods due to lack of space. Instead, learning curves are shown for selected methods. In Figure 4, learning curves for five methods of SLD, WGU, LIIS, FIR, and random are shown. It can be seen that the new WGU method has the best performance in early stages of active learning but soon declines and stays above the curve of LIIS. Also, the difference of SLD with other methods is more remarkable in the noisy conditions.

	Random	LC	SE	QBC	ID	FIR	LIIS	LSIS	WGU	LD	SLD
Noise-free	84.5	91.8	91.9	90.5	90.2	89.5	91.7	91.8	**92.1**	**92.1**	**92.4**
Noise level 1	84.1	**91.5**	90.7	90	89.6	89.1	91.1	90.8	**91.7**	91.2	**91.7**
Noise level 2	83.2	88.9	89	88.2	88.1	89.2	89.4	88.9	**90.4**	89.4	**91.1**
Noise level 3	83	88.4	88.4	87.5	87.7	**88.9**	88.7	87.8	**90**	89.3	**91**
Average	83.7	90.1	90	89	88.9	89.2	90.2	89.8	**91.1**	90.5	**91.6**
Runtime	**5**	**5**	8	20	**5.5**	8	8	8	8	**5.5**	**5.5**

Table 2. Area under F1 learning curves (max possible score is 100) and runtimes of various active learning methods on different levels of noise.

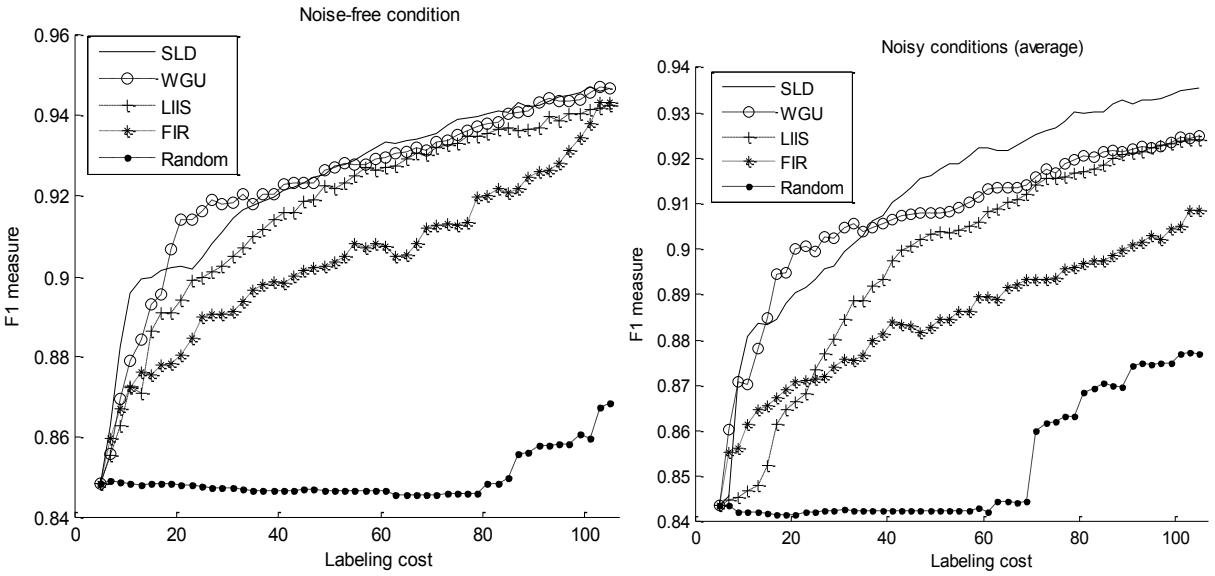


Figure 4. Learning curves for five selected methods: SLD, WGU, LIIS, FIR, and random for noise-free and noisy conditions (averaged across noise levels 1-3). Each learning curve shows the F1 measures achieved by the corresponding method for different labelling costs up to 100.

6 Conclusion

In this paper, best known active learning methods applicable to sequence labeling tasks were evaluated in the field of SLU (Spoken Language Understanding) in real conditions of noise. The new method of WGU (Weighted Gradient Uncertainty) with theoretical justification was proposed and performed well in the evaluations. Also, to deal directly with noisy instances, two methods of LD (Local Density) and SLD (Strict LD) were proposed based on the local density concept. It is possible to apply local density to WGU or other methods to achieve even better results but this could be the subject of future work.

References

Burr Settles and Mark Craven. 2008. *An Analysis of Active Learning Strategies for Sequence Labeling Tasks*, In EMNLP '08 Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1070-1079.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. *The ATIS spoken language systems pilot corpus*. In Proceedings of the workshop on Speech and Natural Language. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 96-101.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. *Expanding the scope of the ATIS task: the ATIS-3 corpus*. In Proceedings of the workshop on Human Language Technology (HLT '94). Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 43-48.

Gokhan Tur, Dilek Hakkani-Tür, and Larry P. Heck. 2010. *What is left to be understood in ATIS?* IEEE Spoken Language Technology Workshop (SLT), Berkeley, California, USA, December 12-15, pp. 19-24.

Gokhan Tur and Renato De Mori. 2011. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*, First Edition, John Wiley & Sons.

Gokhan Tur, Marzin Rahim, and Dilek Hakkani-Tür. 2003. *Active Learning for Spoken Language Understanding*, In Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 276-279.

Isabelle Jars and Franck Panaget. 2008. *Improving Spoken Language Understanding with information retrieval and active learning methods*, In Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 5001-5004.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*, In Proceedings of the International Conference on Machine Learning (ICML), pp. 282–289.

Jingbo Zhu and Matthew Ma. 2012. *Uncertainty-based active learning with instability estimation for text classification*. ACM Transactions on Speech and Language Processing (TSLP), vol. 8(4) - 01/2012.

Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K. Tsou. 2008. *Active learning with sampling by uncertainty and density for word sense disambiguation and text classification*. In Proceedings of the 22nd International Conference on Computational Linguistics (COLING '08), Vol. 1, pp. 1137-1144.

Katrin Tomanek and Udo Hahn. 2010. *A comparison of models for cost-sensitive active learning*. In Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10), pp. 1247-1255.

Lynette Hirschman. 1992. *Multi-Site Data Collection for a Spoken Language Corpus,* In Proceedings of International Conference on Spoken Language Processing, Banff, Canada.

Min Tang, Xiaoqiang Luo, and Salim Roukos. 2002. *Active learning for statistical natural language parsing*, In Proceedings of the Association for Computational Linguistics 40th Anniversary Meeting, pp.120–127.

Nicholas Roy and Andrew McCallum. 2001. *Toward optimal active learning through sampling estimation of error reduction*, In Proceedings of the International Conference on Machine Learning (ICML), pp. 441–448.

Pierre Gotab, Frédéric Béchet, and Géraldine Damnati. 2009. *Active learning for rule-based and corpus-based Spoken Language Understanding models*, In Proceedings of IEEE Conference on Automatic Speech Recognition and Understanding, pp. 444-449.

Yulan He and Steve Young. 2005. *Semantic processing using the hidden vector state model*, Computer Speech & Language, vol. 19, no. 1, pp. 85–106.

A Self-adaptive Classifier for Efficient Text-stream Processing

Naoki Yoshinaga
Institute of Industrial Science,
the University of Tokyo,
Meguro-ku, Tokyo 153-8505, Japan
ynaga@tkl.iis.u-tokyo.ac.jp

Masaru Kitsuregawa
Institute of Industrial Science,
the University of Tokyo,
Meguro-ku, Tokyo 153-8505, Japan
and
National Institute of Informatics,
Chiyoda-ku, Tokyo 101-8430, Japan
kitsure@tkl.iis.u-tokyo.ac.jp

Abstract

A self-adaptive classifier for efficient text-stream processing is proposed. The proposed classifier adaptively speeds up its classification while processing a given text stream for various NLP tasks. The key idea behind the classifier is to reuse results for past classification problems to solve forthcoming classification problems. A set of classification problems commonly seen in a text stream is stored to reuse the classification results, while the set size is controlled by removing the least-frequently-used or least-recently-used classification problems. Experimental results with Twitter streams confirmed that the proposed classifier applied to a state-of-the-art base-phrase chunker and dependency parser speeds up its classification by factors of 3.2 and 5.7, respectively.

1 Introduction

The rapid growth in popularity of microblogs (*e.g.*, Twitter) is enabling more and more people to instantly publish their experiences or thoughts any time they want from mobile devices. Since information in text posted by hundreds of millions of those people covers every space and time in the real world, analyzing such a text stream tells us what is going on in the real world and is therefore beneficial for reducing damage caused by natural disasters (Sakaki et al., 2010; Neubig et al., 2011a; Varga et al., 2013), monitoring political sentiment (Tumasjan et al., 2010) and disease epidemics (Aramaki et al., 2011), and predicting stock market (Gilbert and Karahalios, 2010) and criminal incident (Wang et al., 2012).

Text-stream processing, however, faces a new challenge; namely, the quality (content) and quantity (volume of flow) changes dramatically, reflecting a change in the real world. Current studies on processing microblogs have focused mainly on the difference between the quality of microblogs (or spoken languages) and news articles (or written languages) (Gimpel et al., 2011; Foster et al., 2011; Ritter et al., 2011; Han and Baldwin, 2011), and they have not addressed the issue of so-called "bursts" that increase the volume of text. Although it is desirable to use NLP analyzers with the highest possible accuracy for processing a text stream, high accuracy is generally attained by costly structured classification or classification with rich features, typically conjunctive features (Liang et al., 2008). It is therefore inevitable to trade accuracy for speed by using only a small fraction of features to assure real-time processing.

In this study, the aforementioned text-quantity issue concerning processing a text stream is addressed, and a self-adaptive algorithm that speeds up an NLP classifier trained with many conjunctive features (or with a polynomial kernel) for a given text stream is proposed and validated. Since globally-observable events such as natural disasters or sports events incur a rapid growth in the number of posts (Twitter, Inc., 2011), a text stream is expected to contain similar contents concerning these events when the volume of flow in a text stream increases. To adaptively speed up the NLP classifier, the proposed algorithm thus enumerates common classification problems from seen classification problems and keeps their classification results as partial results for use in solving forthcoming classification problems.

The proposed classifier was evaluated by applying it to streams of classification problems generated during the processing of the Twitter streams on the day of the 2011 Great East Japan Earthquake and on another day in March 2012 using a state-of-the-art base-phrase chunker (Sassano, 2008) and dependency parser (Sassano, 2004), and the obtained results confirm the effectiveness of the proposed algorithm.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers,
pages 1091–1102, Dublin, Ireland, August 23-29 2014.

2 Related work

A sentence is the processing unit used for fundamental NLP tasks such as word segmentation, part-of-speech tagging, phrase chunking, syntactic parsing, and semantic role labeling. Most efficient algorithms solving these tasks thus aim at speeding up the processing based on this unit (Kaji et al., 2010; Koo et al., 2010; Rush and Petrov, 2012), and few studies have attempted to speed up the processing of a given text (a set of sentences) as a whole. In the following, reported algorithms that adaptively speed up NLP analyzers for a given text are introduced.

A method of speeding up a classifier trained with many conjunctive features by using precomputed results for common classification problems was proposed by Yoshinaga and Kitsuregawa (2009; 2012). It solves classification problems that commonly appear in the processing of a large amount of text in advance and stores the results in a trie, so that they can be reused as partial results for solving new classification problems. This method was reported to achieve speed-up factors of 3.3 and 10.6 for base-phrase chunking and dependency parsing, respectively. An analogous algorithm for integer linear program (ILP) used to solve structured classification was proposed by Srikumar, Kundu, and Roth (2012; 2013). The algorithm was reported to achieve speed-up factors of 2.6 and 1.6 for semantic role labeling and entity-relation extraction, respectively. Although these two algorithms can be applied to various NLP tasks that can be solved by using a linear classifier or an ILP solver, how effective they are for processing a text stream is not clear.

A method of feature sharing for beam-search incremental parsers was proposed by Goldberg et al. (2013). Motivated by the observation that beam parsers solve similar classification problems in different parts of the beam, this method reuses partial results computed in the previous beam items. It reportedly achieved a speed-up factor of 1.2 for arc-standard and arc-eager dependency parsers. The key differences between the method proposed in this study and their feature-sharing method are twofold. First, the feature sharing in Goldberg et al. (2013) is performed in a token-wise manner in the sense that a key to retrieve a cached result is represented by a bag of tokens that invoke features, which manner prevents fine-grained caching. Second, the feature sharing is dynamically performed during parsing, but the cached results are cleared after processing each sentence.

An adaptive pruning method for fast HPSG parsing was proposed by van Noord (2009). This method preprocesses a large amount of text by using a target parser to collect derivation steps that are unlikely to contribute to the best parse, and it speeds up the parser by filtering out those unpromising derivation steps. Although this method was reported to attain a speed-up factor of four while keeping parsing accuracy, it needs to be tuned to trade parsing accuracy and speed for each domain. It is difficult to derive the true potential of their method in regard to processing a text stream whose domain shifts from time to time.

It has been demonstrated by Wachsmuth et al. (2011) that tuning a pipeline schedule of an information extraction (IE) system improves the efficiency of the system. Furthermore, the self-supervised learning algorithm devised by Wachsmuth et al. (2013) predicts the processing time for each possible pipeline schedule of an IE system, and the prediction is used to adaptively change the pipeline schedule for a given text stream. This method and the proposed method for speeding up an NLP classifier are complementary, and a combination of both methods is expected to synergistically speed up various NLP-systems.

In this study, based on the classifier proposed by Yoshinaga and Kitsuregawa (2009), a self-adaptive classifier that enumerates common classification problems from a given text stream and reuses their results is proposed. As a result, the proposed classifier adaptively speeds up the classification of forthcoming classification problems.

3 Preliminaries

As the basis of the proposed classifier, the previously-presented classifier that uses results of common classification problems (Yoshinaga and Kitsuregawa, 2009) is described as follows. This base classifier targets a linear classifier trained with many conjunctive features (including one converted from a classifier trained with polynomial kernel (Isozaki and Kazawa, 2002)) that are widely used for many NLP tasks. Although this classifier (and also the one proposed in this paper) can handle a multi-class classification problem, a binary classification problem is assumed here for brevity.

A binary classifier such as a perceptron and a support vector machine determines label $y \in \{+1, -1\}$ of input classification problem $\boldsymbol{x}$ by using the following equation (from which the bias term is omitted for brevity):

$$m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x}) = \sum_i w_i \phi_i(\boldsymbol{x}) \tag{1}$$

$$y = \begin{cases} +1 & (m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w}) \geq 0) \\ -1 & (m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w}) < 0). \end{cases} \tag{2}$$

Here, ϕ_i is a feature function, w_i is a weight for ϕ_i obtained as a result of training, and $m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w})$ is a margin between $\boldsymbol{x}$ and the separating hyperplane.

In most NLP tasks, feature functions are mostly indicator (or binary) functions that typically represent particular linguistic constraints. Here, feature functions are assumed to be indicator functions that return $\{0, 1\}$, and margin $m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w})$ is represented by the following equation:

$$m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w}) = \sum_i w_i \phi_i(\boldsymbol{x}) = \sum_{i, \phi_i(\boldsymbol{x})=1} w_i. \tag{3}$$

Feature function ϕ_i is hereafter referred to as feature ϕ_i; when $\phi_i(\boldsymbol{x}) = 1$ for given $\boldsymbol{x}$, $\boldsymbol{x}$ is said to "include" feature ϕ_i or feature ϕ_i is "active" in $\boldsymbol{x}$ (denoted as $\phi_i \in \boldsymbol{x}$). Having the number of active features, $|\boldsymbol{\phi}(\boldsymbol{x})| \equiv |\{\phi_i \mid \phi_i \in \boldsymbol{x}\}|$, Eq. 3 requires $\mathcal{O}(|\boldsymbol{\phi}(\boldsymbol{x})|)$ when the weights for the active features are summed up.

To speed up the summation in Eq. 3, classification results for some classification problems $\boldsymbol{x}_c$ are precomputed as $M_{\boldsymbol{x}_c} \equiv m(\boldsymbol{x}_c; \boldsymbol{\phi}, \boldsymbol{w})$ in advance, and then these precomputed results are reused as partial results for solving an input classification problem, $\boldsymbol{x}$:

$$m(\boldsymbol{x}; \boldsymbol{x}_c, \boldsymbol{\phi}, \boldsymbol{w}) = M_{\boldsymbol{x}_c} + \sum_{i, \phi_i \in \boldsymbol{x}, \phi_i \notin \boldsymbol{x}_c} w_i \tag{4}$$

$$\textit{where} \quad \forall \phi_j \in \boldsymbol{x}_c, \phi_j \in \boldsymbol{x}.$$

Note that for Eq. 4 to be computed faster than Eq. 3, $M_{\boldsymbol{x}_c}$ must be retrieved in time less than $\mathcal{O}(|\boldsymbol{\phi}(\boldsymbol{x}_c)|)$. It is actually possible when $\boldsymbol{x}_c$ includes conjunctive feature $\phi_{i,j}(\boldsymbol{x}_c) = \phi_i(\boldsymbol{x}_c)\phi_j(\boldsymbol{x}_c)$. If it is necessary to retrieve margin $M_{\boldsymbol{x}_c}$ precomputed for $\boldsymbol{x}_c$ including ϕ_i, ϕ_j, and $\phi_{i,j}$, it is necessary to check only non-conjunctive (or *primitive*) features ϕ_i and ϕ_j, since $\phi_{i,j}$ is active whenever ϕ_i and ϕ_j are active (so checking $\phi_{i,j}$ can be skipped). The second term of Eq. 4 sums up the weights of the remaining features that are not included in $\boldsymbol{x}_c$ but are included in $\boldsymbol{x}$. For example, under the assumption that $\boldsymbol{x}$ includes features ϕ_i, ϕ_j, ϕ_k, $\phi_{i,j}$, $\phi_{i,k}$, and $\phi_{j,k}$ and that margin $M_{\boldsymbol{x}_c}$ has been obtained for $\boldsymbol{x}_c$ (including ϕ_i, ϕ_j, and $\phi_{i,j}$), five features must be checked (two to retrieve $M_{\boldsymbol{x}_c}$ and three to sum up the weights of the remaining features ϕ_k, $\phi_{i,k}$ and $\phi_{j,k}$) by using Eq. 4. On the other hand, to compute $m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w})$ by Eq. 3, the weights for the six features must be checked.

To maximize the speed-up obtained by Eq. 4, reuse of margin $M_{\boldsymbol{x}_c}$ of common classification problem $\boldsymbol{x}_c$ should minimize the number of remaining features included only in $\boldsymbol{x}$. In other words, $\boldsymbol{x}_c$ should be as similar to $\boldsymbol{x}$ as possible (ideally, $\boldsymbol{x}_c = \boldsymbol{x}$). It is not, however, realistic to precompute margin $M_{\boldsymbol{x}} \equiv m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w})$ for every possible classification problem $\boldsymbol{x}$ since it requires $\mathcal{O}(2^{|\boldsymbol{\phi}'|})$ space where $|\boldsymbol{\phi}'|$ is the number of primitive features ($\boldsymbol{\phi}' \subset \boldsymbol{\phi}$) and $|\boldsymbol{\phi}'|$ is usually more than 10,000 in NLP tasks due to lexical features. Yoshinaga and Kitsuregawa (2009) therefore preprocess a large amount of text to enumerate possible classification problems, and select common classification problems, $\mathcal{X}_c \subset 2^{\boldsymbol{\phi}'}$, according to their probability and the reduction in the number of features to be checked by Eq 4.

Yoshinaga and Kitsuregawa (2009) then represent the common classification problems $\boldsymbol{x}_c \in \mathcal{X}_c$ by sequences of active primitive feature indices, and store those feature (index) sequences as keys in a prefix trie with precomputed margin $M(\boldsymbol{x}_c)$ as their values. To reuse a margin of common classification problem that is similar to input $\boldsymbol{x}$ in Eq. 4, features are ordered according to their frequency to form a feature sequence of $\boldsymbol{x}_c$. A longest-prefix search for the trie thereby retrieves a common classification problem similar to the input classification problem in linear time with respect to the number of primitive features in $\boldsymbol{x}_c$, $\mathcal{O}(|\boldsymbol{\phi}'(\boldsymbol{x}_c)|)$.

Algorithm 1 A self-adaptive classifier for enumerating common classification problems

INPUT: $\boldsymbol{x}$, $\boldsymbol{\phi}$, $\boldsymbol{\phi}' \subset \boldsymbol{\phi}$, $\boldsymbol{w} \in \mathbb{R}^{|\boldsymbol{\phi}|}$, $\mathcal{X}_c \subset 2^{\boldsymbol{\phi}'}$, $k > 0$
OUTPUT: $m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w}) \in \mathbb{R}$, $\mathcal{X}_c$
1: INITIALIZE: $\boldsymbol{x}_c$ s.t. $\boldsymbol{\phi}'(\boldsymbol{x}_c) = \mathbf{0}$, $M_{\boldsymbol{x}_c} \leftarrow 0$
2: **repeat**
3: $\quad \boldsymbol{x}_c^{old} \leftarrow \boldsymbol{x}_c$
4: $\quad \phi'_i = \underset{\phi'_i \in \boldsymbol{x}, \phi'_i \notin \boldsymbol{x}_c}{\text{argmax}}\ \text{FREQ}(\phi'_i)$ (extract a primitive feature according to its frequency)
5: $\quad \phi'_i(\boldsymbol{x}_c) \leftarrow 1$ (construct a new common-classification problem)
6: $\quad$ **if** $\boldsymbol{x}_c \notin \mathcal{X}_c$ **then**
7: $\quad\quad M_{\boldsymbol{x}_c} \leftarrow m(\boldsymbol{x}_c; \boldsymbol{x}_c^{old}, \boldsymbol{\phi}, \boldsymbol{w})$ (compute margin by using Eq. 4)
8: $\quad\quad$ **if** $|\mathcal{X}_c| = k$ **then**
9: $\quad\quad\quad \mathcal{X}_c \leftarrow \mathcal{X}_c - \{\text{USELESS}(\mathcal{X}_c)\}$
10: $\quad\quad \mathcal{X}_c \leftarrow \mathcal{X}_c \cup \{\boldsymbol{x}_c\}$
11: **until** $\boldsymbol{\phi}'(\boldsymbol{x}_c) \neq \boldsymbol{\phi}'(\boldsymbol{x})$
12: **return** $m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w}) = M_{\boldsymbol{x}_c}$, $\mathcal{X}_c$

4 Proposed method

The classifier described in Section 3 is extended so that it dynamically enumerates common classification problems from a given text stream[1] to adaptively speed up the classification. This classification "speed up" faces two challenges: which (partial) classification problems should be chosen to reuse their results from a given stream of classification problems, and how to efficiently maintain the extracted common classification problems. These two challenges are addressed in Sections 4.1 and 4.2, respectively.

4.1 Enumerating common classification problems dynamically from a text stream

Although Yoshinaga and Kitsuregawa (2009) select common classification problems according to their probability, such statistics cannot be known before a target text stream is entirely seen. A set of common classification problems was thus kept updated adaptively while processing a text stream; that is, classification problems are added when they will be useful, while they are removed when they will be useless, so that the number of common classification problems, $|\mathcal{X}_c|$, does not exceed a pre-defined threshold, k.

Algorithm 1 depicts the proposed self-adaptive classifier for enumerating common classification problems from an input classification problem, $\boldsymbol{x}$. To incrementally construct common classification problem $\boldsymbol{x}_c$ (Line 4-5), the algorithm extracts the primitive features (ϕ'_i) included in $\boldsymbol{x}$ one by one according to their probability of appearing in the training data of the classifier. When the resulting $\boldsymbol{x}_c$ is included in the current set of common classification problems, $\mathcal{X}_c$, stored margin $M_{\boldsymbol{x}_c}$ is reused. Otherwise, margin $M_{\boldsymbol{x}_c} = m(\boldsymbol{x}_c; \boldsymbol{x}_c^{old}, \boldsymbol{\phi}, \boldsymbol{w})$ is computed by using Eq. 4 (Line 7), and $\boldsymbol{x}_c$ is registered in $\mathcal{X}_c$ as a new common classification problem (Line 10).

An important issue is how to define function USELESS, which selects a common classification problem that will not contribute to speeding up the forthcoming classification, when the number of common classification problems, $|\mathcal{X}_c|$, reaches the pre-defined threshold k. To address this issue, the following two policies (designed originally for CPU caching) are proposed and compared in terms of the efficiency of the classifier in experiments:

Least Frequently Used (LFU) This policy counts frequency of common classification problems in a seen text stream, and it maintains only the top-k common classification problems by removing the least-common classification problem from $\mathcal{X}_c$:

$$\text{USELESS}_{\text{LFU}}(\mathcal{X}_c) = \underset{\boldsymbol{x}_c \in \mathcal{X}_c}{\text{argmin}}\ \text{FREQ}(\boldsymbol{x}_c) \tag{5}$$

[1] More precisely, a stream of classification problems generated during the analysis of a text stream.

A space-saving algorithm, (Metwally et al., 2005), is used to efficiently count the approximated frequency of k classification problems at most and to remove the common classification problem rejected by the space-saving algorithm.

Least Recently Used (LRU) When the volume of flow in a text stream rapidly increases, it is likely to relate to a burst of a certain topic. To exploit this characteristics, this policy preserves k common classification problems whose results are most recently reused:

$$\text{USELESS}_{\text{LRU}}(\mathcal{X}_c) = \operatorname*{argmin}_{\boldsymbol{x}_c \in \mathcal{X}_c} \text{TIME}(\boldsymbol{x}_c) \tag{6}$$

Common classification problems are associated with the last timing when their results are reused, and the least-recently-reused common classification problem is removed when $|\mathcal{X}_c| = k$. To realize this policy, a circular linked-list of size k is used to maintain precomputed results, and the oldest element is just overwritten while the corresponding classification problem is removed.

Fixed threshold k is used throughout the processing of a text stream, and its impact on classification speed was evaluated by experiments.

Since Algorithm 1 naively constructs common classification problems using all the active primitive features in input classification problem $\boldsymbol{x}$, it might repeatedly add and remove classification problems that include rare primitive features such as lexical features. This will incur serious overhead costs. To avoid this situation, the margin computation is terminated as soon as it is determined that the remaining computation does not change the sign of margin (namely, classification label y) of $\boldsymbol{x}$.

When $\boldsymbol{x}$ and $\boldsymbol{x}_c$ are given, lower- and upper-bounds of $m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w})$ can be computed by accumulating bounds of a partial margin computed by adding remaining active primitive features, $\{\phi'_j \in \boldsymbol{x} \mid \phi'_j \notin \boldsymbol{x}_c\}$, one by one to $\boldsymbol{x}_c$. It is assumed that primitive feature ϕ'_i is newly activated in $\boldsymbol{x}_c$ and $\boldsymbol{x}_c^{old}$ refers to $\boldsymbol{x}_c$ without ϕ'_i being activated. The partial margin, $m(\boldsymbol{x}_c; \boldsymbol{\phi}, \boldsymbol{w}) - m(\boldsymbol{x}_c^{old}; \boldsymbol{\phi}, \boldsymbol{w})$, is computed by summing up the weights of primitive feature ϕ'_i and conjunctive features that are composed of ϕ'_i and one or more primitive features $\phi'_j \in \boldsymbol{x}_c^{old}$. This partial margin is upper- and lower-bounded as follows:

$$m(\boldsymbol{x}_c; \boldsymbol{\phi}, \boldsymbol{w}) - m(\boldsymbol{x}_c^{old}; \boldsymbol{\phi}, \boldsymbol{w}) \geq \max(w_i^{min} | \{\phi_j \in \boldsymbol{x}_c \mid \phi_j \notin \boldsymbol{x}_c^{old}\}|, W_i^-) \tag{7}$$

$$m(\boldsymbol{x}_c; \boldsymbol{\phi}, \boldsymbol{w}) - m(\boldsymbol{x}_c^{old}; \boldsymbol{\phi}, \boldsymbol{w}) \leq \min(w_i^{max} | \{\phi_j \in \boldsymbol{x}_c \mid \phi_j \notin \boldsymbol{x}_c^{old}\}|, W_i^+), \tag{8}$$

where w_i^{min} and w_i^{max} refer to minimum and maximum weights among all the features regarding ϕ'_i, while W_i^+ and W_i^- refer to summations of all the features regarding ϕ'_i with positive and negative weights, respectively; that is, this upper or lower-bound is computed by assuming all the features regarding ϕ'_i to have a maximum or minimum weight (each bounded by W_i^+ or W_i^-). Accumulating these bounds for each remaining primitive feature makes it possible to obtain the bounds of $m(\boldsymbol{x}; \boldsymbol{\phi}, \boldsymbol{w})$ and thereby judge whether the sign of the margin can be changed by processing the remaining features.

4.2 Maintaining common classification problems with dynamic double-array trie

To maintain the enumerated common classification problems, a double-array trie (Aoe, 1989) is applied. The trie associates common classification problem $\boldsymbol{x}_c$ with unique index $i (1 \leq i \leq k)$, which is further associated with computed margin $M_{\boldsymbol{x}_c}$ and frequency or access time as described in Section 4.1. Although a double-array trie provides an extremely fast look-up, it had been considered that update operation (adding a new key to a double-array trie) is slow. However, in a recent study (Yata et al., 2009), the update speed of a double-array trie approaches that of a hash table. In the following section, a double-array trie similar to that of Yata et al. (2009) is used to maintain common classification problems.

Efficient dynamic double-array trie with deletion

A double-array trie (Aoe, 1989) and an algorithm that allows a fast update (Yata et al., 2009) are briefly introduced in the following. A double array is a data structure for a compact trie, which consists of two one-dimensional arrays called BASE and CHECK. In a double-array trie, each trie node occupies one element in BASE and CHECK, respectively.[2] For each node, p, BASE stores the offset address of its child

[2] Although the original double-array (Aoe, 1989) realizes a minimal-prefix trie by using another array (called TAIL) to store suffix nodes with only one child, TAIL is not adopted here since it is difficult to support space-efficient deletion with TAIL.

nodes, so a child node takes the address $c = \text{BASE}[p]$ XOR[3] l when the node is traversed from p by label l. For each node, c, CHECK stores the address of its parent node, p, and is used to confirm the validity of the traversal by checking whether $\text{CHECK}[c] = p$ is held after the node is reached by $c = \text{BASE}[p]$ XOR l.

Adding a new node to a trie could cause a conflict, meaning that the newly added node could be assigned to the address taken by an existing node in the trie. In such a case, it is necessary to collect all the sibling nodes of either the newly added node or the existing node that took the conflicting address, and then relocate either branching (with a lower number of child nodes) to empty addresses that are not taken by other nodes in the trie. This relocation is time-consuming, and is the reason for the slow update.

To perform this relocation quickly, Yata et al. (2009) introduced two additional one-dimensional arrays, called NLINK (node link) and BLOCK. For each node, NLINK stores the label needed to reach its first child and the label needed to reach from its parent the sibling node next to the node. It thereby makes it possible to quickly enumerate the sibling nodes for relocation. BLOCK stores information on empty addresses within each 256 consecutive addresses called a block[4] in BASE and CHECK. Each block is classified into three types, called "full," "closed" and "open." Full blocks have no empty addresses and are excluded from the target of relocation. Closed blocks have only one empty address or have failed to be relocated more times than a pre-specified threshold. Open blocks are other blocks, which have more than one empty address. The efficient update of the trie is enabled by choosing appropriate blocks to relocate a branching; a branching with one child node is relocated to a closed block, while a branching with multiple child nodes is relocated to an open block.

The above-described double-array trie was modified to support a deletion operation, which simply registers to each block empty addresses resulting from deletion. In consideration that a new key (common classification problem) will be stored immediately after the deletion (Line 10 in Algorithm 1), the double-array trie is not packed as in Yata et al. (2007) after a key is deleted.

Engineering a double-array trie to reduce trie size

To effectively maintain common classification problems in a trie, it is critical to reduce the number of trie nodes accessed in look-up, update, and deletion operations. The number of trie nodes was therefore reduced as much as possible by adopting a more compact representation of keys (common classification problems) and by elaborating the way to store values for the keys in the double-array trie.

Gap-based key representation To compress representations of common classification problems (feature sequences) in the trie, frequency-based indices are allocated to primitive features (Yoshinaga and Kitsuregawa, 2010). A gap representation (used to compress posting lists in information retrieval (Manning et al., 2008, Chapter 5)) is used to encode feature sequences. Each feature index is replaced with a gap from the preceding feature index (the first feature index is used as is). Each gap is then encoded by variable-byte coding (Williams and Zobel, 1999) to obtain shorter representations of feature sequences.

A reduced double-array trie The standard implementation of a double-array trie stores an (integer) index with a key at a child node (value node) traversed by a terminal symbol `'\0'` (or an alphabet not included in a key, *e.g.*, `'#'`) from the node reached after reading the entire key (Yoshinaga and Kitsuregawa, 2009; Yasuhara et al., 2013). However, when a key is not a prefix to the other keys, the value node has no sibling node, so a value can be directly embedded on the BASE of the node reached after reading the entire key instead of the offset address of the child (value) node. All the value nodes for the longest prefixes are thereby eliminated from the trie. The resulting double-array trie is referred to as a *reduced double-array trie*.

These two tricks reduce the number of trie nodes (memory usage), and make the trie operations faster. A reduced double-array trie is also used to compactly store the weights of conjunctive features, as described in Yoshinaga and Kitsuregawa (2010). Interested readers may refer to **cedar**,[5] open-source software of a dynamic double-array trie, for further implementation details of the reduced double-array trie.

[3]The original double-array (Aoe, 1989) uses addition instead of XOR operation to obtain a child address.

[4]Note that the XOR operation guarantees that all the child nodes are located within a certain block i (assuming 1 byte (0-255) for each label, l, child nodes of a node, p, are all located in addresses ($256i \leq c = \text{BASE}[p]$ XOR $l < 256(i+1)$).

[5]`http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/cedar/`

	base-phrase chunking	dependency parsing
Number of features $\|\phi\|$	645,951	2,084,127
Number of primitive features $\|\phi'\|$	11,509	27,063
Accuracy (partial)	99.01%	92.23%
Accuracy (complete)	94.16%	58.38%

Table 1: Model statistics for base-phrase chunking and dependency parsing.

5 Experiments

The proposed self-adaptive classifier was experimentally evaluated by applying it to streams of classification problems. The streams of classification problems were generated by processing Twitter streams using a state-of-the-art base-phrase chunker and dependency parser. All experiments were conducted with an Intel® Core™ i7-3720QM 2.6-GHz CPU server with 16-GB main memory.

5.1 Setup

Since March 11, 2011 (the day of the Great East Japan Earthquake; "3.11 earthquake" hereafter), Twitter streams were crawled by using Twitter API.[6] Tweets from famous Japanese users were crawled first. Next, timelines of those users were obtained. Then, the set of users were repeatedly expanded by tracing retweets and mentions in their timelines to collect as many tweets as possible. In the following experiments, two sets of 24-hour Twitter streams from the crawled tweets were used. The first Twitter stream was taken from the day of 3.11 earthquake (12:00 on Friday, March 11, 2011 to 12:00 on Saturday, March 12, 2011), and the second one was taken from the second weekend in March, 2012 (12:00 on Friday, March 9, 2012 to 12:00 on Saturday, March 10, 2012). The first Twitter stream is intended to evaluate the classifier performance on days with a significant, continuous burst, while the second one is to evaluate the performance on days without such a burst. No special events, other than a small earthquake (02:25 on March 10), occurred from March 9 to 10, 2012. Because the input to base-phrase chunking and dependency parsing is a sentence, each post was split by using punctuations as clues.

Although it might be better to evaluate the chunking and parsing speed with the proposed classifier for a text stream, the classification speed was evaluated for streams of classification problems generated in processing the Twitter streams by a deterministic base-phrase chunker (Sassano, 2008) and a shift-reduce dependency parser (Sassano, 2004), which are implemented in J.DepP.[12] Note that the chunker and parser are known to spend most of the time for classification (Yoshinaga and Kitsuregawa, 2012), and reducing the classification time leads to efficient processing of Twitter streams.

The base-phrase chunker processes each token in a sentence identified by a morphological analyzer, MeCab,[7] and judges whether the token is the beginning of a base-phrase chunk in Japanese (called a *bunsetsu*[8]) or not. The shift-reduce dependency parser processes each chunk in the chunked sentences and determines whether the head candidate chosen by the parser is correct head or not.

The classifiers for base-phrase chunking and dependency parsing were trained by using a variant of a passive-aggressive algorithm (PA-I) (Crammer et al., 2006) with a standard split[9] of the Kyoto-University Text Corpus (Kawahara et al., 2002) Version 4.0.[10] A third-order polynomial kernel was used to consider combinations of up-to three primitive features. The features used for training the classifiers were identical to those implemented in J.DepP. The polynomial kernel expanded (Kudo and Matsumoto, 2003) was used to make the number of resulting conjunctive features tractable without harming the accuracy.

Table 1 lists the statistics of the models trained for chunking and parsing. In Table 1, "accuracy (partial)" is the ratio of chunks (or dependency arcs) correctly identified by the chunker (or the parser), while "accuracy (complete)" is the exact-match accuracy of complete chunks (or dependency arcs) in a sentence. The accuracy of the resulting parser on the standard split was better than any published results

[6] https://dev.twitter.com/docs/api

[7] http://mecab.sourceforge.net/

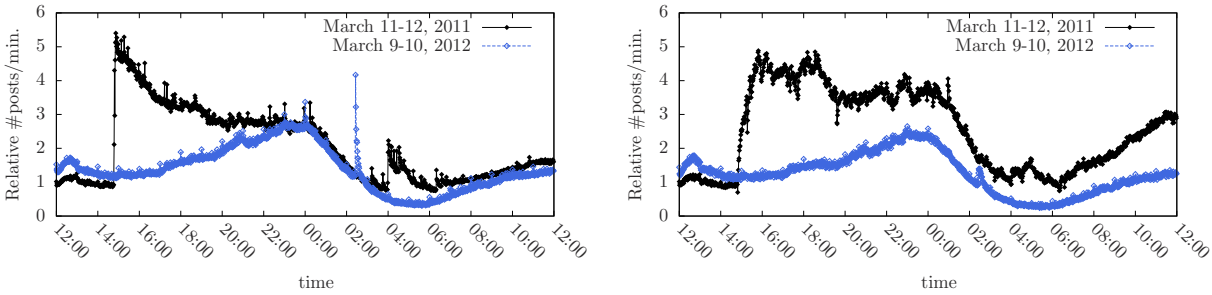
[8] A *bunsetsu* is a linguistic unit consisting of one or more content words followed by zero or more function words.

[9] 24,263, 4,833 and 9,284 sentences (234,685, 47,571 and 89,874 base phrases) for training, development, and testing.

[10] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?Kyoto%20University%20Text%20Corpus

	March 11-12, 2011	March 9-10, 2012
Number of posts	9,291,767	6,360,392
Number of posts/s	108	74
Number of sentences	24,722,596	13,521,196
Number of classification problems (chunking)	220,490,401	109,452,133
Number of classification problems (parsing)	70,096,105	34,380,385

Table 2: Twitter stream used for evaluation.



(a) Change in number of posts per min.

(b) Change in number of classifications per min. (parsing)

Figure 1: Volume of flow of Twitter streams from March 11 to 12, 2011 and from March 9 to 10, 2012.

method	March 11-12, 2011 space [MiB]	speed [ms/sent.]	ratio	March 9-10, 2012 space [MiB]	speed [ms/sent.]	ratio
baseline	12.01	0.0221	1.00	12.01	0.0188	1.00
Y&K '09	30.46	0.0118	1.87	30.46	0.0112	1.69
proposed $k = 2^{16}$	18.05	0.0092	2.40	17.93	0.0098	1.93
(LFU) $k = 2^{20}$	90.70	0.0088	2.51	90.78	0.0089	2.12
$k = 2^{24}$	463.04	0.0081	2.73	473.60	0.0076	2.48
proposed $k = 2^{16}$	17.32	0.0086	2.57	17.32	0.0093	2.02
(LRU) $k = 2^{20}$	85.89	0.0077	2.88	86.09	0.0085	2.22
$k = 2^{24}$	399.17	**0.0070**	**3.16**	409.59	**0.0068**	**2.76**

Table 3: Experimental results obtained with the reduced double array trie: base phrase chunking.

for this dataset other than those reported for a parser based on "stacking" (Iwatate, 2012).[11]

Table 2 lists the detail of the Twitter streams used for evaluating the proposed classifier. Figures 1(a) and 1(b) show the change in the number of posts and classifications for parsing per minute, when the average number of posts and classifications per minute before the 3.11 earthquake is counted as one, respectively. The dataset shows a rapid growth in the number of posts after the 3.11 earthquake occurred (14:46:18). This event also incurs a rapid growth in the number of classifications for parsing. Although space limitations precluded the number of classifications for chunking, it had the same tendency as for parsing. It should be noted that the official retweets (reposts) occupied 25.8% (2,394,025) and 8.5% (542,726) of the entire posts from March 11 to 12, 2011 and from March 9 to 10, 2012, respectively.

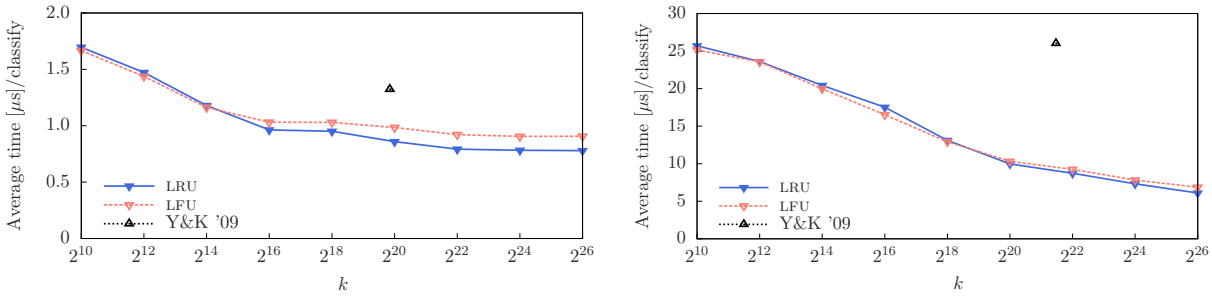
5.2 Results

Tables 3 and 4 list the timings needed to solve the classification problems generated for each sentence by processing the Twitter streams listed in Table 2 using the base-phrase chunker and the dependency parser, respectively. In Table 3, "baseline" refers to the classifier using Eq. 3, while "Y&K '09" refers to Yoshinaga and Kitsuregawa (2009) who used Eq. 4, and enumerates common classification problems from the training corpus[9] of the classifier in advance. To highlight the performance gap caused by the algorithmic differences and make the memory consumptions comparable, the experiments were conducted using the same implementation of the reduced double-array trie, described in Section 4.2, for all the methods. The proposed classifier (with 65,536 ($k = 2^{16}$) common classification problems) achieved higher classifica-

[11] The best reported accuracy of a non-stacking parser is 91.96% (partial) and 57.44% (complete) for Kyoto-University Text Corpus Version 3.0 (Iwatate et al., 2008), and is better than that achieved by the MST algorithm (McDonald et al., 2005).

method	March 11-12, 2011 space [MiB]	speed [ms/sent.]	ratio	March 9-10, 2012 space [MiB]	speed [ms/sent.]	ratio
baseline	31.50	0.1187	1.00	31.50	0.0979	1.00
Y&K '09	99.91	0.0738	1.61	99.91	0.0651	1.51
proposed $k = 2^{16}$	43.21	0.0469	2.53	43.01	0.0542	1.81
(LFU) $k = 2^{20}$	113.40	0.0293	4.06	113.27	0.0399	2.45
$k = 2^{24}$	904.32	0.0222	5.35	905.62	0.0285	3.44
proposed $k = 2^{16}$	42.68	0.0497	2.39	42.66	0.0546	1.79
(LRU) $k = 2^{20}$	108.88	0.0283	4.20	108.94	0.0421	2.32
$k = 2^{24}$	840.85	**0.0208**	**5.71**	840.93	**0.0280**	**3.50**

Table 4: Experimental results obtained with the reduced double array trie: dependency parsing.



(a) base-phrase chunking

(b) dependency parsing

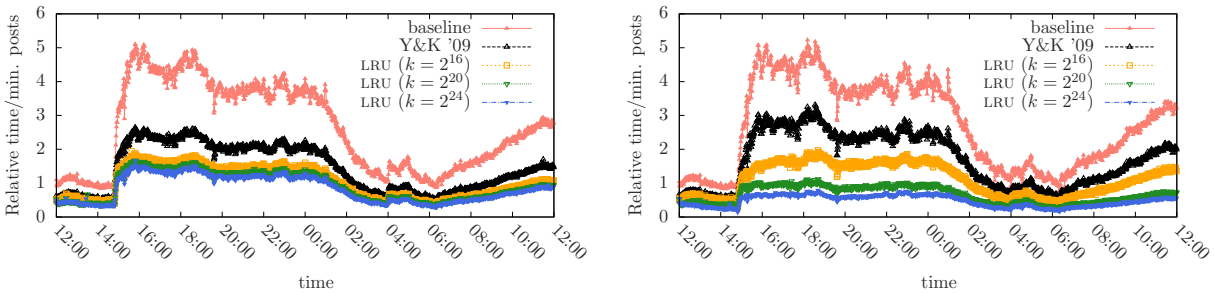
Figure 2: Average classification time per classification problem as a function of number of common classification problems k (2011 tweet stream).

tion speed than that achieved by Y&K '09 (with 943,864 (chunking) and 2,902,679 (parsing) common classification problems). Although the speed up is evident for both tweet datasets, the speed-up is more obvious in the case of the 2011 tweet stream. In the following experiments, in view of space limitations and redundancy, the 2011 tweet stream was used; however, note that the same conclusions are drawn from the results with the 2012 twitter stream.

Figure 2 shows the time needed for solving each classification problem for chunking and parsing of the 2011 tweet stream when the threshold to the number of common classification problems k is varied from 2^{10} to 2^{26}, respectively. In both tasks, the proposed classifier with the LRU policy outperforms the proposed classifier with the LFU policy when k was increased. This is not only because the LFU policy has higher overheads than the LRU policy but also because the LFU policy selects useless classification problems that include lexical features related to a burst in the past. The speed-up is saturated in the case of base-phrase chunking at $k = 2^{22}$ (Figure 2(a)). This is because the proposed classifier often terminates margin computation without seeing lexical features for base phrase chunking, so it rarely reuses results of common classification problems including lexical features that are preserved when k is increased. On the other hand in dependency parsing, the classifier relies on lexical features to resolve semantic ambiguities, so it cannot terminate margin computation without seeing lexical features and thus exploits common classification problems including lexical features.

Figure 3 shows the change in the time needed for solving classification problems generated from a one-minute text stream for chunking and parsing of the 2011 tweet stream. The y-axis shows the relative classification time, when the average classification time of the baseline method before the 3.11 earthquake is counted as one. The classification time of the baseline method and Yoshinaga and Kitsuregawa (2009)'s method rapidly increased in response to the increase of the number of classification problems, while the proposed classifier suppressed the increase in classification time. It is thus concluded that the proposed classifier is more robust in terms of real-time processing for a text stream.

Finally, the contributions of the three tricks of the proposed classifier to the classification performance for dependency parsing were evaluated. The three tricks are a gap-based key representation and a reduced double-array trie (Section 4.1), as well as the early termination of margin computation (Section 4.1).



(a) base-phrase chunking

(b) dependency parsing

Figure 3: Change in classification time of one-minute posts (2011 tweet stream).

method	plain (no tricks) space [MiB]	plain (no tricks) speed [ms/sent.]	plain (no tricks) ratio	+ gap-based key space [MiB]	+ gap-based key speed [ms/sent.]	+ gap-based key ratio	+ reduced double array space [MiB]	+ reduced double array speed [ms/sent.]	+ reduced double array ratio	+ early termination space [MiB]	+ early termination speed [ms/sent.]	+ early termination ratio
baseline	39.88	0.1413	0.84	n/a	n/a	n/a	31.50	0.1187	1.00	38.21	0.0745	1.59
Y&K '09	117.66	0.0922	1.29	110.52	0.0904	1.31	99.91	0.0738	1.61	106.61	0.0406	2.93
proposed $k = 2^{16}$	44.64	0.1037	1.14	44.50	0.1009	1.18	36.09	0.0845	1.41	42.68	0.0497	2.39
(LRU) $k = 2^{20}$	117.21	0.0590	2.01	114.57	0.0572	2.07	105.21	0.0492	2.41	108.88	0.0283	4.20
$k = 2^{24}$	969.48	0.0412	2.88	923.96	0.0398	2.98	897.70	0.0350	3.39	840.85	**0.0208**	**5.71**

Table 5: Contribution of each trick to classification performance; 2011 tweet dataset (underlined numbers are quoted from Table 4).

Table 5 lists the classification times per sentence in the case of dependency parsing, when each trick is cumulatively applied to plain classifiers without all the tricks. Classification is significantly speeded up by early termination of the margin computation and the reduced double-array trie. These tricks also contribute to speeding up the baseline method and Yoshinaga and Kitsuregawa (2009)'s method.

6 Conclusion

Aiming to efficiently process a real-world text stream (such as a Twitter stream) in real-time, a self-adaptive classifier that becomes faster for a given text stream is proposed. It enumerates common classification problems that are generated during the processing of a text stream, and reuses the results of those classification problems as partial results for solving forthcoming classification problems.

The proposed classifier was evaluated by applying it to the streams of classification problems generated by processing two sets of Twitter streams on the day of the 2011 Great East Japan Earthquake and the second weekend in March 2012 using a state-of-the-art base-phrase chunker and dependency parser. The proposed classifier speeds up the classification by factors of 3.2 (chunking) and 5.7 (parsing), which are significant factors in regard to processing a massive text stream.

It is planned to evaluate the classifier on other NLP tasks. A linear classifier with conjunctive features is widely used for NLP tasks such as word segmentation, part-of-speech tagging (Neubig et al., 2011b), and dependency parsing (Nivre and McDonald, 2008). Even for NLP tasks in which structured classification is effective (*e.g.*, named entity recognition), structure compilation (Liang et al., 2008) (or "uptraining" (Petrov et al., 2010)) gives state-of-the-art accuracy when a linear classifier with many conjunctive features is used. The proposed classifier is expected to be applied to a range of NLP tasks.

All the codes have been available for the research community as open-source software, including **pecco** (a self-adaptive classifier)[12] and **J.DepP** (a base-phrase chunker and dependency parser).[13]

Acknowledgments

This work was supported by the Research and Development on Real World Big Data Integration and Analysis program of the Ministry of Education, Culture, Sports, Science, and Technology, JAPAN.

[12] http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/pecco/
[13] http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/jdepp/

References

Jun'ichi Aoe. 1989. An efficient digital search algorithm by using a double-array structure. *IEEE Transactions on Software Engineering*, 15(9):1066–1077.

Eiji Aramaki, Sachiko Maskawa, and Mizuki Morita. 2011. Twitter catches the flu: Detecting influenza epidemics using Twitter. In *Proceedings of EMNLP*, pages 1568–1576.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *JMLR*, 7:551–585, March.

Jennifer Foster, Özlem Çetinoglu, Joachim Wagner, Joseph Le Roux, Stephen Hogan, Joakim Nivre, Deirdre Hogan, and Josef van Genabith. 2011. #hardtoparse: POS tagging and parsing the Twitterverse. In *Proceedings of the AAAI-11 Workshop on Analyzing Microtext*.

Eric Gilbert and Karrie Karahalios. 2010. Widespread worry and the stock market. In *Proceedings of ICWSM*, pages 58–65.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In *Proceedings of ACL-HLT*, pages 42–47.

Yoav Goldberg, Kai Zhao, and Liang Huang. 2013. Efficient implementation of beam-search incremental parsers. In *Proceedings of ACL, Short Papers*, pages 628–633.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of ACL-HLT*, pages 368–378.

Hideki Isozaki and Hideto Kazawa. 2002. Efficient support vector classifiers for named entity recognition. In *Proceedings of COLING*, pages 1–7.

Masakazu Iwatate, Masayuki Asahara, and Yuji Matsumoto. 2008. Japanese dependency parsing using a tournament model. In *Proceedings of COLING*, pages 361–368.

Masakazu Iwatate. 2012. *Development of Pairwise Comparison-based Japanese Dependency Parsers and Application to Corpus Annotation*. Ph.D. thesis, Graduate School of Information Science, Nara Institute of Science and Technology.

Nobuhiro Kaji, Yasuhiro Fujiwara, Naoki Yoshinaga, and Masaru Kitsuregawa. 2010. Efficient staggered decoding for sequence labeling. In *Proceedings of ACL*, pages 485–494.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of LREC*, pages 2008–2013.

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of EMNLP*, pages 1288–1298.

Taku Kudo and Yuji Matsumoto. 2003. Fast methods for kernel-based text analysis. In *Proceedings of ACL*, pages 24–31.

Gourab Kundu, Vivek Srikumar, and Dan Roth. 2013. Margin-based decomposed amortized inference. In *Proceedings of EMNLP*, pages 905–913.

Percy Liang, Hal Daumé III, and Dan Klein. 2008. Structure compilation: trading structure for features. In *Proceedings of ICML*, pages 592–599.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Twitter, Inc. 2011. Twitter's 2011 year in review. `http://yearinreview.twitter.com/en/tps.html`.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 523–530.

Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. 2005. Efficient computation of frequent and top-k elements in data streams. In *Proceedings of ICDT*, pages 398–412.

Graham Neubig, Yuichiroh Matsubayashi, Masato Hagiwara, and Koji Murakami. 2011a. Safety information mining - what can NLP do in a disaster -. In *Proceedings of IJCNLP*, pages 965–973.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011b. Pointwise prediction for robust, adaptable japanese morphological analysis. In *Proceedings of ACL*, pages 529–533.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-HLT*, pages 950–958.

Slav Petrov, Pi-Chuan Chang, Michael Ringgaard, and Hiyan Alshawi. 2010. Uptraining for accurate deterministic question parsing. In *Proceedings of EMNLP*, pages 705–713.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of EMNLP*, pages 1524–1534.

Alexander Rush and Slav Petrov. 2012. Vine pruning for efficient multi-pass dependency parsing. In *Proceedings of NAACL-HLT*, pages 498–507.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: Real-time event detection by social sensors. In *Proceedings of WWW*, pages 851–860.

Manabu Sassano. 2004. Linear-time dependency analysis for Japanese. In *Proceedings of COLING*, pages 8–14.

Manabu Sassano. 2008. An experimental comparison of the voted perceptron and support vector machines in Japanese analysis tasks. In *Proceedings of IJCNLP*, pages 829–834.

Vivek Srikumar, Gourab Kundu, and Dan Roth. 2012. On amortizing inference cost for structured prediction. In *Proceedings of EMNLP-CoNLL*, pages 1114–1124.

Andranik Tumasjan, Timm O. Sprenger, Philipp G. Sandner, and Isabell M. Welpe. 2010. Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proceedings of ICWSM*, pages 178–185.

Gertjan van Noord. 2009. Learning efficient parsing. In *Proceeding of EACL*, pages 817–825.

István Varga, Motoki Sano, Kentaro Torisawa, Chikara Hashimoto, Kiyonori Ohtake, Takao Kawai, Jong-Hoon Oh, and Stijn De Saeger. 2013. Aid is out there: Looking for help from tweets during a large scale disaster. In *Proceedings of ACL*, pages 1619–1629.

Henning Wachsmuth, Benno Stein, and Gregor Engels. 2011. Constructing efficient information extraction pipelines. In *Proceedings of CIKM*, pages 2237–2240.

Henning Wachsmuth, Benno Stein, and Gregor Engels. 2013. Learning efficient information extraction on heterogeneous texts. In *Proceedings of IJCNLP*, pages 534–542.

Xiaofeng Wang, Matthew S. Gerber, and Donald E. Brown. 2012. Automatic crime prediction using events extracted from Twitter posts. In *Proceedings of SBP*, pages 231–238.

Hugh E. Williams and Justin Zobel. 1999. Compressing integers for fast file access. *The Computer Journal*, 42(3):193–201.

Makoto Yasuhara, Toru Tanaka, Jun ya Norimatsu, and Mikio Yamamoto. 2013. An efficient language model using double-array structures. In *Proceedings of EMNLP*, pages 222–232.

Susumu Yata, Masaki Oono, Kazuhiro Morita, Masao Fuketa, and Jun ichi Aoe. 2007. An efficient deletion method for a minimal prefix double array. *Journal of Software: Practice and Experience*, 37(5):523–534.

Susumu Yata, Masahiro Tamura, Kazuhiro Morita, Masao Fuketa, and Jun'ichi Aoe. 2009. Sequential insertions and performance evaluations for double-arrays. In *Proceedings of the 71st National Convention of IPSJ*, pages 1263–1264. (In Japanese).

Naoki Yoshinaga and Masaru Kitsuregawa. 2009. Polynomial to linear: Efficient classification with conjunctive features. In *Proceedings of EMNLP*, pages 1542–1551.

Naoki Yoshinaga and Masaru Kitsuregawa. 2010. Kernel slicing: Scalable online training with conjunctive features. In *Proceedings of COLING*, pages 1245–1253.

Naoki Yoshinaga and Masaru Kitsuregawa. 2012. Efficient classification with conjunctive features. *Journal of Information Processing*, 20(1):228–227.

A Dependency Edge-based Transfer Model for Statistical Machine Translation

Hongshen Chen[†§] **Jun Xie**[†] **Fandong Meng**[†§] **Wenbin Jiang**[†] **Qun Liu**[‡†]
†Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences
§University of Chinese Academy of Sciences
{chenhongshen,xiejun,mengfandong,jiangwenbin}@ict.ac.cn
‡CNGL, School of Computing, Dublin City University
qliu@computing.dcu.ie

Abstract

Previous models in syntax-based statistical machine translation usually resort to some kinds of synchronous procedures, few of these works are based on the analysis-transfer-generation methodology. In this paper, we present a statistical implementation of the analysis-transfer-generation methodology in rule-based translation. The procedures of syntax analysis, syntax transfer and language generation are modeled independently in order to break the synchronous constraint, resorting to dependency structures with dependency edges as atomic manipulating units. Large-scale experiments on Chinese to English translation show that our model exhibits state-of-the-art performance by significantly outperforming the phrase-based model. The statistical transfer-generation method results in significantly better performance with much smaller models.

1 Introduction

Researches in statistical machine translation have been flourishing in recent years. Statistical translation methods can be divided into word-based (Brown et al., 1993), phrase-based (Marcu and Wong, 2002; Koehn et al., 2003) and syntax-based models (Yamada and Knight, 2001; Graehl and Knight, 2004; Chiang, 2005; Liu et al., 2006; Mi et al., 2008; Huang et al., 2006; Lin, 2004; Ding and Palmer, 2004; Quirk et al., 2005; Shen et al., 2008; Xie et al., 2011; Meng et al., 2013). Compared with word-based and phrase-based methods, syntax-based models perform better in long distance reordering and enjoy higher generalization capability by leveraging the hierarchical structures in natural languages, and achieve the state-of-the-art performance in these years.

Most syntax-based models (except for Lin (2004)) utilize some kinds of synchronous generation procedures which directly model the structural correspondence between two languages. In contrast, the analysis-transfer-generation methodology in rule-based translation solves the machine translation problem in a more divided scheme, where the processing procedures of analysis, structural transfer and language generation are modeled separately. The analysis-transfer-generation strategy can tolerate higher non-isomorphism between languages if with a more general transformation unit and it can facilitate elaborating engineering of each processing procedure, however, there isn't a statistical transfer model that shows the comparable performance with the current state-of-the-art SMT model so far.

In this paper, we propose a novel statistical analysis-transfer-generation model for machine translation, to integrate the advantages of the transfer-generation scheme and the statistical modeling. The procedures of transfer and generation are modeled on dependency structures with dependency edges as atomic manipulating units. First, the source sentence is parsed by a dependency parser. Then, the source dependency structure is transferred into a target structure by translation rules, which composed of the source and target edges. Last, the target sentence is finally generated from the target edges which are used as intermediate syntactic structures. By directly modeling the edge, the most basic unit in the dependency tree, which definitely describe the modifying relationship and positional relation between words, our model alleviates the non-isomorphic problem and shows the flexibility of reordering.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1103–1113, Dublin, Ireland, August 23-29 2014.

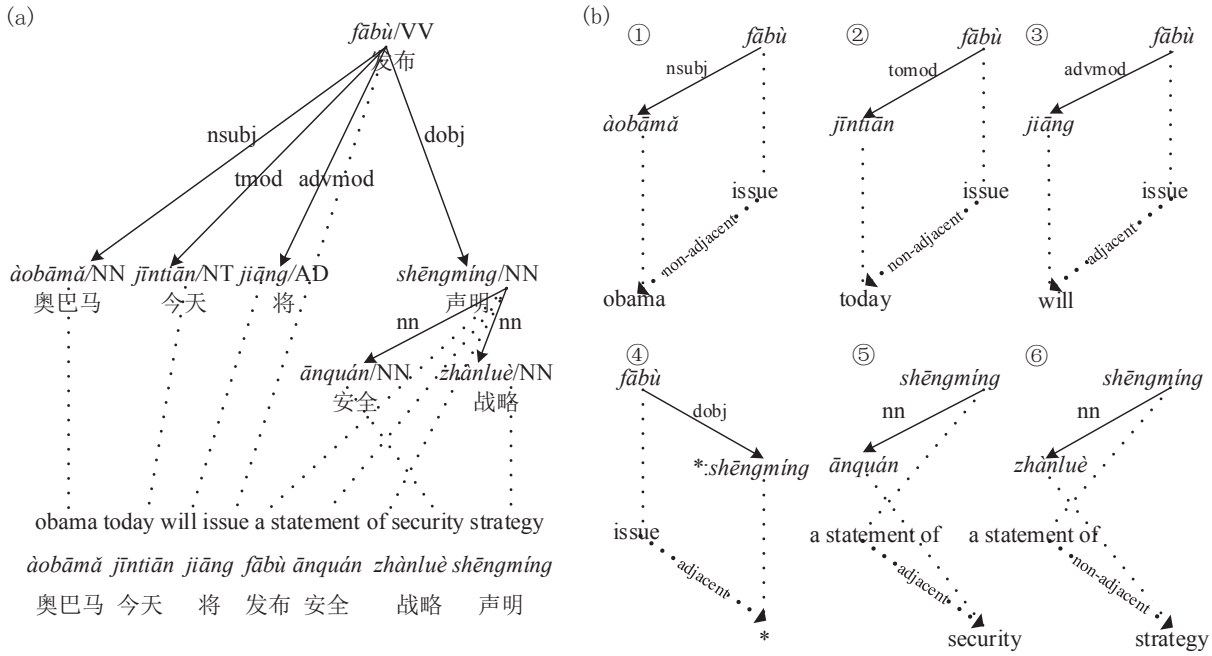


Figure 1: (a)An example of labeled Chinese dependency tree aligned with the corresponding English sentence. (b) Examples of the transfer rules extracted from the tree. "*" denotes a variable. All the inner nodes are treated as variables. The label on the target side of a rule denotes whether the head and the dependent are adjacent or not.

The rest of the paper is organized as follows, we first describe the dependency edge-based transfer model (Section 2). Then, we present our rule acquisition algorithm (Section 3), the decoding and target sentence generation process (Section 4). Finally, large-scale experiments (Section 5) on Chinese-to-English translation show that our edge-based transfer model gains state-of-the-art performance by significantly outperforming the phrase-based model (Koehn et al., 2003) by averaged +1.34 BLEU points on three test sets. To the best of our knowledge, this is the first transfer-generation-based statistical machine translation model that achieves the state-of-the-art performance.

2 Dependency Edge-based Transfer Model

2.1 Edges in Dependency Trees

Given a sentence, its dependency tree is a directed acyclic graph with words in the sentence as nodes. An example dependency tree is shown in Figure 1 (a). An edge in the tree represents a dependency relationship between a pair of words, a head and a dependent. When a nominal dependent acts as a subject and modifies a verbal head, they usually have a fixed relative position. In Figure 1 (a), "*àobāmǎ*" modifies "*fābù*". The grammatical relation label *nsubj* (Chang et al., 2009) between them denotes that a noun phrase acts as the subject of a clause. "*àobāmǎ*" is on the left of "*fābù*".

Based on the above observations, we take the edge as the elementary structure of a dependency tree and regard a dependency tree to be a set of edges.

Definition 1. *An source side* ***edge*** *is a 4-tuple $e = \langle H, D, P, R \rangle$, where H is the head, D is the dependent, P denotes the relative position between H and D, left or right, R is the grammatical relation label*
.

In Figure 1 (b), the upper sides of transfer rules are source side edges extracted from the dependency tree.

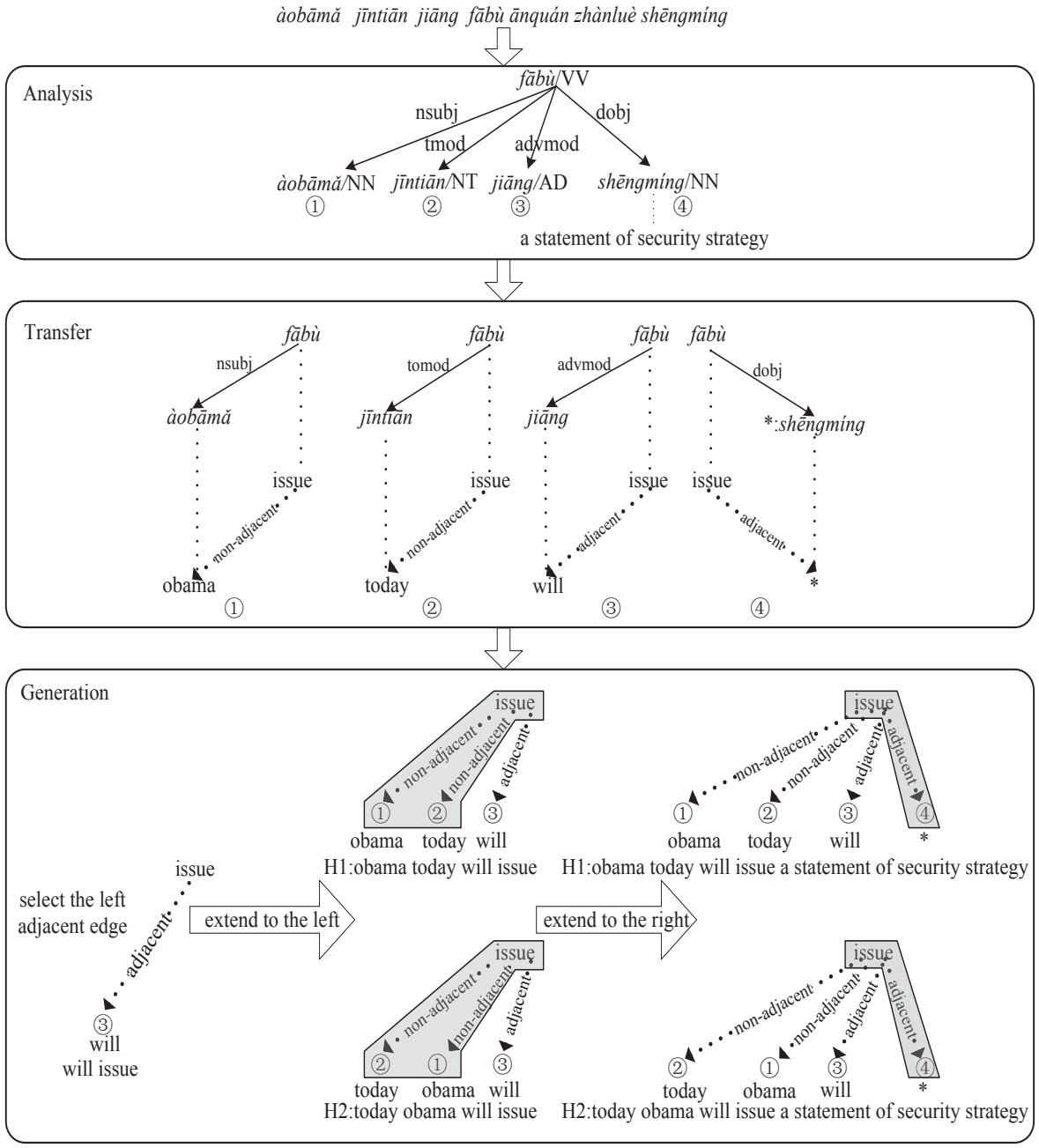


Figure 2: An example partial generation of translation. The same set of rules generate two target hypotheses with the same words and different word order. Assume the sub-tree rooted at "*shēngmíng*" has been translated to the corresponding target sentence fragment.

2.2 Transfer Rules

A transfer rule of our model represents the reordering and relative positions of edges between language pairs. For example, in Figure 1 (b), the first rule shows that when a nominal subject modifies a verb, the target side keeps the same position relations. "obama" is also on the left of "issue", the same with the source side relative position. The 5-th and 6-th rules show the inversion relations between the source and the target. Formally, a transfer rule can be defined as a triple $\langle e, f, \sim \rangle$, where e is an edge extracted from the source dependency tree, f is a target edge. $\sim$ denotes one-to-one correspondence between variables in e and f.

Figure 1 (b) are part of transfer rules extracted from the word aligned sentence in Figure 1 (a). The target edge denotes whether the target dependent is on the left or the right side of the target head, the

label on the edge indicates whether the target head and the target dependent are adjacent or not. If the dependent is an internal node(contrast with the leaf nodes in the dependency tree), then it will be regarded as a substitution node. The dependent in the 4-th transfer rule is an internal node and the its corresponding target side is a substitution variable.

Figure 2 shows a partial transfer-generation of our model which involves three phases. First, *analysis*. Given a source language sentence, we obtain its dependency tree using a dependency parser. We assume that the sub-tree of the substitution node has been translated. Second, *transfer*. For each internal node, we transfer the source side edges between the head and all its dependents into the target sides. In the second block of Figure 2, we transfer four edges into the target sides. Third, *generation*, corresponding to the third block of Figure 2. We generate the target sentence with the target side edges starting from the target head, "issue". We first try to concatenate the edges to the left. First, we select a target side edge that is on the left side of "issue" and adjacent to it to form a consecutive phrase. Edge 3 is selected and "to issue" is generated. Then, we enumerate all possible left concatenations of the other edges that are not adjacent to "issue". The two sequences(1,2,3 and 2,1,3) of the edges are generated, corresponding to the two hypotheses. After that, we extend the two hypotheses to the right. The internal node "*shēngmíng*" is a substitution node, so the candidate translation of the sub-tree rooted at "*shēngmíng*" is concatenated to the two hypotheses. Finally, we generate the two candidate translations of the input sentence.

3 Acquisition of Transfer Rules

Transfer rules can be extracted automatically from a word-aligned corpus, which is a set of triples $\langle T, S, A\rangle$, where T is a source dependency tree, S is a target side sentence and A is an alignment relation between T and S. Following the dependency-to-string model (Xie et al., 2011), we extract transfer rules from each triple $\langle T, S, A\rangle$ by three steps:

1. Tree Annotation: Label each node in the dependency tree with the alignment information
2. Edges Identification: Identify acceptable edges from the annotated dependency tree
3. Rule induction: Induce a set of lexicalized and un-lexicalized transfer rules from the acceptable edges.

3.1 Tree Annotation

Given a triple $\langle T, S, A\rangle$ as Figure 3 shows, we define two attributes for every node in T: node span and sub-tree span:

Definition 2. *Given a node n, its* ***node span*** $nsp(n)$ *is a set of consecutive indexes of the target words aligned with the node n.*

For example, nsp(*ānquán*)={7-8}, which corresponds to the target word "of" and "security".

Definition 3. *A node span $nsp(n)$ is* ***consistent*** *if for any other node n' in the dependency tree, $nsp(n)$ and $nsp(n')$ are not overlapping.*

For example, nsp(*zhànluè*) is consistent, while nsp(*ānquán*) is not consistent for it corresponds to the same word "of" with nsp(*shēngmíng*).

Definition 4. *Given a sub-tree T' rooted at n, the* ***sub-tree span*** $tsp(n)$ *of n is a consecutive target word indexes from the lower bound of the nsp of all the nodes in T' to the upper bound of those spans.*

For example, tsp(*shēngmíng*)={5-9},which corresponds to the target phrase "a statement of security strategy".

Definition 5. *A sub-tree span $tsp(n)$ is* ***consistent*** *if for any other node n' that is not in the sub-tree rooted at n in the dependency tree, $tsp(n)$ and $nsp(n')$ are not overlapping.*

For example, tsp(*shēngmíng*) is consistent, even though nsp(*shēngmíng*) is not consistent, while tsp(*ānquán*) is not consistent for "*shēngmíng*" is not a node in sub-tree rooted at "*ānquán*" and "*ānquán*" corresponds to the same word "*of*" with nsp(*shēngmíng*) .

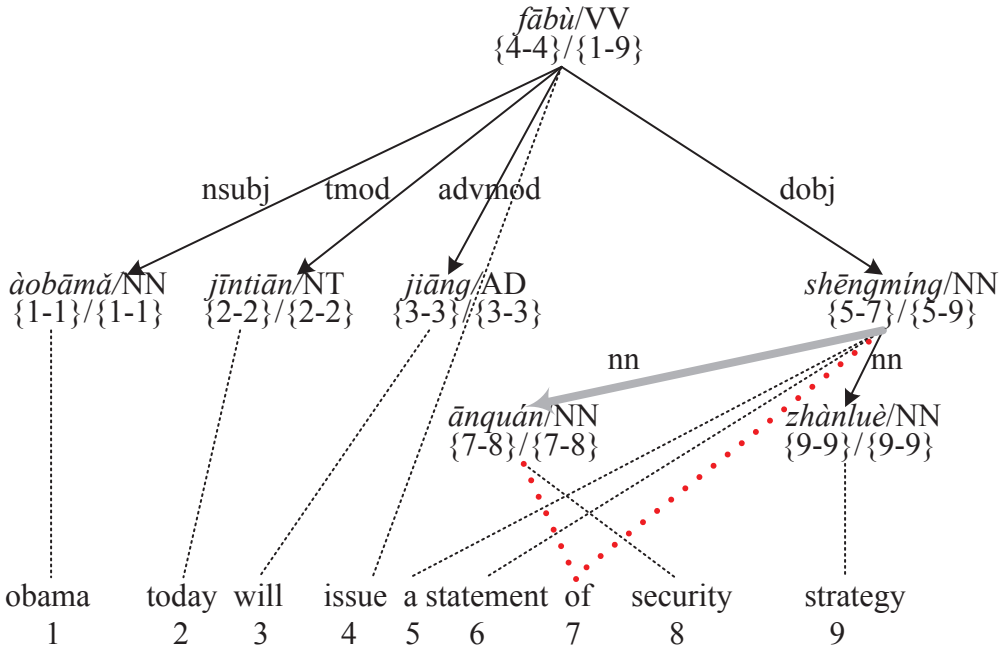


Figure 3: An example of annotated dependency tree. Each node is annotated with two spans, the former is node span and the latter is sub-tree span. The gray edge is not acceptable. It is different from Figure 1, because "ānquán" aligned with two words in Figure 3. "of" in the target side is aligned with both "ānquán" and "shēngmíng" which makes the gray edge un-acceptable.

3.2 Acceptable Edges Identification

We identify the edges from the annotated dependency tree that are **acceptable** for rule induction. For an acceptable edge, its *node span* of the head nsp(head) and the *sub-tree span* of the dependent tsp(dependent) satisfy the following properties:

1. nsp(head) and tsp(dependent) are consistent.
2. nsp(head) and tsp(dependent) are non-overlapping.

For example, tsp(*ānquán*) and nsp(*shēngmíng*) are neither consistent nor non-overlapping. So the gray edge between head "*shēngmíng*" and dependent "*ānquán*" is not an acceptable edge. nsp(*fābù*) and tsp(*shēngmíng*) are consistent and the two spans are non-overlapping. Thus, the edge between head "*fābù*" and dependent "*shēngmíng*" is an acceptable edge.

3.3 Transfer Rule Induction

From each acceptable source side edge, we induce a set of lexicalized and un-lexicalized transfer rules. We induce a lexicalized transfer rule from an acceptable edge by the following procedures:

1. extract the source side edge and mark the internal nodes as substitution sites. This form the input of a transfer rule.

2. extract the position information according to nsp(head) and tsp(dependent), whether they are adjacent or not and whether tsp(dependent) is on the left side or the right side of nsp(head).

In Figure 4, the first transfer rule is lexicalized rule, it is induced from the edge between "*fābù*" and "*àobāmǎ*".

In addition to the lexicalized rules described above, we also generalized the rules by replacing the word in an source side edge with a wild card and the part of speech of the word. For example, the rule in Figure 4 can be generalized in two ways. The generalized versions of the rule apply to "*àobāmǎ*" modifying any verb and "*fābù*" modifying any noun, respectively. The generalized rules are also called

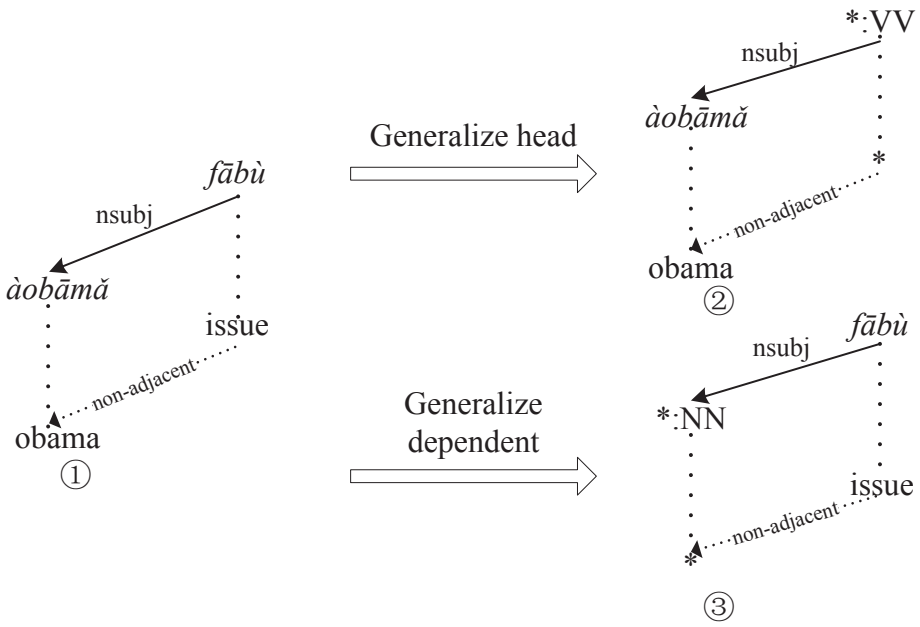


Figure 4: Generalization of transfer rule.

un-lexicalized rules for the loss of word information. The single node translations of the generalized words are also extracted.

The unaligned words of the target side is handled by extending nsp(head) and tsp(dependent) on both left and right directions. We do this process similar with the method of Och and Ney (2004). We might obtain $m(m \geq 1)$ extended rules from an acceptable edge. The frequency of each rule is divided by m. We take the extracted rule set as observed data and make use of relative frequency estimator to obtain the translation probabilities $P(t|s)$ and $P(s|t)$.

4 Decoding and Generation

We follow Och and Ney (2002), using a general log-linear model to score the sentence generated by each concatenation of the target edges. Let c be concatenations that concatenate the target edges to generate the target sentence e. The probability of e is defined as:

$$P(c) \propto \prod_i \phi_i(c)^{\lambda_i} \tag{1}$$

where $\phi_i(c)$ are features defined on concatenations and λ_i are feature weights. In our experiments of this paper, thirteen features are used as follows:

- Transfer rules translation probabilities $P(t|s)$ and $P(s|t)$, and lexical translation probabilities $P_{lex}(t|s)$ and $P_{lex}(s|t)$;
- Bilingual phrases probabilities $P_{bp}(t|s)$ and $P_{bp}(s|t)$, and bilingual phrases lexical translation probabilities $P_{bplex}(t|s)$ and $P_{bplex}(s|t)$;
- Transfer rule penalty $exp(-1)$;
- Bilingual phrase penalty $exp(-1)$;
- Pseudo translation rule penalty $exp(-1)$;
- Target word penalty $exp(|e|)$;
- Language model $P_{lm}(e)$.

Our decoder is based on a bottom-up chart-based beam-search algorithm. We regard the decoding process as the composition of the target side edges. For a given source language sentence, we obtain its

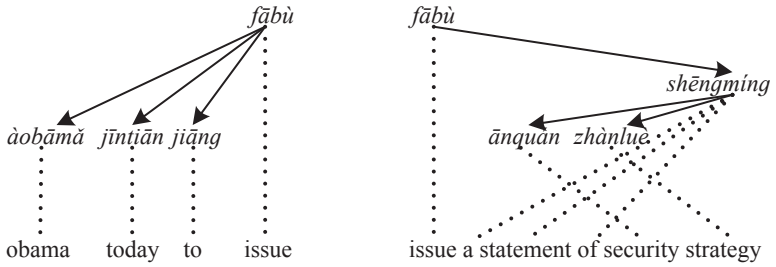


Figure 5: Two examples of the phrases incorporated in our model.

dependency tree T with an external dependency parser. Each node in T is traversed in post-order. For each internal node and root node n, we do the transfer-generation translation as the following procedures:

1. Extract all the source side edges including the lexicalized and generalized edges between n and all its dependents using the same way we extract the source side edges of the transfer rules.
2. *Transfer* the source side edges into target side edges. For a generalized rule, we restore it to a lexicalized rule by combining it with the single word translation. For no matched edges, we construct the pseudo translation rule according to the word order of the source head-dependent relation.
3. *Generate* the target sentence by bi-directional extension from an adjacent target edge. We first group all the target edges by their heads. For each group, we generate translation hypotheses with the following procedures:

   (a) Select an adjacent target edge as the starting position;

   (b) Extend to the left side and enumerate all possible permutations of the target edges directing left;

   (c) Extend to the right side and enumerate all possible permutations of the target edges directing right.

   Considering that in dependency trees, a head may relate to more than 4 edges which results in massive search space. We reduce the time complexity by using the maximum distortion limit. The distortion is defined as $(a_i - b_{i-1} - 1)$, where a_i denotes the start position of the source side edge that is translated into the ith target side edge and b_{i-1} denotes the end position of the source side edge translated into the $(i-1)$th target side edge.

When we reach the root node, the candidate translations of the input sentence are generated.

In our model, only the adjacent target edge of a transfer rule can be regarded as a consecutive phrase and its corresponding source side length is only 2. As we start extending the target sentence from the target head, it is quite natural to incorporate the bilingual phrases to make the target sentences be extended from the phrases as well as the single target head word. Due to the flexibility of our model, we can incorporate not only the syntactic phrases which are phrases covering a whole sub-tree, but also the non-syntactic phrases as the fixed dependency structures in Shen et al. (2008) which are consecutive phrases covering the head. Figure 5 shows two examples of the phrases incorporated in our model.

We prune the search space in several ways. First, beam threshold β, items with a score worse than β times of the best score in the same span will be discarded; second, beam size b, items with a score worse than the bth best item will be discarded. For our experiments, we set $\beta = 10^{-3}$ and $b = 300$; Third, we also prune rules for the same edge with a fixed rule limit ($r = 200$), which denotes the maximum number of rules we keep.

5 Experiments

In this section, the performance of our model is evaluated by comparing with phrase-based model (Koehn et al., 2003), on the NIST Chinese-to-English translation tasks. We also present the influence of the

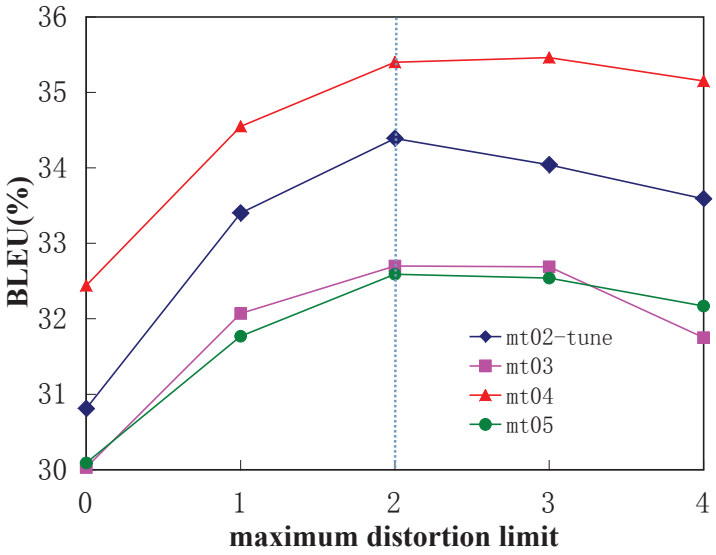


Figure 6: Effect of different maximum distortion limits on development set (mt02) and three tests(mt03,04,05). The performance of all the sets are consistent.

maximum distortion limit to our model. We take open source phrase-based system *Moses* (with default configuration)[1] as our baseline system.

5.1 Experimental Setting

Our training corpus consists of 1.25M sentence pairs from LDC data, including LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

To obtain the dependency trees of the source side, we parse the source sentences with Stanford Parser (Klein and Manning, 2003) into projective dependency structures with nodes annotated by POS tags and edges by dependency labels.

To obtain the word alignments, we run GIZA++ (Och and Ney, 2003) on the corpus in both directions and apply "grow-diag-and" refinement (Koehn et al., 2003). We extract the phrases covering no more than 10 nodes of the fixed structures.

We use SRILM (Stolcke, 2002) to train a 4-gram language model with modified Kneser-Ney smoothing on the Xinhua portion of the Gigaword corpus.

We use NIST MT Evaluation test set 2002 as our development set, 2003-2005 NIST datasets as testsets. The quality of translations is evaluated by the case insensitive NIST BLEU-4 metric[2].

We make use of the minimum error rate training algorithm (Och, 2003) in order to maximize the BLEU score of the development set.

The statistical significance test is performed by *sign-test* (Collins et al., 2005).

5.2 Influence of Maximum Distortion Limit

Figure 6 gives the performance of our system with different maximum distortion limits in terms of uncased BLEU of three NIST test sets. The performance of different distortion limit are consistent on both development set and three test sets. Maximum distortion limit 2 gets the best performances. A low distortion limit may cause the target sentence been translated more close to the sequence of the source, especially when the distortion limit equals to 0, none of the reordering is allowed, while a high distortion limit may lead the good translations be flooded by too many ambiguities when enumerating the possible sequences of the target non-adjacent dependents. We choose 2 as the maximum distortion limit in the next experiments.

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

[2] ftp://jaguar.ncsl.nist.glv/mt/resources/mteval-v11b.pl.

System	Rule #	MT03	MT04	MT05	Average
Moses	44.49M	32.03	32.83	31.81	32.22
DEBT	30.7M	**32.7***	**35.4***	**32.59***	**33.56**

Table 1: Statistics of the extracted rules on training data and the BLEU scores (%) on the test sets. "DEBT" denotes our edge-based transfer model. The "*" denotes that the results are significantly better than the baseline system ($p<0.01$).

5.3 Performance of Our Model

Tabel 1 illustrates the translation results of our experiments. We (*DEBT*) surpass the baseline over +1.34 BLEU points on average. Our model significant outperforms the baseline phrase-based model, with $p < 0.01$ on statistical significance test *sign-test* (Collins et al., 2005).

We also list the statistical number of rules extracted from the training corpus. The number of our transfer rules is only 69.0% of the rules extracted by *Moses*, thus, the total rules in our model is 31% smaller than *Moses*.

6 Related Work

Transfer-based MT systems usually take a parse tree in the source language and translate it into a parse tree in the target language with transfer rules. Both our model and some of those previous works acquired transfer rules automatically from word-aligned corpus (Richardson et al., 2001; Carbonell et al., 2002; Lavoie et al., 2002; Lin, 2004). Gimpel and Smith (2009) and Gimpel and Smith (2014) used quasi-synchronous dependency grammar for MT and they are similar to our idea of doing transfer of dependency syntax in a non-synchronous setting. They do the translation as monolingual lattice parsing.

As dependency-based system, Lin (2004) used path as the transfer unit and regarded the translation problem with minimal path covering. Quirk et al. (2005) and Xiong et al. (2007) used treelets to model the source dependency tree using synchronous grammars. Quirk et al. (2005) projected the source dependency structure into target side by word alignment and faced the problem of non-isomorphism between languages. Xiong et al. (2007) directly modeled the treelet to the corresponding target string to alleviate the problem. Xie et al. (2011) directly specified the ordering information in head-dependents rules that represent the source side as head-dependents relations and the target side as string.

Differently, our model uses a much simpler elementary structure, edge, which consist of only a head and a dependent. As a transfer-generation model, we transfer an edge in the source dependency tree into target side and incorporate the position information on the target edge , which alleviate non-isomorphism problem and incorporate ordering among different target edges simultaneously. Moreover, our decoding method is quite different from previous dependency tree-based works. After parsing a given source language sentence, we transfer and generate the target sentence fragments recursively on each internal node of the dependency tree bottom-up.

7 Conclusions and Future Work

In this paper, we present a novel dependency edge-based transfer model using dependency trees on the source side for machine translation. We directly *transfer* the edges in source dependency tree into the target sides and then *generate* the target sentences by beam-search. With the concise transfer rules, our model is compatible with both the syntactic and non-syntactic phrases. Although the generation process of our model seems relatively simple, it still exhibits a good performance and outperforms the phrase-based model on large scale experiments. For the first time, a statistical transfer model shows a comparable performance with the state-of-the-art translation models.

Since the translation procedure is divided into three phases and each phase can be modeled independently, we would like to take further steps focusing on modeling the target language generation process specifically to ensure a better grammatical translation with the help of natural language generation methods.

Acknowledgments

The authors were supported by National Key Technology R&D Program (No. 2012BAH39B03), CAS Action Plan for the Development of Western China (No. KGZD-EW-501), and Sino-Thai Scientific and Technical Cooperation (No. 60-625J). Sincere thanks to the anonymous reviewers for their thorough reviewing and valuable suggestions.

References

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

Jaime Carbonell, Katharina Probst, Erik Peterson, Christian Monson, Alon Lavie, Ralf Brown, and Lori Levin. 2002. Automatic rule learning for resource-limited mt. In *Machine Translation: From Research to Real Users*, pages 1–10. Springer.

Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D Manning. 2009. Discriminative reordering with chinese grammatical relations features. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation*, pages 51–59. Association for Computational Linguistics.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 531–540. Association for Computational Linguistics.

Yuan Ding and Martha Palmer. 2004. Synchronous dependency insertion grammars: A grammar formalism for syntax based statistical mt. In *Workshop on Recent Advances in Dependency Grammars (COLING)*, pages 90–97.

Kevin Gimpel and Noah A Smith. 2009. Feature-rich translation by quasi-synchronous lattice parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 219–228. Association for Computational Linguistics.

Kevin Gimpel and Noah A Smith. 2014. Phrase dependency machine translation with quasi-synchronous tree-to-tree features. *Computational Linguistics*.

Jonathan Graehl and Kevin Knight. 2004. Training tree transducers. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 105–112, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, pages 66–73.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Benoit Lavoie, Michael White, and Tanya Korelsky. 2002. Learning domain-specific transfer rules: an experiment with korean to english translation. In *Proceedings of the 2002 COLING workshop on Machine translation in Asia-Volume 16*, pages 1–7. Association for Computational Linguistics.

Dekang Lin. 2004. A path-based transfer model for machine translation. In *Proceedings of Coling 2004*, pages 625–630, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 609–616. Association for Computational Linguistics.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 133–139. Association for Computational Linguistics.

Fandong Meng, Jun Xie, Linfeng Song, Yajuan Lü, and Qun Liu. 2013. Translation with source constituency and dependency trees. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1066–1076, Seattle, Washington, USA, October. Association for Computational Linguistics.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proceedings of ACL-08: HLT*, pages 192–199, Columbus, Ohio, June. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 295–302, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

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

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Stephen Richardson, William Dolan, Arul Menezes, and Jessie Pinkham. 2001. Achieving commercial-quality translation with example-based methods. In *Proceedings of MT Summit VIII*, pages 293–298. Santiago De Compostela, Spain.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio, June. Association for Computational Linguistics.

Andreas Stolcke. 2002. Srilm-an extensible language modeling toolkit. In *Proceedings of ICSLP*, volume 30, pages 901–904.

Jun Xie, Haitao Mi, and Qun Liu. 2011. A novel dependency-to-string model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 216–226, Stroudsburg, PA, USA. Association for Computational Linguistics.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2007. A dependency treelet string correspondence model for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 40–47, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 523–530. Association for Computational Linguistics.

Fast Domain Adaptation of SMT models without in-Domain Parallel Data

Prashant Mathur* Fondazione Bruno Keseler Povo - 38100 (IT) first@fbk.eu

Sriram Venkatapathy Xerox Research Center Europe Meylan (FR) first.last@xrce.xerox.com

Nicola Cancedda* Microsoft, London (UK) first.last@gmail.com

Abstract

We address a challenging problem frequently faced by MT service providers: creating a domain-specific system based on a purely source-monolingual sample of text from the domain. We solve this problem by introducing methods for domain adaptation requiring no in-domain parallel data. Our approach yields results comparable to state-of-the-art systems optimized on an in-domain parallel set with a drop of as little as 0.5 BLEU points across 4 domains.

1 Introduction

We consider the problem of creating the best possible statistical machine translation (SMT) system for a specific domain when no parallel sample or training data from such domain is available. We assume that we have access to a collection of phrase tables (PT) and other models independently created from now **unavailable** corpora, and we receive a monolingual source language sample from a text source we would like to optimize for.

For a MT provider to deliver a SMT system tailored to a customer's domain, a sample dataset is requested. In most cases, the customer is able to provide an in-domain mono-lingual sample from his operations. However, it is generally not feasible for the customer to provide the translations as well because the customer has to hire professional translators to do that. In such a scenario, the translations has to be generated by MT service provider itself by hiring human translators thus requiring an investment upfront. The methods proposed in this paper aim to avoid that by building a good quality pilot SMT system leveraging only sample mono-lingual source corpus, and previously trained library of models. This in turn postpones the task of generating in-domain parallel data to a later date when there is a commitment by the customer.

Unavailability of the raw parallel data could derive from a trading model where data owners share intermediate-level resources like PTs, Reordering Models (RM) and Language Models (LM), but can not, or do not want to, share the textual data such resources were derived from. This particular scenario has been explained in (Cancedda, 2012).

This scenario is similar to the multi-model framework studied in (Sennrich et al., 2013), with the additional challenge that no parallel development set is available. We build on the linear mixture model combination of the cited work, extending it to our more challenging environment:

1. We propose a new measure derived from the popular BLEU score (Papineni et al., 2002) to assess the fitness of a PT to cope with a given monolingual sample S. This measure is computed from n-gram statistics that can be easily extracted from a PT.

2. We propose a new method for tuning the parameters of a log linear model that does not require an in-domain parallel development set, and yet achieves results very close to traditional tuning on parallel in-domain data.

We present our proposed metric *BLEU-PT* and computation of multi-model in Section 2. The parameter estimation of log-linear parameters of the SMT system is described in Section 3. We present experiments and results in Sections 4 and 6 respectively.

*Major part of the work was performed when the authors were in Xerox Research Center Europe.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1114–1123, Dublin, Ireland, August 23-29 2014.

2 Building Multi-Model

Given a library of phrase tables, the goal of this step is to generate a domain adapted multi-model. The challenging aspect in our scenario is the lack of in-domain parallel data, as well as absence of original parallel corpora corresponding to the library of models. This rules out the possibility of using metrics such as cross-entropy (Sennrich, 2012b) or LM-perplexity for computing the mixing coefficients. We present our proposed metric in section 2.1, and interpolation of the phrase tables in section 2.2.

2.1 BLEU-PT

Given a source corpus s, and a set of phrase tables $\{pt_1, pt_2,\ldots,pt_n\}$, the goal is to measure the similarity of each of these tables with s. For measuring the similarity, we use BLEU-PT which is an adaptation of the popular BLEU score for measuring the similarity between a corpus and a phrase table. The metric BLEU-PT is measured as described in Equation 1.

$$\text{BLEU-PT}(PT, S) = \left(\prod_{n=1}^{4} \frac{match(n|pt, s)}{total(n|s)} \right)^{1/4} \tag{1}$$

where $match(n|pt, s)$ is the count of n-grams of order n in the source corpus s that exist in the source side of the phrase table pt. $total(n|s)$ is the number of n-grams of order n in the source corpus.

2.2 Interpolating Models

A state-of-the-art approach for building multi-models is through linear interpolation of component models, exemplified in Equation 2 for the case of the forward conditional phrase translation model.

$$h_{phr}(s, t) = log \sum_{j=1}^{N} \phi_j P_{phr,j}(t|s) \tag{2}$$

Various approaches have been suggested for computing the coefficients ϕ of the interpolated model, the most recent being perplexity minimization described in (Sennrich, 2012b), where each translation model feature is optimized separately on the parallel development set. Our work is set in a scenario where no parallel development set is available for optimizing the interpolation coefficients. We have also observed that perplexity minimization is computationally intensive, requires aligned parallel development set, and the optimization time increases rapidly with increasing number of component models (for details, see Section 4.2).

We propose a simple approach for computation of the mixing coefficients that relies on the similarity of each model with respect to the test set. The mixing coefficients are obtained by normalizing similarity values. The similarity between a model (phrase table) and a corpus is computed using the BLEU-PT metric proposed in the previous section. Another similarity metric that can be used is *LM Perplexity*. However, in the current scenario we do not have resources (training data) to build a source side LM for computing the perplexity.

We empirically compare our method for computing mixing coefficients with the the perplexity minimization method. We also experiment with applying the mixing coefficients obtained by using our method for mixing features of a reordering and language model.

3 Parameter Estimation

The overall quality of translation is strongly impacted by how optimized the weights of the log-linear combination of various translation features are for a domain of interest. MERT (Och, 2003) and MIRA (Watanabe et al., 2007) are popular solution to compute an optimal weight vector by minimizing the error on a held-out parallel development set. BLEU and its approximations are commonly used error metrics. In this paper we assume lack of a parallel development set, therefore the above methods cannot be used.

Pecina et. al. (2012) showed that the optimized log-linear weight vector [1] of a SMT system does not depend as much on the actual domain of the development set (on which the system was optimized), as

[1] Not to be confused with the mixing coefficients in a linear combination of model components.

on how "distant" the relevant domain is from the domain of the training corpus used to build the SMT models. This is an important finding. It means that the weight vector can be modeled as a function of the **distance/similarity** between the in-domain development set and the model built from the training set. In this work, we learn this function from examples of previous parameter optimizations, using our BLEU-PT as a similarity metric. Once we have retrieved the most relevant PTs (translation and reordering models) from our library, and we have linearly interpolated them using normalized BLEU-PT, we use the learned model to estimate the optimal value of the log-linear weights, instead of optimizing them.

In order to learn this mapping, we create a dataset of examples (pairs of the form <BLEU-PT, log-linear weight vector>, where weight vectors are normalized to ensure comparability across models) by performing repeated optimizations for out-domain models on a number of parallel development sets (see section 4 for more details of this data) using a traditional optimization method (MIRA in this work). Based on this dataset, the function of our interest can therefore be learnt using a supervised approach. We explore two parametric methods and a non-parametric method. We present these in Section 3.1, and 3.2 respectively. For a mono-lingual source in a new domain, the BLEU-PT can be computed, and then mapped to the appropriate weight vector using the methods presented below.

3.1 Parametric Methods

We considered two distinct parametric methods for estimating the mapping from model/corpus similarity into weight vectors. The first one makes the assumption that parameters can be estimated independently of one another, given the similarity, whereas the second tries to leverage known covariance between distinct parameters in the vector.

3.1.1 Linear Regression

Motivated by initial experiments highlighting strong correlation between BLEU-PT and optimal feature weights (see Section 5.1 below), we assumed here a simple linear relation of the form:

$$\lambda_i^* = W_i \mathrm{X} + b_i \tag{3}$$

where λ_i^* is the optimal log-linear weight for feature i, X is the feature vector (BLEU-PT vector), W_i and b_i are coefficients to be estimated. While a drastic assumption, this has the advantage of limiting the risk of overfitting in a situation like ours where there is only relatively few datapoints to learn from. We estimate a_i and b_i by simple least squares regression. Once these are available for all features, we can predict the log linear weights of any model given its BLEU-PT similarity to a monolingual source sample using Eq. 3.

3.1.2 Multi-Task learning

Optimal log-linear parameters might not be fully independent given BLEU-PT, especially since it is known that model features can be highly correlated. To account for correlation between parameter weights, we explore the use of multi-task lasso[2] (Caruana, 1997) where several functions corresponding to each parameter are learned jointly considering the correlation between their values observed in the training data. Multi-task lasso consists of a least square loss model trained along with a regularizer and the objective is to minimize the following:

$$\arg\min_{w} \frac{1}{2N} ||\mathrm{X} \cdot W - \lambda||_2^2 + \alpha ||W||_{21} \quad where; \quad ||W||_{21} = \sum_{j}^{M} \sqrt{\sum_{i} w_{ij}^2} \tag{4}$$

Here, N is the number of training samples, X is the feature vector(BLEU-PT score vector) λ is the label vector(log linear weights). $||W||_{21}$ is the l_{21} regularizer (Yang et al., 2011). The problem of prediction of log linear weights is reduced to prediction of i interlinked tasks where each task has M features[3]. Coefficients are calculated using coordinate descent algorithm in Multi-Task lasso. Once the coefficients are calculated we use Eq. 3 to predict the log linear weights.

[2] `http://scikit-learn.org/`

[3] In our case we only have 1 feature i.e. BLEU-PT score.

3.2 Non Parametric: Nearest Neighbor

Finally, instead of building a parametric predictor for log linear weights, we experimented with a simple nearest-neighbor approach:

$$\lambda_i^* = \lambda_i(M_{j^*}) \tag{5}$$

where M_j ranges over the linearly interpolated phrase tables, and $\lambda_i(M)$ returns the stored optimal value for the i^{th} log-linear weight, and:

$$j^* = \arg\min_j \min_{s'}(|\text{BLEU-PT}(M, s) - \text{BLEU-PT}'(M_j, s')|) \tag{6}$$

where s is the monolingual sample on which we want to calculate the BLEU-PT and s' ranges over the source sides of our available parallel development sets. In other words, a BLEU-PT of a model is calculated on the source sample to be translated and the log-linear weight is chosen which corresponds to BLEU-PT′, where BLEU-PT′ is a training data point closest to BLEU-PT. This approach is close to the cross-domain tuning of Pecina et. al. (2012).

4 Experimental Program

We conducted a number of experiments for English-French language pair, comparing the methods proposed in the previous sections among one another and against state-of-the-art baselines and oracles.

4.1 Datasets

In this section, we present the datasets (EN-FR) that we have used for our experiments and the training data that was created for the purpose of supervised learning. We collected a set of 12 publicly available corpora and 1 proprietary corpus, statistics of datasets are provided in Table 1.

Corpus	Train	Development	Test
Commoncrawl	78M	12.4K	12.6K
ECB	4.7M	13.9K	14K
EMEA	13.8M	14K	15.7K
EUconst	133K	8K	8.4K
Europarl	52.8M	13.5K	13.5K
P1	5M	35K	14.5K
KDE4	1.5M	12.8K	5.8K
News Comm.	4M	12.7K	65K
OpenOffice	400K	5.4K	5.6K
OpenSubs	156M	16K	15.7K
PHP	314K	3.5K	4K
TED	2.65M	21K	14.6K
UN	1.92M	21K	21K

Table 1: Statistics of parallel sets (# of source tokens)

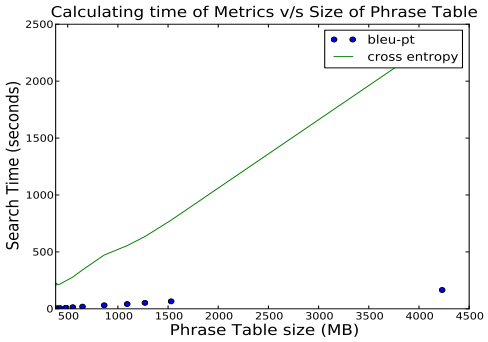


Figure 1: BLEU-PT v/s Cross-Entropy

Commoncrawl (CC) (Smith et al., 2013) and News Commentary (Bojar et al., 2013) corpora were provided in the 2013 shared translation task organized with workshop on machine translation. TED talks data was released as a part of IWSLT evaluation task (Cettolo et al., 2012). ECB, EMEA, EUconst, OpenOffice, OpenSubs 2011, PHP and UN corpora are provided as a part of OPUS parallel corpora (Tiedemann, 2012). The parallel corpora from OPUS were randomly split into training, development and testsets. Commoncrawl, News Commentary and TED datasets were used as they were provided in the evaluation task.

Out of 13 different domain datasets we selected 4 datasets randomly: Commoncrawl, KDE4, TED and UN (in bold in Table 1), to test our methods.

4.2 BLEU-PT v/s Cross-Entropy

We compared the overheads of calculating BLEU-PT and Cross-Entropy[4]. We are interested in estimating whether with increasing number of phrase tables the computation of both measures becomes slow or memory intensive.

[4]We used tmcombine.py script that comes along with the moses package to calculate the mixing coefficients.

Another advantage of using BLEU-PT apart from fast retrieval is that we can index the phrase tables using wFSA based indexing (explanation of indexing the phrase tables is not in the scope of this paper) and store the FSTs in binarised format on disk. When a source sample comes, we just load the indexed binaries and calculate the BLEU-PT while this cannot be achieved when we want to calculate cross entropy because we have to do one pass over all the phrase tables in question.

Experimental results depicted in Figure 1 shows that computation of BLEU-PT is fast (160 seconds) while computation of cross-entropy is slow (42 minutes) when we combine 12 phrase tables with total size of 4.2GB.

4.3 Training data for supervised learning and testing

As mentioned earlier, for estimating the parameters we require a training data containing the tuples of <BLEU-PT, log-linear-weight>. We perform parameter estimation on four of our datasets: Commoncrawl, KDE4, TED and UN. So, for obtaining evaluation results on say, UN, the rest of the resources are used for generating the training data. Our experimental setup can be explained well using the Venn diagram shown in Figure 2.

We set one of four domains as the test domain (in this case, UN) whose parallel set is not available to us and call it setup-UN. The training data tuples obtained from the rest of the 12 datasets are used to estimate parameters for the UN domain. From these 12 datasets we perform a round-robin experiment where one by one each dataset is considered as in-domain and the rest as out-domain. In-domain dataset provides the development set and the rest 11 out-domain models are linearly combined to build translation models. In figure 2, for example, the development set from the TED domain is taken as the development set of the multi-model build using the rest (i.e. excluding TED and UN). This multi-model is built by a weighted linear combination of the out-domain models (11 models). The parameters of this multi-model are tuned on the in-domain development set using MIRA. Simultaneously, we also calculate the BLEU-PT of the linear interpolated model on the source side of the in-domain development set (i.e. TED). This provides us the tuples of BLEU-PT and the log linear weights, which is our training data. So, four sets of experiments are conducted (one each for four datasets considered for testing), and for each set of experiments, there are 12 training data points. The final evaluation is done by measuring the BLEU score obtained on each test set using the predicted parameter estimates.

Reiterating, our optimizing method is fast, and hence, we are not not looking to learn the parameters apriori for all the domains based on a source side of the development set. The goal is to do a fast adaptation by predicting the parameters using statistical models for every new test in a particular domain even in the absence of a parallel development set.

4.4 Prediction

For prediction of parameters for a new domain, the BLEU-PT of the sample source corpus (UN in our example) is measured with the multi-model built on all the models (all the rest of 12 datasets including the TED model) and then the supervised predictor is applied. In our experiments, we test both parametric and non-parametric methods to estimate the parameters based on the training data obtained using the 12 domains.

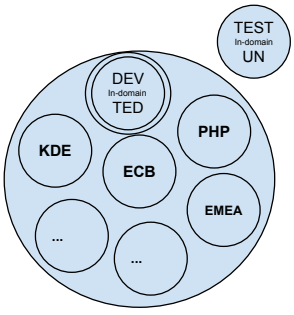


Figure 2: Cross domain tuning setup

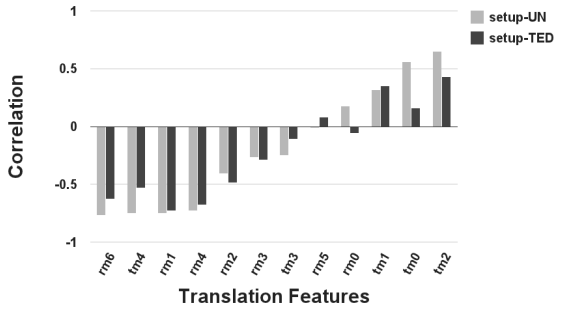


Figure 3: Correlation of log linear weights with BLEU-PT when indomain sets set to UN and TED

	Domain			Linear Interpolation		
System	**Train**	**Dev**	**Param. Est.**	**TM(coeff.)**	**RM(coeff.)**	**LM(coeff.)**
in-dom-train	In	In	mira	N.A	N.A	N.A
mira-bleupt-tm-rm	Out	In	mira	✓	✓	✗
mira-perp-tm-bleupt-rm	Out	In	mira	✓(Perp. Min)	✓	✗
mira-bleupt-tm-rm-perp-lm	Out	In	mira	✓	✓	✓(LM Perp. Min.)
mira-bleupt-all	Out	In	mira	✓	✓	✓
def-bleupt-all	Out	✗	def	✓	✓	✓
gen-reg-bleupt-all	Out	✗	regression	✓	✓	✓
gen-mtl-bleupt-all	Out	✗	multi-task	✓	✓	✓
gen-nn-bleupt-all	Out	✗	Near.Neigh.	✓	✓	✓
top5-reg-bleupt-all	Out	✗	regression	✓	✓	✓
top5-mtl-bleupt-all	Out	✗	multi-task	✓	✓	✓
top5-nn-bleupt-all	Out	✗	Near.Neigh.	✓	✓	✓

Table 2: System Description: Each system's training domain and development set domain along with the optimizer/predictor is mentioned. *def-bleupt-all* uses default weights from Moses decoder. Near.Neigh. shows that we used Nearest Neighbor predictor for optimizing weights. ✗ represent log linear interpolation of models while ✓ represents linear interpolation. The mixing coefficients for linear interpolation are calculated by normalizing bleu-pt scores unless mentioned otherwise.

5 Experiments and Results

5.1 Correlation analysis

Before embarking in the actual regression task, we examined the correlation between the similarity values (BLEU-PT) and the various weights in the training data. If there is good correlation between BLEU-PT and a particular parameter, then the linear regressor is expected to fit well and then predict an accurate parameter value for a new domain. For computing the correlation, we use Pearson correlation coefficient (PCC). Figure 3 shows the PCC between the feature weights and the BLEU-PT scores. The tm's are the translation model features, and rm's are the reordering model features.

We see that there is either a strong positive correlation or a strong negative correlation for most features in both the experimental setups shown in the figure 3. This validates our hypothesis that optimal parameters for a new test domain can indeed be estimated with good reliability. One can also observe that the correlation level also varies based on the mixture of training models. For example, the correlation is much higher in the training data that excluded UN (setup-UN) than the one that excluded TED (setup-TED).

In figure 3, one can also see that $tm0$ (forward phrase conditional probability) and $tm2$ (backward phrase conditional probability) which are shown in previous work to be the two most important features amongst all SMT features (Lopez and Resnik, 2006) in terms of their impact on translation quality, have a high correlation in setup-UN.

5.2 Systems

All SMT systems were built using the Moses toolkit (Koehn et al., 2007). To automatically align the parallel corpora we used MGIZA (Gao and Vogel, 2008). Aligned training data in each domain was then used to create the corresponding component translation models and lexical reordering models. We created 5-gram language models for every domain using SRILM (Stolcke, 2002) with improved Kneser-Ney smoothing (Chen and Goodman, 1999) on the target side of the training parallel corpora. Log linear weights for the systems were optimized using MIRA (Watanabe et al., 2007; Hasler et al., 2011) which is provided in the Moses toolkit. Performance of the systems are measured in terms of BLEU computed using the MultEval script (mteval-v13.pl).

We built one *in-dom-train* system where only in-domain training data is taken into account. This system shows the importance of in-domain training data in SMT (Haddow and Koehn, 2012). Three oracle systems are trained on out-domain training corpus and tuned on in-domain development data (in this case there are four domains we chose to test on: UN, TED, CommonCrawl and KDE4), thus 4 systems for each of the in-domain test sets.

We build another set of SMT systems in which language models are combined by linear interpolation[5].

[5]Linear interpolation of 12 LMs result in one single large LM, thus, one weight. So, a total of 14 weights have to be optimized or predicted

The systems using linear interpolated LM (mixing coefficients are normalized BLEU-PT scores) are *def-bleupt-all*, *mira-bleupt-all*, *gen-reg-bleupt-all*, *gen-mtl-bleupt-all* and *gen-nn-bleupt-all*. We compare *mira-bleupt-all* with *mira-bleupt-tm-rm-perp-lm* where mixing coefficients for LM interpolation are calculated by standard LM perplexity minimization method over target side of development set.

As mentioned earlier, ideally only a subset of all the models closer to the source sample should be taken into account for **quick** adaptation, so we select the top five domains related to the source sample and interpolate the respective models and address them as *top5-** systems. Adding more domains would unnecesary increase the size of the model and add more noise. Table 2 shows the configuration of different systems. In the next section we compare the performances of these systems and report the findings.

6 Results and Discussion

Table 3 presents results of the systems that use an in-domain parallel data. As expected, when an in-domain corpus is used both for training as well as for optimizing the log-linear parameters, the peformance is much higher than those systems that do not use in-domain parallel corpus for training (Koehn and Schroeder, 2007). We also observe that the use of normalized BLEU-PT for computing mixing coefficients gives comparable performance to using Cross-Entropy. The primary advantage in using BLEU-PT is that it can be compute much faster than Cross-Entropy (as shown in Figure 1). Evidently, normalized BLEU-PT scores as mixing coefficients performs at par with mixing coefficients retrieved by standard perplexity minimization method (Bertoldi and Federico, 2009). One can also use BLEU-PT for LM interpolation in cases where target side in-domain text is not available.

System	UN	TED	CC	KDE
in-dom-train	67.87	29.98	26.62	35.82
mira-bleupt-tm-rm	**44.14**	31.20	17.43	24.25
mira-perp-tm-bleupt-rm	43.56	31.36	17.54	**24.72**
mira-bleupt-tm-rm-perp-lm	43.96	31.85	**18.45**	23.39
mira-bleupt-all	43.66	**32.04**	18.44	23.09

Table 3: Comparison of In-Domain system versus the established Oracles in different setups.

System	UN	TED	CC	KDE
gen-reg-bleupt-all	43.27	32.18	17.95	21.05
gen-mtl-bleupt-all	43.35	32.61	18.26	20.67
gen-nn-bleupt-all	42.73	31.04	18.24	21.85

Table 4: Performance of generic systems (gen-*) in all setups.

Table 4 illustrates the impact of phrase table retrieval on the performance of multi-model. All the systems presented in this table use BLEU-PT for computing mixing coefficients, while the weights are computed using the three techniques that we explored in this paper. We see that in case of regression, the phrase table retrieval also results in a better MT performance. In the other two cases, the results are comparable. It shows that retrieval helps in building smaller sized multi-models while being more accurate on an average. Phrase table retrieval, thus, becomes particularly useful when a multi-model needs to be built from a library of dozens of pre-trained phrase tables of various domains.

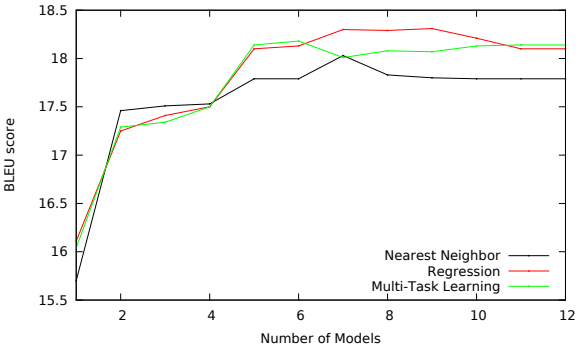


Figure 4: BLEU scores when top k models were used to evaluate commoncrawl test set where $k \in 1..12$.

System	UN	TED	CC	KDE
def-bleupt-all	42.03	30.82	17.97	19.66
mira-bleupt-all	43.66	32.04	18.44	23.09
top5-reg-bleupt-all	43.39$^{\blacktriangle}$	32.31$^{\blacktriangle}$	18.10	21.54$^{\blacktriangle}$
top5-mtl-bleupt-all	43.56$^{\blacktriangle}$	32.60$^{\blacktriangle}$	18.14	20.91$^{\blacktriangle}$
top5-nn-bleupt-all	42.96$^{\blacktriangle}$	30.89$^{\triangle}$	17.79	22.24$^{\blacktriangle}$

Table 5: Comparing the baseline system (def-bleupt-all) and Oracle (mira-bleupt-all) with domain specific multi-model systems trained on top5 domains. $^{\blacktriangle}$and $^{\triangle}$denotes significantly better results in comparison with def-bleupt-all system with p-value < 0.0001 and < 0.05 respectively.

Table 5 compares our approach of computing log-linear weights (in the absence of in-domain development set) to the state-of-art weight optimization technique MIRA (which requires an in-domain development set). As a baseline, we set default weights to all the parameters, which was shown to a strong

baseline in (Pecina et al., 2012). We see that the methods proposed by us perform significantly better than the default weights baseline (improvement of more than 1.5 BLEU score on an average across 4 domains). Among the three approaches for computing weights, the method that uses multi-task lasso performs best (except in setup-KDE where the non-parametric method performs best), along the expected lines as multi-task lasso considers the correlation between various features. In comparison to MIRA, our methods result in an average drop of as little as 0.5 BLEU points across 4 domains (see Table 5).

Figure 4 shows BLEU score curve when we vary the k in top-k systems. BLEU score curve is almost tangential zero when k is between 5 and 6 which essentially means that selection of $k = 5$ is a good choice. For CommonCrawl test set, the top five domains used were Europarl, OpenSubs, NewsCommentary, TED and ECB. This is a significant result which indicates that one can build a good system for a domain even in the absence of the parallel data in the domain of interest.

7 Related Work

Domain adaptation in statistical machine translation has been widely studied and leveraged through adding more training data (Koehn and Knight, 2001), filtering of out of domain training data (Axelrod et al., 2011; Koehn and Haddow, 2012), fillup technique (Bisazza et al., 2011), language model adaptation by perplexity minimization over in-domain data (Bertoldi and Federico, 2009) and various other approaches. However, all the above adaptation approaches require either parallel in-domain corpus or monolingual in-domain target side corpus, thus, not applicable in our scenario.

In this paper we studied mixture modelling of heterogeneous translation models which was first proposed in Foster et. al. (2007). They showed various ways of computing mixing coefficients for linear interpolation using several distance based metrics borrowed from information theory. However, to calculate any such metrics it was required that one has an access to the source/target training corpus and source/target development corpus. Other noteable works in mixture modelling in SMT are (Civera and Juan, 2007; Razmara et al., 2012; Duan et al., 2010).

More recently, Sennrich (2012b) designed an approach to calculate mixing coefficients by minimizing the perplexity of translation models over an **aligned** development set for mixture modelling via linear interpolation or by weighting the corpora. Sennrich et. al. (2012a) clustered of a large heterogeneous development corpus and tuned a translation system on different clusters. In the decoding phase each sentence was assigned to a cluster and the translation system tuned on that cluster was used to translate that sentence.

(Banerjee et al., 2010) build several domain specific translation systems, and trained a classifier to assign each incoming sentence to a domain and use the domain specific system to translate the corresponding sentence. They assume that each sentence in test set belongs to one of the already existing domains which means it would fail in the case where the sentence doesn't belong to any of the existing domains. In our case we do not make any such assumptions.

Academically, above approaches are well suited for solving the problem of domain adaptation, but during the deployment of SMT systems in industrial scenario where the client is unable to deliver the parallel in-domain data these approaches fail to provide a quick solution.

8 Conclusion

We present an approach to multi-model domain adaptation in a particularly challenging setting where there is no parallel in-domain data. Parameter estimation without in-domain development set is a problem that, to the best of our knowledge, has not been addressed before. We designed a method for tuning model parameters without parallel development set and validated it through an experimental program for which we compared performances against an array of Oracles and Baselines. The effectiveness of the proposed method empirically supports the findings of (Pecina et al., 2012), who discovered that the log linear weights largely depend on the **distance** of training domain from the domain on which the models are being optimized on. As a side result, we designed in the process a novel similarity metric between a phrase table and a source sample and implemented it effectively using wFSAs. We empirically showed the excellent computation speed of BLEU-PT scores as compared to standard Cross-Entropy measure using standard toolkits.

Acknowledgement

The authors thank the three anonymous reviewers for their comments and suggestions.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 355–362, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pratyush Banerjee, Jinhua Du, Baoli Li, Sudip Kumar Naskar, Andy Way, and Josef Van Genabith. 2010. Combining multi-domain statistical machine translation models using automatic classifiers. In *Proceedings of 9th Conference of the Association for Machine Translation in the Americas*.

Nicola Bertoldi and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, pages 182–189, Stroudsburg, PA, USA. Association for Computational Linguistics.

Arianna Bisazza, Nick Ruiz, and Marcello Federico. 2011. Fill-up versus Interpolation Methods for Phrase-based SMT Adaptation. In *International Workshop on Spoken Language Translation (IWSLT)*, pages 136–143, San Francisco, CA.

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria, August. Association for Computational Linguistics.

Nicola Cancedda. 2012. Private access to phrase tables for statistical machine translation. In *ACL (2)*, pages 23–27.

Rich Caruana. 1997. Multitask learning. *Mach. Learn.*, 28(1):41–75, July.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy, May.

Stanley F. Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 4(13):359–393.

Jorge Civera and Alfons Juan. 2007. Domain adaptation in statistical machine translation with mixture modelling. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.

Nan Duan, Mu Li, Dongdong Zhang, and Ming Zhou. 2010. Mixture model-based minimum bayes risk decoding using multiple machine translation systems. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 313–321. Association for Computational Linguistics.

George Foster and Roland Kuhn. 2007. Mixture-model adaptation for smt. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 128–135, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, SETQA-NLP '08, pages 49–57, Stroudsburg, PA, USA. Association for Computational Linguistics.

Barry Haddow and Philipp Koehn. 2012. Analysing the effect of out-of-domain data on smt systems. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 422–432, Montreal, Canada, June. Association for Computational Linguistics.

Eva Hasler, Barry Haddow, and Philipp Koehn. 2011. Margin Infused Relaxed Algorithm for Moses. *The Prague Bulletin of Mathematical Linguistics*, 96:69–78.

Philipp Koehn and Barry Haddow. 2012. Towards effective use of training data in statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 317–321, Stroudsburg, PA, USA. Association for Computational Linguistics.

Philipp Koehn and Kevin Knight. 2001. Knowledge sources for word-level translation models. In *In Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 27–35.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 224–227, Stroudsburg, PA, USA. Association for Computational Linguistics.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic.

Adam Lopez and Philipp Resnik. 2006. Word-based alignment, phrase-based translation: What's the link? In *In Proceedings of AMTA*, pages 90–99.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-jing Zhu. 2002. BLEU : a Method for Automatic Evaluation of Machine Translation. In *Computational Linguistics*, volume pages, pages 311–318.

Pavel Pecina, Antonio Toral, and Josef van Genabith. 2012. Simple and effective parameter tuning for domain adaptation of statistical machine translation. In *COLING*, pages 2209–2224.

Majid Razmara, George Foster, Baskaran Sankaran, and Anoop Sarkar. 2012. Mixing multiple translation models in statistical machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 940–949. Association for Computational Linguistics.

Rico Sennrich, Holger Schwenk, and Walid Aransa. 2013. A multi-domain translation model framework for statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 832–840, Sofia, Bulgaria, August. Association for Computational Linguistics.

Rico Sennrich. 2012a. Mixture-modeling with unsupervised clusters for domain adaptation in statistical machine translation. In *Proceedings of the 16th Annual Conference of the European Association of Machine Translation (EAMT)*.

Rico Sennrich. 2012b. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 539–549, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jason R. Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. 2013. Dirt cheap web-scale parallel text from the common crawl. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1374–1383, Sofia, Bulgaria, August. Association for Computational Linguistics.

Andreas Stolcke. 2002. Srilm - an extensible language modeling toolkit. In *Proceedings of ICSLP*, Denver, Colorado.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uur Doan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 764–773, Prague, Czech Republic, June. Association for Computational Linguistics.

Yi Yang, Heng Tao Shen, Zhigang Ma, Zi Huang, and Xiaofang Zhou. 2011. 12, 1-norm regularized discriminative feature selection for unsupervised learning. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Two*, pages 1589–1594. AAAI Press.

Discriminative Language Models as a Tool for Machine Translation Error Analysis

Koichi Akabe Graham Neubig Sakriani Sakti Tomoki Toda Satoshi Nakamura
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma-shi, Nara, Japan
{akabe.koichi.zx8, neubig, ssakti, tomoki, s-nakamura}@is.naist.jp

Abstract

In this paper, we propose a new method for effective error analysis of machine translation (MT) systems. In previous work on error analysis of MT, error trends are often shown by frequency. However, if we attempt to perform a more detailed analysis based on frequently erroneous word strings, the word strings also often occur in correct translations, and analyzing these correct sentences decreases the overall efficiency of error analysis. In this paper, we propose the use of regularized discriminative language models (LMs) to allow for more focused MT error analysis. In experiments, we demonstrate that our method is more efficient than frequency-based analysis, and examine differences across systems, language pairs, and evaluation measures. [1]

1 Introduction

Accuracy of Statistical Machine Translation (SMT) systems is continually increasing, but systems are now more complex than ever before. As a result, not all effects of making modifications to a system are known without actually making the modification and generating translations. Therefore, in the process of developing an SMT system, it is common to evaluate actual translations to identify problems to make improvements. This process is time consuming, as it is often necessary to analyze a large number of translations to get an overall grasp of the system's error trends. In addition, many sentences will contain no errors, or only errors from the long tail that are not representative of the system as a whole. On the other hand, if we are able to detect and rank important errors automatically, we will likely be able to find representative errors of the SMT system more efficiently.

Previous work has proposed methods for automatic error analysis of MT systems based on automatically separating errors into classes and sorting these classes by frequency (Vilar et al., 2006; Popovic and Ney, 2011). These classes cover common mistakes of MT systems, e.g. conjugation, reordering, word deletion, and insertion. This makes it possible to view overall error trends, but when the goal of analysis is to identify errors to make some concrete improvement to the system, it is often necessary to perform a more focused analysis, looking at actual errors made by a particular language pair or system. We show examples of errors types that are informative, but are language- or task-specific, and not covered by previous methods in Figure 1. In this example, the type given by more standard error typologies is indicated by "Traditional type," but we would prefer a more detailed analysis such as "Fine-grained type," would allows us to take specific steps to fix the machine translation system (such as ensuring that Wikipedia titles are not punctuated, or normalizing full-width characters to half-width). These fine-grained types are difficult to conceive without actually observing the MT system output, but if we are able to group actual errors into fine-grained classes based on, for example, lexical clues, this sort of analysis will become possible and more efficient.

Previous research on improving the efficiency of error analysis has generally focused on grouping error types by frequency, but try to apply such frequency-based techniques to individual errors, selected errors

[1] Our implementation is available open-source at https://github.com/vbkaisetsu/dlm-analyzer

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1124–1132, Dublin, Ireland, August 23-29 2014.

Src	学術 交流 協定
Ref	the academic exchange agreement
MT	academic exchange agreement .
Traditional type	Insertion error
Fine-grained type	Insertion error (unneeded period)

Src	覚行 法 親王 （ １０７５ － １１０５ ） － 仁和 寺門 跡
Ref	prince kakugyoho -lrb- 1075 - 1105 -rrb- ninna-ji monzeki
MT	imperial prince kakugyo -lrb- １０７５ - 1105 -rrb- : ninna-ji temple ruins
Traditional type	Replacement error or Unknown word
Fine-grained type	Unknown word (number) or Half-/Full-width error

Figure 1: Example of errors in Japanese to English translation, classified into traditional, or more fine-grained and useful classes.

1-gram		2-gram	
the	61	(BOS) the	42
,	47	. (EOS)	41
and	43	, and	32
of	42	of the	27
:	42	in the	21

Table 1: Frequently occurring erroneous n-grams

are often dominated by frequently occurring linguistic phenomena that are not necessarily indicative of translation errors. To show examples of this problem, in Table 1 we provide a list of erroneous n-grams that were produced by an MT system (described in Section 4.1) but not contained in the respective references. From this table, we can see that frequently occurring erroneous n-grams are simply n-grams that frequently occur in English, and because of this we cannot discover characteristic errors of the system for improvement just from this information.

In this paper, we propose a new method that uses regularized discriminative LMs to solve the above problem. Discriminative LMs are LMs trained to fix common output errors of a particular system. From the viewpoint of error analysis, if we train a discriminative LM using n-gram features and examine the weights learned by this model, n-grams with large negative or positive weights will be indicative of patterns that are over- or under-produced by the MT system. Because the weights are specifically trained to fix errors, it is likely that these patterns will be more informative than mistakes that are simply frequently occurring. We can also use a number of features of discriminative LMs to perform a more focused and efficient analysis. For example, if we perform training with L1 regularization, many features will be removed and only important patterns will remain in the model. Additionally, we can focus on specific varieties of errors by changing the evaluation measure used for training the LMs.

In our experiments, we validate the effectiveness of error analysis based on discriminative LMs. We perform a manual evaluation of the n-gram patterns discovered by random selection, by frequency-based analysis, and by the proposed method. As a result, the proposed method is more effective at identifying errors than other methods.

2 Discriminative Language Models

In this section, we first introduce the discriminative LM used in our method. As a target for our analysis, we have input sentences $\mathcal{F} = \{F_1, \ldots, F_K\}$, n-best outputs $\hat{\mathcal{E}} = \{\hat{\boldsymbol{E}}_1, \ldots, \hat{\boldsymbol{E}}_K\}$ of an MT system, and reference translations $\mathcal{R} = \{R_1, \ldots, R_K\}$. Discriminative LMs define feature vectors $\boldsymbol{\phi}(E_i)$ for each candidate in $\hat{\boldsymbol{E}}_k = \{E_1, E_2, \ldots, E_I\}$, and calculate inner products $\boldsymbol{w} \cdot \boldsymbol{\phi}(E_i)$ as scores.

To train the weight vector $\boldsymbol{w}$, we first calculate evaluation scores of all candidates using a sentence-level evaluation measure EV such as BLEU+1 (Lin and Och, 2004) given the reference sentence R_k.

We choose the sentence with the highest evaluation EV as an oracle E_k^*. Oracles are chosen for each n-best, and we train $\boldsymbol{w}$ so that the oracle's score becomes higher than the other candidates.

2.1 Structured Perceptron

While there are a number of methods for training discriminative LMs, we follow Roark et al. (2007) in using the structured perceptron as a simple and effective method for LM training. The structured perceptron is a widely used on-line learning method that examines one training instance and updates the weight vector using the difference between feature vectors generated from the oracle E^* and the hypothesis $\hat{E}$ calculated by the current model. For each iteration, $\boldsymbol{w}$ is updated using the difference between E^* and $\hat{E}$. If $\hat{E}$ is equal to E^*, the difference becomes $\mathbf{0}$, so no update is performed. This process is run for all $\mathcal{F}$ sequentially, and iterated until weights converge or we reach a fixed iteration limit N. We show the above procedure in Algorithm 1.

Algorithm 1 Structured perceptron training of the discriminative LM

for $n = 1$ **to** N **do**
  for all $\hat{\boldsymbol{E}} \in \hat{\mathcal{E}}$ **do**
    $E^* \leftarrow \arg\max_{E \in \hat{\boldsymbol{E}}} EV(E)$
    $\hat{E} \leftarrow \arg\max_{E \in \hat{\boldsymbol{E}}} \boldsymbol{w} \cdot \boldsymbol{\phi}(E)$
    $\boldsymbol{w} \leftarrow \boldsymbol{w} + \boldsymbol{\phi}(E^*) - \boldsymbol{\phi}(\hat{E})$
  end for
end for

2.2 Learning Sparse Discriminative LMs

While the structured perceptron is a simple and effective method for learning discriminative LMs, it also has no bias towards reducing the number of features used in the model. However, if we add a bias towards learning smaller models, we can keep only salient features (Tsuruoka et al., 2009).

In our work, we use L1 regularization to add this bias. L1 regularization gives a penalty to $\boldsymbol{w}$ proportional to the L1 norm $\|\boldsymbol{w}\|_1 = \sum_i |w_i|$, pushing a large number of elements in $\boldsymbol{w}$ to 0, so ineffective features are removed from the model.

To train L1 regularized discriminative LMs, we use the forward-backward splitting (FOBOS) algorithm proposed by Duchi and Singer (2009). FOBOS splits update and regularization, and lazily calculates the regularization upon using the weight to improve efficiency.

2.3 Features of Discriminative LMs

In the LM, we used the following three features:

1. System score feature ϕ_s: As our goal is fixing the output of the system, we add this feature to allow a default ordering of n-bests by score.
2. n-gram feature ϕ_n: We add a binary feature counting the frequency of each n-gram in the hypothesis. The weights of these features will be the main target of our analysis.
3. Hypothesis length feature ϕ_l: If the evaluation measure has a penalty for the number of words, this allows us to adjust it.

In this work, we do not use other features, but our method theoretically allows for addition of other features such as POS tags or syntactic information, which could also potentially be used as a target for analysis.

3 Discriminative LMs for Error Analysis

In this section, we describe how to incorporate information from discriminative LMs into manual error analysis.

Error types	
Replacement	(Context dependent) (Context independent)
Insertion	
Deletion	
Reordering	
Conjugation	
Polarity	
Unknown words	

Table 2: Error categories for annotation

Src	京 ちゃん （ 市 バス ）
Ref	kyo-chan -lrb- city bus -rrb-
MT	<s> kyoto chan -lrb- kyoto city bus -rrb- </s>
Rules	SYMP (x0:SYM SYMP (NP (NN (”市”) NN (”バス”)) x1:SYM)) → x0 ”kyoto” ”city” ”bus” x1
Eval	Insertion error
Src	公開 特許 件数 13 件
Ref	there are 13 open patents .
MT	<s> the number of public patent 13 cases </s>
Rules	NP (NP (x0:NN x1:NN) NN (”件数”)) → ”number” ”of” x0 x1 NN (”公開”) → ”public”
Eval	Context-dependent replacement error

Figure 2: Example of the evaluation sheet. Boxed words are chosen n-grams.

3.1 Focused Error Analysis of MT output

We first define the following general framework for focused analysis of errors in MT output. Using this, we can find error trends of chosen n-grams:

1. Automatically choose potentially erroneous n-grams in the MT output.
2. Select one or more 1-best translations that contain each chosen n-gram.
3. Show selected translations to an annotator with the selected n-gram highlighted.
4. The annotator looks at the indicated n-gram, and marks whether or not by examining the n-gram whether they were able to identify an error in the MT output. If the answer is “yes,” the annotator additionally indicates which variety of error was found according to Table 2.

A part of the actual evaluation sheet is shown in Fig. 2. The first four rows are the input, and the final row is the annotator’s evaluation.

3.2 Selection of Target n-grams

We can think of the following three methods for choosing potentially erroneous n-grams:

Random: n-grams that are selected randomly. This corresponds to the standard method of error analysis, where sentences are randomly sampled and analyzed.

	Sent	Words	
		English	Japanese
Train	330k	5.91M	6.09M
Dev	1166	24.3k	26.8k
Test	1160	26.7k	28.5k

Table 3: Data size of KFTT

Frequency: n-grams that are most frequently over-generated (occur in the hypothesis, but not in the references). This corresponds to a focused version of the frequency-based automatic error analysis methods of Vilar et al. (2006) and Popovic and Ney (2011).

LM: n-grams that have the lowest weight according to the discriminative LM. This is our proposed method.

In particular, for discriminative LMs, n-gram features that have large positive or negative weights indicate n-grams that are under-generated or over-generated by the system. Therefore, by examining high-weighted or low-weighted n-grams, it is likely that we will be able to get a grasp of the system mistakes. When performing actual evaluation, we want to analyze n-grams with 1-best translations. Almost high-weighted n-grams are only contained in oracle translations, and not contained in 1-best translation. Therefore, we use low-weighted n-grams for evaluation. If the discriminative LM is properly trained, low-weighted n-grams will often correspond to actual errors.

3.3 System Comparison

When developing MT systems, it is common to not only evaluate a single system, but also compare multiple systems, such as when comparing a new system with baselines.

To do this in the current work, we create discriminative LMs from n-bests generated by multiple translation systems, and choose representative n-grams using the proposed method. Then we examine the selected n-grams in context and then compare the result of this analysis.

4 Experiments

We evaluate the effectiveness of our method by performing a manual evaluation over three translation systems, two translation directions, and two evaluation measures.

4.1 Experiment Setup

For each MT system, we use Japanese-English data from the KFTT (Neubig, 2011) as a corpus. The size of the corpus is shown in Table 3. In our experiment, we use a forest-to-string (F2S) system trained using the Travatar toolkit (Neubig, 2013) for single system evaluation. For system comparison, we compare the above F2S system with a phrase based (PBMT) system and a hierarchical phrase based (HIERO) system built using Moses (Koehn et al., 2007).

The F2S system is built using Nile[2] for making word alignments, and syntax trees generated with Egret[3]. PBMT and HIERO are built using GIZA++ (Och and Ney, 2003) for word alignments. Each system is optimized using MERT (Och, 2003) with BLEU (Papineni et al., 2002) as an evaluation measure. For single system evaluation, we also use the reordering-oriented evaluation metric RIBES (Isozaki et al., 2010) as additional metric for training the discriminative LM.

For training discriminative LMs, our method uses the structured perceptron with 100 iterations and FOBOS for L1 regularization as described in Section 2.2. The regularization factor is chosen from the range 10^{-6}-10^{-2} to give the highest performance on the KFTT test data.

LMs are trained using 500-bests from each MT system and features described in Section 2.3. We use 1-grams to 3-grams as n-gram features.

[2] http://code.google.com/p/nile/

[3] http://code.google.com/p/egret-parser/

System	BLEU(dev)		BLEU(test)	
	Original	LM applied	Original	LM applied
PBMT	0.2929	0.3521	0.2460	0.2485
HIERO	0.2953	0.3859	0.2616	0.2562
F2S	0.2958	0.3887	0.2669	0.2676

Table 4: Translation accuracy of each system, without LMs and with LMs

Method	Ja → En	En → Ja
Random	0.46	0.37
Frequency	0.30	0.31
LM	0.55	0.48

Table 5: Precision of top 30 n-grams that select errors in both directions

We show translation accuracies of each system before and after training in Table 4. From this table, we can see that the LM increases the accuracy of all dev data, but it does not necessarily have a large effect for the test data. The main reason for this is because the development set used to train the LM is relatively small, at only 1166 sentences. However, as our goal in this paper is to perform error analysis on set of data which we already have parallel references (in this case, the development set), the generalization ability of the model is not necessarily fundamental to our task at hand. We directly identify the ability to identify errors in the next section.

4.2 Evaluation of Error Identification Ability

This section evaluates the ability of our method to identify errors in MT output. As we are proposing our method as a tool for manual analysis of MT output, it is necessary to perform manual evaluation to ensure that our method is identifying locations that are actually erroneous according to human subjective evaluation. To measure the accuracy of each method, we perform an evaluation as described in Section 3.1 and use the precision of selected n-grams (the percentage of selected n-grams for which then annotator indicated that an error actually existed) as our evaluation measure. The annotator is an MT specialist who is proficient in English and Japanese. The order of the evaluation sentences is shuffled so the annotator can not determine which method was responsible for choosing each n-gram.

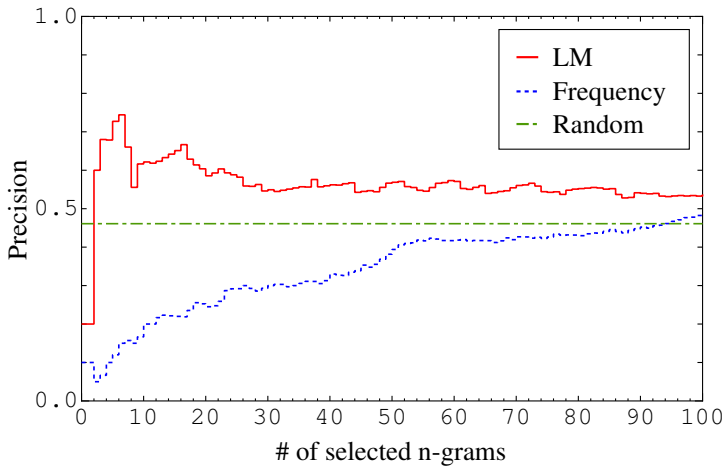


Figure 3: Precision of n-grams that select errors (Japanese to English)

We show the precision results for each number of selected n-grams over three methods for Japanese-English translation in Fig. 3, and the precision of the top 30 n-grams in both directions in Table 5. From

n-gram	Weight	Examples	
-rrb- of	-7.50950	Src	幕府 滅亡 の 勲功 第 一 と さ れ 、 後醍醐 天皇 [の] 諱 ・ 尊治 （ たかはる [）] の 御 一 字 を 賜 り 、 名 を 尊 氏 に 改め る 。
		Ref	his achievements were evaluated by emperor go-daigo , and he was awarded the letter -lrb- 尊 -rrb- , which came from the emperor [’s] real name takaharu -lrb- 尊治 [-rrb-] , so he changed the letter in his name from ” 高氏 ” to ” 尊氏 ” .
		MT	<s> it is regarded as a valor in the fall of the bakufu , and was the first character of takaharu , imina -lrb- たかはる [-rrb- of] emperor godaigo , and changed his name to takauji . </s>
		Eval	Reordering error
<s> the first	-6.55510	(Only contained in other candidates in n-bests)	
senior	-6.52024	Src	教龍 会 － 龍谷 大学 の 卒業 生 で 、 [中高] など の 学校 教員 から な る 組織 。
		Ref	kyoryukai-this organization consists of teachers of [junior high , high] , and other schools who are ryukoku university graduates .
		MT	<s> graduates of 教龍 association - ryukoku university , and is a organization consisting of teachers such as [senior] . </s>
		Eval	Context independent replacement error
the ko clan	-6.52021	Src	この 時 、 [高 氏] の 側室 の 子 ・ 竹若丸 が 混乱 の 最中 に 殺 さ れ て い る 。
		Ref	in this fighting , takewakamaru , the son of [takauji] ’s concubine , was killed .
		MT	<s> on this occasion , was killed during the confusion 竹若丸 , the son of a concubine of [the ko clan] . </s>
		Eval	Context dependent replacement error
foundation of	-6.50773	Src	藤原 基経 [創 建] と い わ れ る 京都 九条 に あ っ た 九条 殿 に 住 ん だ 事 が 家名 の 由来 。
		Ref	the family name comes from the fact that the kujo family lived in kujo-den , which was located in kyoto kujo and said to have been [built by] fujiwara no mototsune .
		MT	<s> the origin of the family name that lived in kujo dono , which was located in kyoto kujo is said to be a [foundation of] fujiwara no mototsune . </s>
		Eval	Context dependent replacement error

Table 6: Top 5 erroneous n-grams learned by the discriminative LM and examples. Boxes on MT indicates the selected n-gram, and boxes in Src and Ref indicate the corresponding words.

these results, we can see that each method is able to detect erroneous n-grams, but the proposed method achieves a precision that outperforms other methods.

To demonstrate why this is the case, in Table 6 we show examples, in context, of potentially erroneous n-grams chosen by our proposed method. Compared to the baseline n-grams in Table 1, we can see that these n-grams are not limited to frequently occurring n-grams in English, and are more likely to have a high probability of indicating actual errors.

In addition, to give a better idea of the prominence of the selected n-grams, in Table 7, we show the mean number of locations of the KFTT test data that contain the top 100 n-grams selected by each method. We can see that randomly selected n-grams are rarely contained in the separate test set, while the proposed method tends to select n-grams that are more frequent than random, and thus have a better chance of generalizing.

4.3 Effect of Evaluation Measure Choice

We can also hypothesize that by varying the evaluation measure used in training the LM, we can select different varieties of errors for analysis. To test this, we compare analysis results obtained using one

Method	Ja → En	En → Ja
Random	1.1	1.5
Frequency	381.0	432.6
LM	6.2	14.0

Table 7: Mean number of occurrences of selected n-grams in the test set

Type		+BLEU	+RIBES
Actual Error		0.55	0.41
Replacement	(Context dependent)	**0.36**	0.30
	(Context independent)	**0.15**	0
Insertion		0.17	**0.25**
Deletion		**0.18**	0.10
Reordering		0.14	**0.27**
Conjugation		0	**0.08**
Polarity		0	0
Unknown words		0	0

Table 8: Error statistics found when optimizing different metrics. Bold indicates the higher score.

LM optimized with BLEU and another with RIBES, which is a reordering-oriented evaluation metric. We show a breakdown of the identified errors in Table 8. From this table, we can see that the BLEU-optimized LM is able to detect more deletion errors than the RIBES-optimized LM. This is a natural result, as the BLEU metric puts a heavier weight on the brevity penalty assigned to shorter translations. On the other hand, the RIBES-optimized LM detects more reordering errors than the BLEU-optimized LM. The RIBES metric is sensitive to reordering errors, and thus reordering errors will cause larger decreases in RIBES. From this experiment, we can see that it is possible to focus on different error types by using different metrics in the optimization of the LM.

4.4 Result of System Comparison

Finally, we examine whether discriminative LMs allow us to grasp characteristic errors for system comparison. Similarly with single system analysis, we generated the top 30 potentially erroneous n-grams for PBMT, HIERO, and F2S in two directions, and evaluated them manually. The result is listed in Table 9. From this table, we can see that PBMT and HIERO count reordering errors as one of the three most frequent types, while F2S does not, especially for English to Japanese. This is consistent with common knowledge that syntactic information can be used to improve reordering accuracy. We can also see insertion is a problem when translating into English, and conjugation is a problem when translating into morphologically-rich Japanese. While these are only general trends, they largely match with intuition, even after analysis of only the top 30 n-grams.

5 Conclusion

In this paper, we proposed a new method for efficiently analyzing the output of MT systems using L1 regularized discriminative LMs, and evaluate its effectiveness. As a result, weights trained by discriminative LMs are more effective at identifying errors than n-grams chosen either randomly or by error frequency. This indicates that our method allows an MT system engineer to inspect fewer sentences in the course of identifying characteristic errors of the MT system.

The overall framework of using discriminative LMs in error analysis opens up a number of directions for future work, and there are a number of additional points we plan to analyze in the future. For example, while it is clear that the proposed method allows errors to be identified more efficiently, it is still necessary to quantify the overall benefit of having an MT expert use the result of this error analysis to improve

Type		Ja → En			En → Ja		
		PBMT	HIERO	F2S	PBMT	HIERO	F2S
Actual Error		0.58	0.60	0.55	0.81	0.64	0.48
Replacement	(Context dependent)	**0.41**	**0.33**	**0.36**	**0.10**	**0.17**	**0.52**
	(Context independent)	0.03	0.08	0.15	**0.55**	0.03	**0.12**
Insertion		**0.26**	**0.22**	**0.17**	0.06	0.13	**0.15**
Deletion		0.10	0.09	**0.18**	0.07	0.14	0.06
Reordering		**0.13**	**0.28**	0.14	**0.19**	**0.32**	0.04
Conjugation		0.07	0	0	0.04	**0.20**	**0.12**
Polarity		0	0	0	0	0.01	0
Unknown words		0	0	0	0	0	0

Table 9: Error statistics of three systems with in both directions. Bold scores are the top 3 most occuring error types in each system.

an MT system. In addition, we plan on examining the effect of using larger training data for the LM, incorporating different features based on POS patterns or syntactic features, and using more sophisticated training methods.

References

John Duchi and Yoram Singer. 2009. Efficient online and batch learning using forward backward splitting. In *Journal of Machine Learning Research*, volume 10.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proc. EMNLP*, pages 944–952.

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

Chin-Yew Lin and Franz Josef Och. 2004. Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Proc. COLING*, pages 501–507.

Graham Neubig. 2011. The Kyoto free translation task. `http://www.phontron.com/kftt`.

Graham Neubig. 2013. Travatar: A forest-to-string machine translation engine based on tree transducers. In *Proc. ACL*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL*, pages 311–318.

Maja Popovic and Hermann Ney. 2011. Towards automatic error analysis of machine translation output. In *Computational Linguistics*, pages 657–688.

Brian Roark, Murat Saraclar, and Michael Collins. 2007. Discriminative n-gram language modeling. *Computer Speech & Language*, 21(2):373–392.

Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. 2009. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proc. ACL*, pages 477–485.

David Vilar, Jia Xu, Luis Fernando D'Haro, and Hermann Ney. 2006. Error analysis of statistical machine translation output. In *Proc. LREC*, pages 697–702.

A Structured Language Model for Incremental Tree-to-String Translation

Heng Yu[1]
[1]Institute of Computing Technology. CAS
University of Chinese Academy of Sciences
yuheng@ict.ac.cn

Haitao Mi
T.J. Watson Research Center
IBM
hmi@us.ibm.com

Liang Huang
Queens College & Grad. Center
City University of New York
huang@cs.qc.cuny.edu

Qun Liu[1,2]
[2]Centre for Next Generation Localisation.
Faculty of Engineering and Computing
Dublin City University
qliu@computing.dcu.ie

Abstract

Tree-to-string systems have gained significant popularity thanks to their simplicity and efficiency by exploring the source syntax information, but they lack in the target syntax to guarantee the grammaticality of the output. Instead of using complex tree-to-tree models, we integrate a structured language model, a left-to-right shift-reduce parser in specific, into an incremental tree-to-string model, and introduce an efficient grouping and pruning mechanism for this integration. Large-scale experiments on various Chinese-English test sets show that with a reasonable speed our method gains an average improvement of 0.7 points in terms of (T -B)/2 than a state-of-the-art tree-to-string system.

1 Introduction

Tree-to-string models (Liu et al., 2006; Huang et al., 2006) have made promising progress and gained significant popularity in recent years, as they run faster than string-to-tree counterparts (e.g. (Galley et al., 2006)), and do not need binarized grammars. Especially, Huang and Mi (2010) make it much faster by proposing an incremental tree-to-string model, which generates the target translation exactly in a left-to-right manner. Although, tree-to-string models have made those progresses, they can not utilize the target syntax information to guarantee the grammaticality of the output, as they only generate strings on the target side.

One direct approach to handle this problem is to extend tree-to-string models into complex tree-to-tree models (e.g. (Quirk et al., 2005; Liu et al., 2009; Mi and Liu, 2010)). However, tree-to-tree approaches still significantly under-perform than tree-to-string systems due to the poor rule coverage (Liu et al., 2009) and bi-parsing failures (Liu et al., 2009; Mi and Liu, 2010).

Another potential solution is to use structured language models (S) (Chelba and Jelinek, 2000; Charniak et al., 2003; Post and Gildea, 2008; Post and Gildea, 2009), as the monolingual S has achieved better perplexity than the traditional *n*-gram word sequence model. More importantly, the S is independent of any translation model. Thus, integrating a S into a tree-to-string model will not face the problems that tree-to-tree models have. However, integration is not easy, as the following two questions arise. First, the search space grows significantly, as a partial translation has a lot of syntax structures. Second, hypotheses in the same bin may not be comparable, since their syntactic structures may not be comparable, and the future costs are hard to estimate. Hassan et al. (2009) skip those problems by only keeping the best parsing structure for each hypothesis.

In this paper, we integrate a shift-reduce parser into an incremental tree-to-string model, and introduce an efficient grouping and pruning method to handle the growing search space and incomparable hypotheses problems. Large-scale experiments on various Chinese-English test sets show that with a rea-

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1133–1143, Dublin, Ireland, August 23-29 2014.

sonable speed our method gains an average improvement of 0.7 points in terms of (T -B)/2 than a state-of-the-art tree-to-string system.

2 Linear-time Shift-reduce Parsing

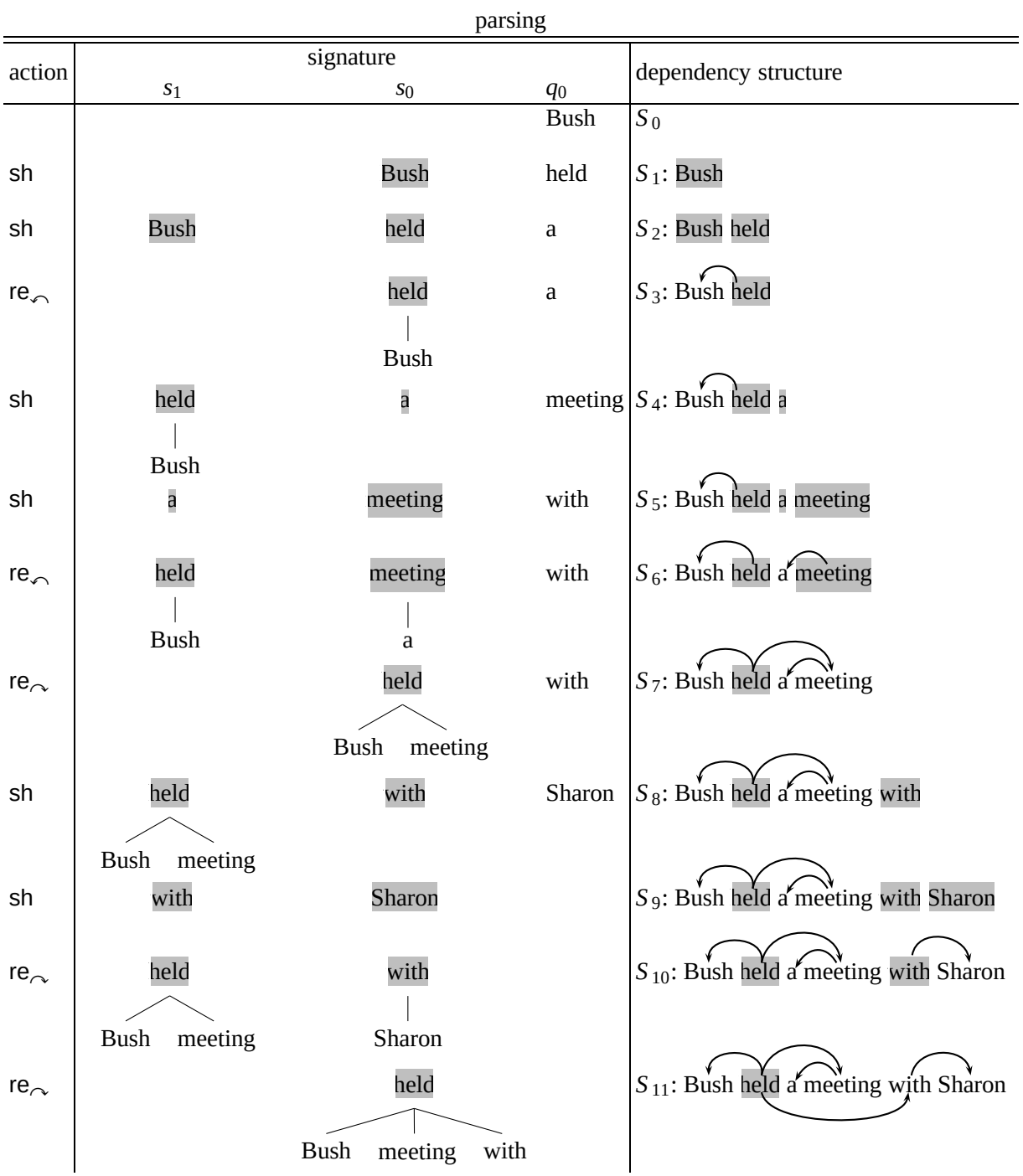


Figure 1: Linear-time left-to-right dependency parsing.

A shift-reduce parser performs a left-to-right scan of the input sentence, and at each *parsing step*, chooses one of two *parsing actions*: either **shift** (sh) the current word onto the stack, or **reduce** (re) the top two (or more) items at the end of the stack (Aho and Ullman, 1972). In the dependency parsing scenario, the reduce action is further divided into two cases: **left-reduce** (re$_\curvearrowleft$) and **right-reduce** (re$_\curvearrowright$), depending on which one of the two items becomes the head after reduction. Each parsing derivation can be represented by a sequence of parsing actions.

2.1 Shift-reduce Dependency Parsing

We will use the following sentence as the running example:

Bush held a meeting with Sharon

Given an input sentence **e**, where e_i is the ith token, $e_i...e_j$ is the substring of **e** from i to j, a shift-reduce parser searches for a dependency tree with a sequence of shift-reduce moves (see Figure 1). Starting from an initial structure S_0, we first shift (sh) a word e_1, "Bush", onto the parsing stack s_0, and form a structure S_1 with a singleton tree. Then e_2, "held", is shifted, and there are two or more structures in the parsing stack, we can use $\mathsf{re}_{\curvearrowleft}$ or $\mathsf{re}_{\curvearrowright}$ step to combine the top two trees on the stack, replace them with dependency structure $e_1 \curvearrowleft e_0$ or $e_1 \curvearrowright e_0$ (shown as S_3), and add one more dependency edge between e_0 and e_1.

Note that the shade nodes are exposed heads on which $\mathsf{re}_{\curvearrowleft}$ or $\mathsf{re}_{\curvearrowright}$ parsing actions can be performed. The middle columns in Figure 1 are the parsing signatures: q_0 (parsing queue), s_0 and s_1 (parsing stack), where s_0 and s_1 only have one level dependency. Take the line of S_{11} for example, "a" is not in the signature. As each action results in an update of cost, we can pick the best one (or few, with beam) after each action. Costs are accumulated in each step by extracting contextual features from the structure and the action. As the sentence gets longer, the number of partial structures generated at each steps grows exponentially, which makes it impossible to search all of the hypothesis. In practice, we usually use beam search instead.

(a)	atomic features	
	s_0.w	s_0.t
	s_1.w	s_1.t
	s_0.lc.t	s_0.rc.t
	q_0.w	q_0.t

(b)	feature templates		
unigram	s_0.w	s_0.t	s_0.w ∘ s_0.t
	s_1.w	s_1.t	s_1.w ∘ s_1.t
	q_0.w	q_0.t	q_0.w ∘ q_0.t
bigram	s_0.w ∘ s_1.w		s_0.t ∘ s_1.t
	s_0.t ∘ q_0.t		s_0.w ∘ s_0.t ∘ s_1.t
	s_0.w ∘ s_1.w ∘ s_1.t		s_0.t ∘ s_1.w ∘ s_1.t
	s_0.w ∘ s_0.t ∘ s_1.w		
trigram	s_0.t ∘ s_1.t ∘ q_0.t		s_1.t ∘ s_0.t ∘ s_0.lc.t
	s_1.t ∘ s_0.t ∘ q_0.t		s_1.t ∘ s_0.t ∘ s_0.rc.t

(c)

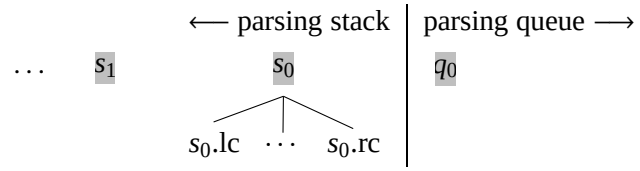


Table 1: (a) atomic features, used for parsing signatures. (b): parsing feature templates, adapted from Huang and Sagae (2010). x.w and x.t denotes the root word and POS tag of the partial dependency tree, x.lc and x.rc denote x's leftmost and rightmost child respectively. (c) the feature window.

2.2 Features

We view features as "abstractions" or (partial) observations of the current structure. Feature templates f are functions that draw information from the feature window, consisting of current partial tree and first word to be processed. All Feature functions are listed in Table 1(b), which is a conjunction of atomic

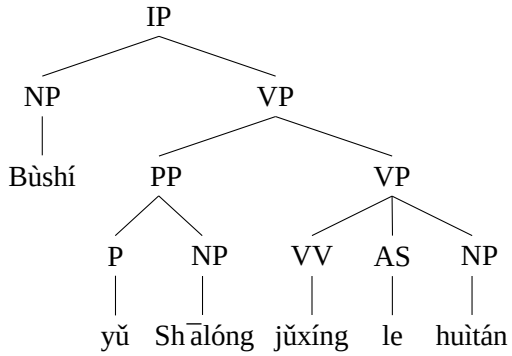


Figure 2: A parse tree

features in Table 1(a). To decide which action is the best of the current structure, we perform a three-way classification based on *f*, and conjoin these feature instances with each action:

$$[f \circ (\text{action=sh/re}_{\curvearrowright}/\text{re}_{\curvearrowleft})]$$

We extract all the feature templates from training data, and use the average perceptron algorithm and early-update strategy (Collins and Roark, 2004; Huang et al., 2012) to train the model.

3 Incremental Tree-to-string Translation with S

The incremental tree-to-string decoding (Huang and Mi, 2010) performs translation in two separate steps: parsing and decoding. A parser first parses the source language input into a 1-best tree in Figure 2, and the linear incremental decoder then searches for the best derivation that generates a target-language string in strictly left-to-right manner. Figure 3 works out the full running example, and we describe it in the following section.

3.1 Decoding with S

Since the incremental tree-to-string model generates translation in strictly left-to-right fashion, and the shift-reduce dependency parser also processes an input sentence in left-to-right order, it is intuitive to combine them together. The last two columns in Figure 3 show the dependency structures for the corresponding hypotheses. Start at the root *translation stack* with a dot **.** before the root node IP:

$$[\mathbf{.}\ \text{IP}\ ],$$

we first **predict** (pr) with rule r_1,

$$(r_1) \qquad \text{IP}\ (x_1\text{:NP}\ x_2\text{:VP}) \rightarrow\ x_1\ x_2,$$

and push its English-side to the translation stack, with variables replaced by matched tree nodes, here x_1 for NP and x_2 for VP. Since this *translation action* does not generate any translation string, we don't perform any dependency parsing actions. So we have the following translation stack

$$[\mathbf{.}\ \text{IP}\ ][\mathbf{.}\ \text{NP VP}],$$

where the dot **.** indicates the next symbol to process in the English word-order. Since node NP is the next symbol, we then predict with rule r_2,

$$(r_2) \qquad \text{NP(Bùshí)} \rightarrow \text{Bush},$$

and add it to the translation stack:

$$[\mathbf{.}\ \text{IP}\ ]\ [\mathbf{.}\ \text{NP VP}\ ]\ [\mathbf{.}\ \text{Bush}]$$

Since the symbol right after the dot in the top rule is a word, we **scan** (sc) it, and append it to the current translation, which results in the new translation stack

$$[\mathbf{.}\ \text{IP}\ ]\ [\mathbf{.}\ \text{NP VP}\ ]\ [\text{Bush}\ \mathbf{.}\ ]$$

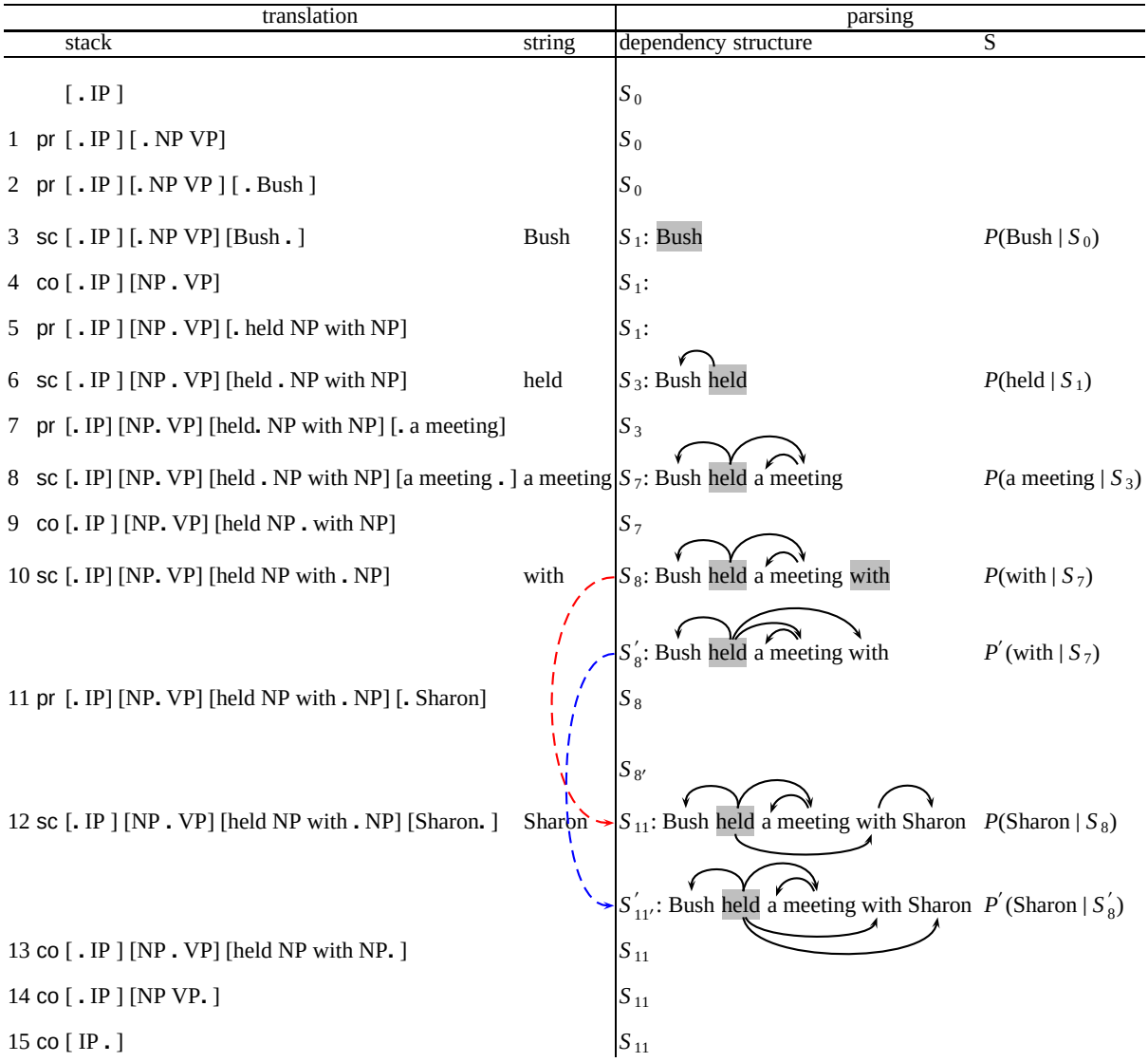


Figure 3: Simulation of the integraton of an S into an incremental tree-to-string decoding. The first column is the line number. The second column shows the translation actions: predict (pr), scan (sc), and complete (co). S_i denotes a dependency parsing structure. The shaded nodes are exposed roots of S_i.

Immediately after each sc translation action, our shift-reduce parser is triggered. Here, our parser applies the parsing action sh, and shift "Bush" into a partial dependency structure S_1 as a root "Bush" (shaded node) in Figure 3. Now the top rule on the translation stack has finished (dot is at the end), so we **complete** (co) it, pop the top rule and advance the dot in the second-to-top rule, denoting that NP is completed:

[. IP] [NP . VP].

Following this procedure, we have a dependency structure S_3 after we scan (sc) the word "held" and take a shift (sh) and a left reduce ($\text{re}_{\curvearrowleft}$) parsing actions. The shaded node "held" means exposed roots, that the shift-reduce parser takes actions on.

Following Huang and Mi (2010), the hypotheses with same *translation step*[1] fall into the same bin. Thus, only the prediction (pr) actions actually make a jump from a bin to another. Here line 2 to 4 fall into one bin (translation step = 4, as there are 4 nodes, IP, NP, VP and Bùshí, in the source tree are covered). Similarly, lines from 7 to 10 fall into another bin (translation step = 15).

[1]The step number is defined by the number of tree nodes covered in the source tree, and it is not equal to the number of translation actions taken so far.

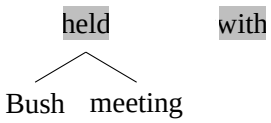
Noted that as we number the bins by the translation step, only pr actions make progress, the sc and co actions are treated as ”closure” operators in practice. Thus we always do as many sc/co actions as possible immediately after a pr step until the symbol after the dot is another non-terminal. The total number of bins is equal to the size of the parse tree, and each hypothesis has a constant number of outgoing hyper-edges to predict, so the time complexity is linear in the sentence length.

After adding our S to this translation, an interesting branch occurs after we scan the word “with”, we have two different partial dependency structures S_8 and S_8' for the same translation. If we denote $N(S_i)$ as the number of re actions that S_i takes, $N(S_8)$ is 3, while $N(S_8')$ is 4. Here $N(S_i)$ does not take into account the number of sh parsing actions, since all partial structures with same translations should shift the same number of translations. Thus, $N(S_i)$ determines the score of dependency structures, and only the hypotheses with same $N(S_i)$ are comparable to each other. In this case, we should distinguish S_8 with S_8', and if we make a prediction over the hypothesis of S_8, we can reach the correct parsing state S_{11} (shown in the red dashed line in Figure 3).

So the key problem of our integration is that, after each translation step, we will apply different sequences of parsing actions, which result in different and incomparable dependency structures with the same translation. In the following two Sections, we introduce three ways for this integration.

3.2 Naïve: Adding Parsing Signatures into Translation Signatures

One straightforward approach is to add the parsing signatures (in Figure 1) of each dependency structure (in Figure 1 and Figure 3) to translation signatures. Here, we only take into account of the s_0 and s_1 in the parsing stack, as the q_0 is the future word that is not available in translation strings. For example, the dependency structure S_8 has parsing signatures:



We add those information to its translation signature, and only the hypothesis that have same translation and parsing signatures can be recombined.

So, in each translation bin, different dependency structures with same translation strings are treated as different hypothesis, and all the hypothesis are sorted and ranked in the same way. For example, S_8 and S_8' are compared in the bin, and we only keep top b (the beam size) hypothesis for each bin.

Obviously, this simple approach suffers from the incomparable problem for those hypothesis that have different number of parsing actions (e.g. S_8 and S_8'). Moreover, it may result in very low translation variance in each beam.

3.3 Best-parse: Keeping the Best Dependency Structure for Each Translation

Following Hassan et al. (2009), we only keep the best parsing tree for each translation. That means after a consecutive translation sc actions, our shift-reduce parser applies all the possible parsing actions, and generates a set of new partial dependency structures. Then we only choose the best one with the highest S score, and only use this dependency structure for future predictions.

For example, for the translation in line 10 in Figure 3, we only keep S_8, if the parsing score of S_8 is higher than S_8', although they are not comparable. Another complicate example is shown in Figure 4, within the translation step 15, there are many alternatives with different parsing structures for the same translation (“a meeting with”) in the third column, but we can only choose the top one in the final.

3.4 Grouping: Regrouping Hypothesis by $N(S)$ in Each Bin

In order to do comparable sorting and pruning, our basic idea is to regroup those hypotheses in a same bin into small groups by $N(S)$. For each translation, we first apply all the possible parsing actions, and generate all dependency structures. Then we regroup all the hypothesis with different dependency structures based on the size of $N(S)$.

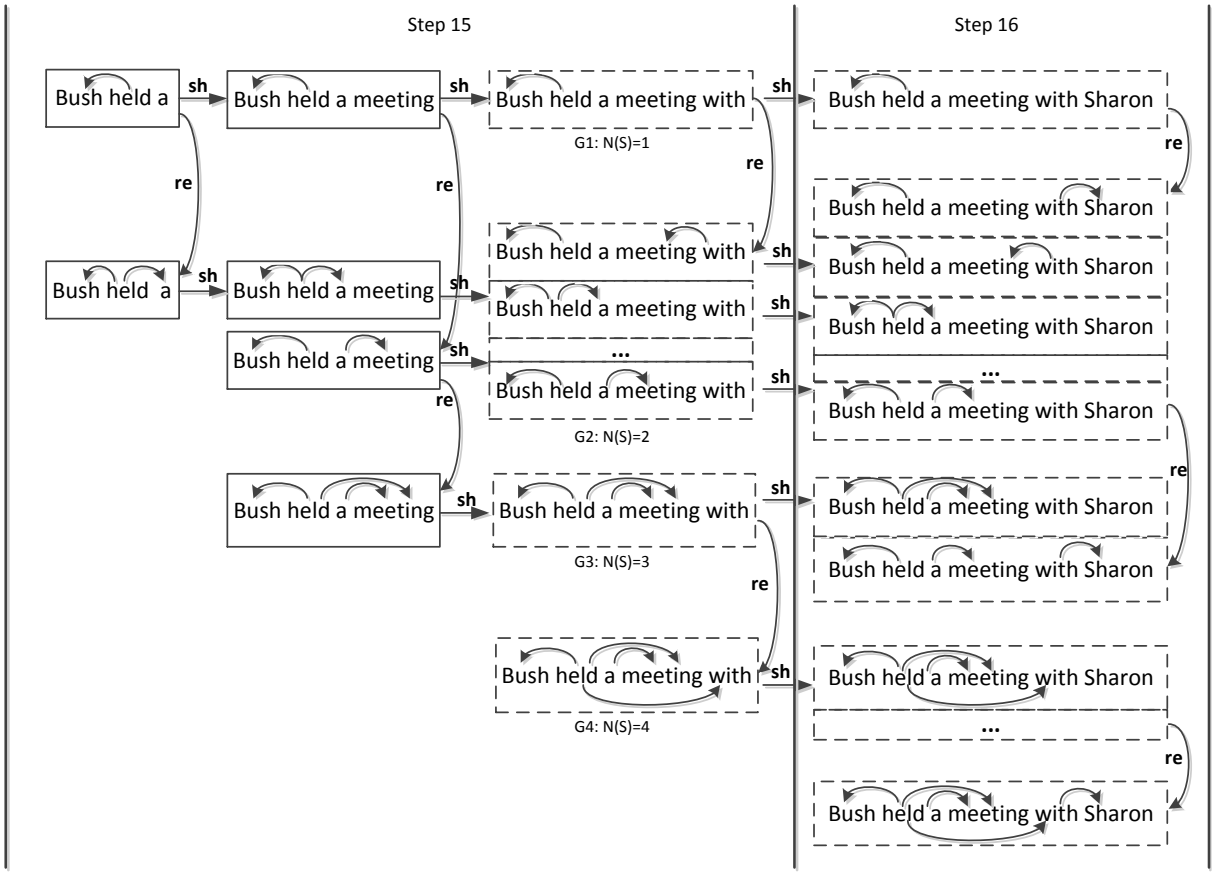


Figure 4: Multi-beam structures of two bins with different translation steps (15 and 16). The first three columns show the parsing movements in bin 15. Each dashed box is a group based on the number of reduce actions over the new translation strings ("a meeting with" for bin 15, and "Sharon" for bin 16). G_2 means two reduce actions have been applied. After this regrouping, we perform the pruning in two phases: 1) keep top b states in each group, and labeled each group with the state with the highest parsing score in this group; 2) sort the different groups, and keep top g groups.

For example, Figure 4 shows two bins with two different translation steps (15 and 16). In bin 15, the graph shows the parsing movements after we scan three new words ("a", "meeting", and "with"). The parsing sh action happens from a parsing state in one column to another state in the next column, while re happens from a state to another state in the same column. The third column in bin 15 lists some partial dependency structures that have all new words parsed. Here each dashed box is a group of hypothesis with a same $N(S)$, e.g. the G_2 contains all the dependency structures that have two reduce actions after parsed all the new words. Then, we sort and prune each group by the beam size b, and each group labeled as the highest hypothesis in this group. Finally, we sort those groups and only keep top g groups for the future predictions. Again, in Figure 4, we can keep the whole group G_3 and partial group of G_2 if $b = 2$. In our experiments, we set the group size g to 5.

3.5 Log-linear Model

We integrate our dependency parser into the log-linear model as an additional feature. So the decoder searches for the best translation $\mathbf{e}^*$ with a latent tree structure (evaluated by our S) according to the following equation:

$$\mathbf{e}^* = \underset{\mathbf{e} \in \mathbf{E}}{\operatorname{argmax}} \exp(\mathrm{S}(\mathbf{e}) \cdot w_s + \sum_i f_i \cdot w_i) \quad (1)$$

where S (**e**) is the dependency parsing score calculated by our parser, w_s is the weight of S (**e**), f_i are the features in the baseline model and w_i are the weights.

4 Experiments

4.1 Data Preparation

The training corpus consists of 1.5M sentence pairs with 38M/32M words of Chinese/English, respectively. We use the NIST evaluation sets of MT06 as our development set, and MT03, 04, 05, and 08 (newswire portion) as our test sets. We word-aligned the training data using GIZA++ with refinement option "grow-diag-and" (Koehn et al., 2003), and then parsed the Chinese sentences using the Berkeley parser (Petrov and Klein, 2007). we applied the algorithm of Galley et al. (2004) to extract tree-to-string translation rules. Our trigram word language model was trained on the target side of the training corpus using the SRILM toolkit (Stolcke, 2002) with modified Kneser-Ney smoothing. At decoding time, we again parse the input sentences using the Berkeley parser, and convert them into translation forests using rule pattern-matching (Mi et al., 2008).

Our baseline system is the incremental tree-to-string decoder of Huang and Mi (2010). We use the same feature set shown in Huang and Mi (2010), and tune all the weights using minimum error-rate training (Och, 2003) to maximize the B score on the development set.

Our dependency parser is an implementation of the "arc-standard" shift-reduce parser (Nivre, 2004), and it is trained on the standard split of English Penn Tree-bank (PTB): Sections 02-21 as the training set, Section 22 as the held-out set, and Section 23 as the test set. Using the same features as Huang and Sagae (2010), our dependency parser achieves a similar performance as Huang and Sagae (2010). We add the structured language model as an additional feature into the baseline system.

We evaluate translation quality using case-insensitive IBM B -4, calculated by the script `mteval-v13a.pl`. We also report the T scores.

4.2 Complete Comparisons on MT08

To explore the soundness of our approach, we carry out some experiments in Table 2. With a beam size 100, the baseline decoder achieves a B score of 21.06 with a speed of 1.7 seconds per sentence.

Since our dependency parser is trained on the English PTB, which is not included in the MT training set, there is a chance that the gain of B score is due to the increase of new *n*-grams in the PTB data. In order to rule out this possibility, we use the tool SRILM to train another tri-gram language model on English PTB and use it as a secondary language model for the decoder. The B score is 21.10, which is similar to the baseline result. Thus we can conclude that any gain of the following +S experiments is not because of the using of the additional English PTB.

Our second experiment re-ranks the 100-best translations of the baseline with our structured language model trained on PTB. The improvement is less than 0.2 B , which is not statistically significant, as the search space for re-ranking is relatively small compared with the decoding space.

As shown in Section 3, we have three different ways to integrate an S to the baseline system:

- **naïve**: adding the parsing signature to the translation signature;
- **best-parse**: keeping the best dependency structure for each translation;
- **grouping**: regrouping the hypothesis by $N(S)$ in each bin.

The naïve approach achieves a B score of 19.12, which is significantly lower than the baseline. The main reason is that adding parsing signatures leads to very restricted translation variance in each beam. We also tried to increase the beam size to 1000, but we do not see any improvement.

The fourth line in Table 2 shows the result of the best-parse (Hassan et al., 2009). This approach only slows the speed by a factor of two, but the improvement is not statistically significant. We manually looked into some dependency trees this approach generates, and found this approach always introduce local parsing errors.

The last line shows our efficient beam grouping scheme with a grouping size 5, it achieves a significant improvement with an acceptable speed, which is about 6 times slower than the baseline system.

System		B	Speed
baseline		21.06	1.7
+S	re-ranking	21.23	1.73
	naïve	19.12	2.6
	best-parse	21.30	3.4
	grouping (g=5)	**21.64**	10.6

Table 2: Results on MT08. The bold score is significantly better than the baseline result at level $p < 0.05$.

System	MT03		MT04		MT05		MT08		Avg.
	B	(T-B)/2	B	(T-B)/2	B	(T-B)/2	B	(T-B)/2	(T-B)/2
baseline	19.94	10.73	22.03	18.63	19.92	11.45	21.06	10.37	12.80
+S	**21.49**	**9.44**	22.33	18.38	**20.51**	**10.71**	**21.64**	**9.88**	**12.10**

Table 3: Results on all test sets. Bold scores are significantly better than the baseline system ($p < 0.5$).

4.3 Final Results on All Test Sets

Table 3 shows our main results on all test sets. Our method gains an average improvement of 0.7 points in terms of (T-B)/2. Results on NIST MT 03, 05, and 08 are statistically significant with $p < 0.05$, using bootstrap re-sampling with 1000 samples (Koehn, 2004). The average decoding speed is about 10 times slower than the baseline.

5 Related Work

The work of Schwartz et al. (2011) is similar in spirit to ours. We are different in the following ways. First, they integrate an S into a phrase-based system (Koehn et al., 2003), we pay more attention to a syntax-based system. Second, their approach slowdowns the speed at near 2000 times, thus, they can only tune their system on short sentences less than 20 words. Furthermore, their results are from a much bigger beam (10 times larger than their baseline), so it is not clear which factor contributes more, the larger beam size or the S . In contrast, our approach gains significant improvements over a state-of-the-art tree-to-string baseline at a reasonable speed, about 6 times slower. And we answer some questions beyond their work.

Hassan et al. (2009) incorporate a linear-time CCG parser into a DTM system, and achieve a significant improvement. Different from their work, we pay more attention to the dependency parser, and we also test this approach in our experiments. As they only keep 1-best parsing states during the decoding, they are suffering from the local parsing errors.

Galley and Manning (2009) adapt the maximum spanning tree (MST) parser of McDonald et al. (2005) to an incremental dependency parsing, and incorporate it into a phrase-based system. But this incremental parser remains in quadratic time.

Besides, there are also some other efforts that are less closely related to ours. Shen et al. (2008) and Mi and Liu (2010) develop a generative dependency language model for string-to-dependency and tree-to-tree models. But they need parse the target side first, and encode target syntactic structures in translation rules. Both papers integrate dependency structures into translation model, we instead model the dependency structures with a monolingual parsing model over translation strings.

6 Conclusion

In this paper, we presented an efficient algorithm to integrate a structured language model (an incremental shift-reduce parser in specific) into an incremental tree-to-string system. We calculate the structured language model scores incrementally at the decoding step, rather than re-scoring a complete translation. Our experiments suggest that it is important to design efficient pruning strategies, which have been

overlooked in previous work. Experimental results on large-scale data set show that our approach significantly improves the translation quality at a reasonable slower speed than a state-of-the-art tree-to-string system.

The structured language model introduced in our work only takes into account the target string, and ignores the reordering information in the source side. Thus, our future work seeks to incorporate more source side syntax information to guide the parsing of the target side, and tune a structured language model for both B and paring accuracy. Another potential work lies in the more efficient searching and pruning algorithms for integration.

Acknowledgments

We thank the three anonymous reviewers for helpful suggestions, and Dan Gildea and Licheng Fang for discussions. Yu and Liu were supported in part by CAS Action Plan for the Development of Western China (No. KGZD-EW-501) and a grant from Huawei Noah's Ark Lab, Hong Kong. Liu was partially supported by the Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the CNGL at Dublin City University. Huang was supported by DARPA FA8750-13-2-0041 (DEFT), a Google Faculty Research Award, and a PSC-CUNY Award, and Mi by DARPA HR0011-12-C-0015. The views and findings in this paper are those of the authors and are not endorsed by the US or Chinese governments.

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. Parsing of series in automatic computation. In *The Theory of Parsing, Translation, and Compiling*, page Volume I.

Eugene Charniak, Kevin Knight, and Kenji Yamada. 2003. Syntax-based language models for statistical machine translation. In *Proceedings of MT Summit IX. Intl. Assoc. for Machine Translation*.

Ciprian Chelba and Frederick Jelinek. 2000. Structured language modeling. volume 14, pages 283 – 332.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of ACL*.

Michel Galley and Christopher D. Manning. 2009. Quadratic-time dependency parsing for machine translation. In *Proceedings of the Joint Conference of ACL 2009 and AFNLP*, pages 773–781, Suntec, Singapore, August. Association for Computational Linguistics.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of HLT-NAACL*, pages 273–280.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of COLING-ACL*, pages 961–968.

Hany Hassan, Khalil Sima'an, and Andy Way. 2009. A syntactified direct translation model with linear-time decoding. In *Proceedings of EMNLP 2009*, pages 1182–1191, Singapore, August. Association for Computational Linguistics.

Liang Huang and Haitao Mi. 2010. Efficient incremental decoding for tree-to-string translation. In *Proceedings of EMNLP*, pages 273–283.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of ACL 2010*, pages 1077–1086, Uppsala, Sweden, July. Association for Computational Linguistics.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, pages 66–73.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of NAACL 2012*, Montreal, Quebec.

Philipp Koehn, Franz Joseph Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL*, pages 127–133.

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

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of COLING-ACL*, pages 609–616.

Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proceedings of ACL/IJCNLP*, pages 558–566, Suntec, Singapore, August.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP*, pages 523–530, Vancouver, British Columbia, Canada, October.

Haitao Mi and Qun Liu. 2010. Constituency to dependency translation with forests. In *Proceedings of ACL*, pages 1433–1442, Uppsala, Sweden, July.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proceedings of ACL: HLT*, pages 192–199.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In Frank Keller, Stephen Clark, Matthew Crocker, and Mark Steedman, editors, *Proceedings of the ACL Workshop Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain, July. Association for Computational Linguistics.

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

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

Matt Post and Daniel Gildea. 2008. Language modeling with tree substitution grammars. In *Proceedings of AMTA*.

Matt Post and Daniel Gildea. 2009. Language modeling with tree substitution grammars. In *Proceedings of NIPS workshop on Grammar Induction, Representation of Language, and Language Learning*.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of the 43rd ACL*, Ann Arbor, MI, June.

Lane Schwartz, Chris Callison-Burch, William Schuler, and Stephen Wu. 2011. Incremental syntactic language models for phrase-based translation. In *Proceedings of ACL 2011*, pages 620–631, June.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio, June. Association for Computational Linguistics.

Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*, volume 30, pages 901–904.

A Lexicalized Reordering Model for Hierarchical Phrase-based Translation

Hailong Cao[1], Dongdong Zhang[2], Mu Li[2], Ming Zhou[2] and Tiejun Zhao[1]
[1]Harbin Institute of Technology, Harbin, P.R. China
[2]Microsoft Research Asia, Beijing, P.R. China
{hailong, tjzhao}@mtlab.hit.edu.cn
{Dongdong.Zhang, muli, mingzhou}@microsoft.com

Abstract

Lexicalized reordering model plays a central role in phrase-based statistical machine translation systems. The reordering model specifies the orientation for each phrase and calculates its probability conditioned on the phrase. In this paper, we describe the necessity and the challenge of introducing such a reordering model for hierarchical phrase-based translation. To deal with the challenge, we propose a novel lexicalized reordering model which is built directly on synchronous rules. For each target phrase contained in a rule, we calculate its orientation probability conditioned on the rule. We test our model on both small and large scale data. On NIST machine translation test sets, our reordering model achieved a 0.6-1.2 BLEU point improvements for Chinese-English translation over a strong baseline hierarchical phrase-based system.

1 Introduction

In statistical machine translation, the problem of reordering source language into the word order of the target language remains a central research topic. Statistical phrase-based translation models (Och and Ney, 2004; Koehn et al., 2003) are good at local reordering, or the reordering of words within the phrase, since the order is specified by phrasal translations. However, phrase-based models remain weak at long-distance reordering, or the reordering of the phrases. To improve the reordering of the phrases, two types of models have been developed.

The first one is lexicalized reordering models (Tillman, 2004; Huang et al., 2005; Al-Onaizan and Papineni, 2006; Nagata et al., 2006; Xiong et al., 2006; Zens and Ney, 2006; Koehn et al., 2007; Galley and Manning, 2008; Cherry et al., 2012) which predict reordering by taking advantage of lexical information. The model in (Koehn et al., 2007) distinguishes three orientations with respect to the previous and the next phrase—monotone (*M*), swap (*S*) and discontinuous (*D*). For example, we can extract a phrase pair "xiayou ||| the lower reach of" whose orientations with respect to the previous and the next phrase are *D* and *S* respectively, as shown in Figure 1. Such a model is simple and effective, and has become a standard component of phrase-based systems such as MOSES.

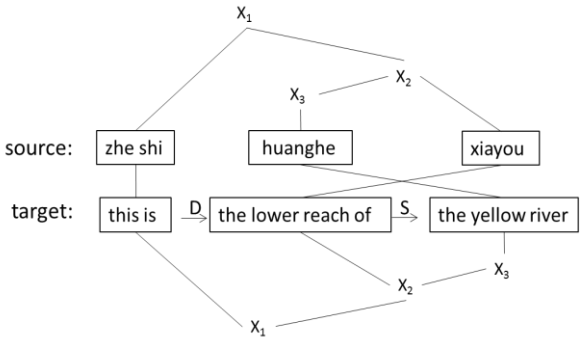


Figure 1. Phrase orientations for Chinese-English translation.

The other is a hierarchical phrase-based (HPB) translation model (Chiang, 2007) based on synchronous grammar. In the HPB model, a synchronous grammar rule may contain both terminals (words) and nonterminals (sub-phrases). The order of terminals and nonterminal are specified by the rule. For

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1144–1153, Dublin, Ireland, August 23-29 2014.

example, the translation rule <X xiayou, the lower reach of X > specifies that the translation of sub phrase X before "xiayou" should be put after "the lower reach of".

One problem with the HPB model is that the application of a rule is independent of the actual sub phrase. For example, the rule <X xiayou, the lower reach of X > will always swap the translation of X and "xiayou", no matter what is covered by X. This is an over-generalization problem. Much work has been done to solve this issue. For example, Zollmann and Venugopal (2006) annotate non-terminals by syntactic categories. He et al. (2008) proposes maximum entropy models which combine rich context information for selecting translation rules during decoding. Huang et al. (2010) automatically induce a set of latent syntactic categories to annotate nonterminals. These works alleviate the over-generalization problem by considering the content of X. In this paper, we try to solve it from an alternative view by modeling whether the phrases covered by X prefer the order specified by the rule. This has led us to borrow the lexicalized reordering model from the phrase-based model for the HPB model. We propose a novel lexicalized reordering model for hierarchical phrase-based translation and achieved a 0.6-1.2 BLEU point improvements for Chinese-English translation over a strong HPB baseline system.

2 Related work

In this section, we briefly review two types of related work which are a nonterminal-based lexicalized reordering models and a path-based lexicalized reordering model. Both of them calculate the orientation for HPB translation.

2.1 Nonterminal-based lexicalized reordering models

Xiao et al. (2011) proposed an orientation model for HPB translation. The orientation probability of a derivation is calculated as the product of orientation probabilities of all nonterminals except the root. In order to define the relative orders of nonterminals and their adjacent phrase, they expand the alignment in a rule to include both terminals and nonterminals. There may be multiple ways to segment a rule into phrases; they use the maximum adjacent phrase similar to Galley and Manning (2008). They significantly outperformed the HPB system on both Chinese-English and German-English translation.

Xiao et al. (2011) use the boundary word feature of nonterminals without considering their internal structure. For example, in Figure 1, suppose nonterminal X_1 is not the root node and the orientation probability of X_1 will condition on "zhe, xiayou, this, river".

In this paper, we will consider how the words covered by the nonterminal X_1 are reordered. Rather than using "xiayou" as a feature to determine the orientation of X_1 with respect to the next phrase, we think the immediately translated source word "huanghe" could be more informative through it is not on the boundary of X_1 , since "huanghe" is the exact starting point from where we search for the next phrase to translate.

Huck et al. (2013) proposed a very effective phrase orientation model for HPB translation. The model is also based on nonterminal. They extracted phrase orientation probabilities from word-aligned training data for use with hierarchical phrase inventories, and scored orientations in hierarchical decoding.

2.2 Path-based lexicalized reordering model

The most recent related work is Nguyen and Vogel (2013). They map a HPB derivation into a discontinuous phrase-based translation path in the following two steps:

1) Represent each rule as a sequence of phrase pairs and non-terminals.
2) The rules' sequences are used to find the corresponding phrase-based path of a HPB derivation and calculate the phrase-based reordering features.

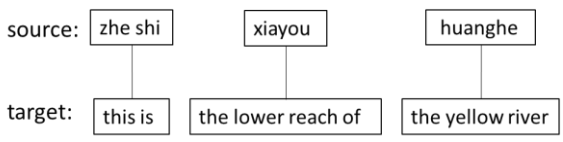


Figure 2. The phrase-based path of the derivation in Figure 1.

A phrase-based path is the sequence of phrase pairs, whose source sides covers the source sentences and whose target sides generated the target sentences from left to right. For example, the phrase-based path of the derivation in Figure 1 is shown in Figure 2.

The phrase-based reordering features for the above phrase-based path are:

$\log P_{next}(D \mid <\text{zheshi, this is}>)$, $\log P_{previous}(D \mid <\text{xiayou, the lower reach of}>)$,

$\log P_{next}(S \mid <\text{xiayou, the lower reach of}>)$, $\log P_{previous}(S \mid <\text{huanghe, the yellow river}>)$.

Nguyen and Vogel (2013) achieved significant improvement over both phrase-based and HPB models on three language pairs respectively.

One problem with the above work is that they did not use rules with unaligned source or target phrases. Though this can get faster and better Arabic-English translation, it leads to a 0.49 BLEU point loss for Chinese-English translation.

Another problem with path-based model is: there are many forms of HPB rules which we cannot map into a reasonable sequence of phrase pairs and non-terminals. We will show this with an example derivation shown in Figure 3. The main difference between Figure 3 and Figure 1 is there is such a rule <fangzhi X, prevent X from> that a source phrase "fangzhi" is aligned with a discontinuous target phrase "prevent…from". This makes it hard to find the corresponding phrase-based path because we do not know what is the right order of "fangzhi ||| prevent…from" and "daozei ||| the thieves" in the discontinuous phrase-based path. We face the following dilemmas:

- If "fangzhi ||| prevent…from" goes first, then the discontinuous phrase-based path is as shown in Figure 4(a). On such a path, we will consider the orientation of "the thieves" with respect to "breaking in". This is unreasonable because "the thieves" and "breaking in" are not adjacent in the target side. It does not satisfy the definition of the phrase-based reordering model which predicts the orientation with respect to previous or next adjacent target phrase.
- If "daozei ||| the thieves" goes first, then the discontinuous phrase-based path is as shown in Figure 4(b). This is unreasonable because "The policeman" and "the thieves" are not adjacent on the target side.

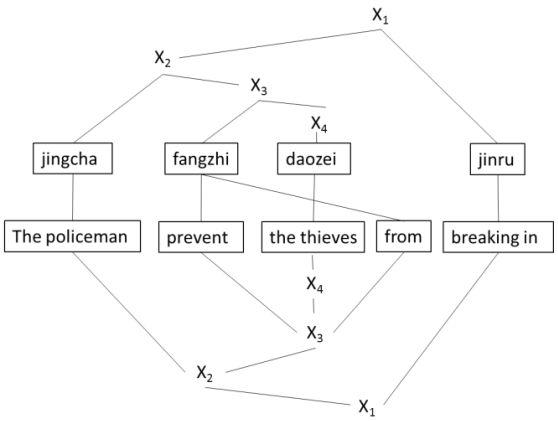


Figure 3. Example of Chinese-English translation and its derivation.

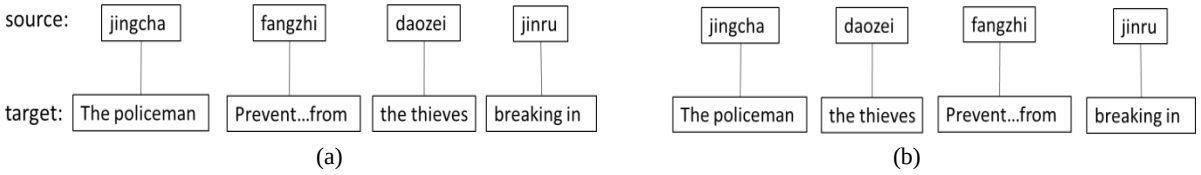


Figure 4. Two discontinuous phrase-based path candidates of the HPB derivation.

From the above example, we can see that if a target phrase is aligned to a discontinuous target phrase in a HPB rule, then it is hard to find a reasonable path whose target sides can generate the target sentence from left to right.

3 Our lexicalized reordering model

Rather than mapping a HPB derivation into a discontinuous phrase-based path and applying reordering model built on phrases, we propose a lexicalized reordering model which is built directly on HPB rules. For each target phrase contained in a HPB rule, we calculate its orientation probability conditioned on the rule. For the example derivation in Figure 3, we represent it by the structure shown in the following figure:

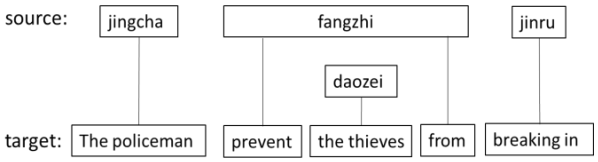


Figure 5. Our representation of the HPB derivation in Figure 3.

Different from Figure 4(a) and Figure 4(b) which contain a discontinuous phrase "prevent…from", we represent "prevent…from" as two individual target phrases: "prevent" and "from". Instead of considering the orientation of "prevent…from", we consider the orientation of "prevent" and "from" respectively. For example, we will consider the orientation of "prevent" with respect the previous phrase "the policeman" $O_{previous}(\text{prevent})$, and the orientation of "prevent" with respect the next phrase "the thieves" $O_{next}(\text{prevent})$. The probabilities of both $O_{previous}(\text{prevent})$ and $O_{next}(\text{prevent})$ are conditioned on the rule <fangzhi X, prevent X from>.

In Figure 5, every two neighboring target phrases are adjacent in the original target side. In this way, we can borrow the phrase-based reordering model which calculates the orientation with respect to previous and next adjacent phrase.

More formally, we represent a HPB rule in the general form of:

$$r = < s_0 \mathrm{X}_1 s_1 \mathrm{X}_2 s_2 ... \mathrm{X}_n s_n, t_0 \mathrm{X}_1 t_1 \mathrm{X}_2 t_2 ... \mathrm{X}_n t_n, \alpha >$$

where *n* is the number of nonterminals. $s_i, i = 1...n$, is the source phrase which is a continuous source word sequences. $t_i, i = 1...n$, is the target phrase which is a continuous target word sequences. We use α to represent the alignment of words and nonterminals in the rule. Note that s_i or t_i can be empty if there are adjacent nonterminals or there is nonterminal on the boundary. The lexicalized reordering probability of rule *r* is defined as the product of each target phrase's orientation probabilities conditioned on the rule *r*:

$$\prod_{i=0}^{n} P_{previous}(O_{previous}(t_i) \mid r, i) P_{next}(O_{next}(t_i) \mid r, i)$$

In the above equation, each probability is conditioned on the whole rule. In this way, we avoid the problem of mapping a HPB derivation into a discontinuous phrase-based path. There are two advantages for our reordering model:

- It is compatible with HPB rules which contain unaligned phrases.
- It is compatible with HPB rules in which a source phrase is aligned to a discontinuous target phrase.

Actually, our model is compatible with any kind of HPB rules since it is defined on the general form of rule.

Now we describe how to define $O_{previous}(t_i)$ and $O_{next}(t_i)$ in the model. Suppose t_i contains k_i target words and we write t_i as $t_i = w_{i(1)} w_{i(2)} ... w_{i(k_i\text{-}1)} w_{i(k_i)}$. Then we define:

$$O_{previous}(t_i) = O_{previous}(w_{i(1)}) = O(w_{i(1)\text{-}1}, w_{i(1)}), \qquad O_{next}(t_i) = O_{next}(w_{i(k_i)}) = O(w_{i(k_i)}, w_{i(k_i)+1})$$

where $O(w_j, w_{j+1})$ is the orientation of two adjacent target words and is determined as follows:

$$\begin{aligned}
&\text{If } (\ rm(w_j)+1 = lm(w_{j+1})\ ) && O(w_j, w_{j+1}) = M;\\
&\text{Else if } (\ rm(w_{j+1})+1 = lm(w_j)\ ) && O(w_j, w_{j+1}) = S;\\
&\text{Else} && O(w_j, w_{j+1}) = D;
\end{aligned}$$

$rm(w)$ is the position of the right most source word aligned to target word *w*; $lm(w)$ is the position of the left most source word aligned to target word *w*.

Above is our lexicalized reordering model which is built upon HPB rules. We complete its description using an example. For the rule <fangzhi X, prevent X from>, *n*=1, $t_0 = \text{“prevent”}$ and $t_1 = \text{“from”}$, the lexicalized reordering probability is:

$$P_{previous}(O_{previous}(\text{prevent})|\text{<fangzhi X, prevent X from>},0) \cdot P_{next}(O_{next}(\text{prevent})|\text{<fangzhi X, prevent X from>},0)$$
$$\cdot P_{previous}(O_{previous}(\text{from})|\text{<fangzhi X, prevent X from>},1) \cdot P_{next}(O_{next}(\text{from})|\text{<fangzhi X, prevent X from>},1)$$

Note that we calculate the orientation of plain phrase pairs in the same way as for HPB rules. We can represent a phrase pair in the form of $r = < s_0, t_0, \alpha >$, which is a rule that does not contain any nonterminal. Then we can apply our above model which is general enough to cover both HPB rules and plain phrase pairs.

4 Training and decoding

The training of our model is similar to the reordering model of Moses. During the standard phrase pair extraction and rule extraction, besides the nonterminal alignment in rules, we also keep the lexical alignments and orientations. If a phrase pair or a rule is observed with more than one set of alignment, we only keep the most frequent one and only count the orientations corresponding to the most frequent alignment.

Following Moses, we use relative frequency and add 0.5 smoothing technique to estimate the orientation probability based on all samples collected from the training corpus. Generally, given a rule *r* with *n* target phrases, we estimated the reordering probability for each t_i as follows:

$$P_{previous}(O_{previous}(t_i)|r,i) = \frac{0.5 + \#(O_{previous}(t_i),\ r)}{1.5 + \#(r)}, \qquad P_{next}(O_{next}(t_i)\,|\,r,i) = \frac{0.5 + \#(O_{next}(t_i), r)}{1.5 + \#(r)}$$

For each parallel sentences pair, we add a start and an end mark on both sides. They are aligned respectively.

Our phrase pairs and rules are extracted from word aligned parallel sentences. There are many phrase pairs and rules which contain unaligned target or source words. How to deal with them is quite important for our reordering model. We will describe how to process them in the following two subsections.

4.1 The processing of unaligned target words

Our main principle for processing an unaligned word is to: skip it and use the nearest aligned word. For example in Figure 3, the orientation of “prevent” with respect to the next phrase is determined by:

$$O_{next}(\text{prevent}) = O(\text{prevent}, \text{the})$$

If the target word “the” is unaligned and “thieves” is aligned with “daozei”, we will define:

$$O_{next}(\text{prevent}) = O(\text{prevent, the}) = O(\text{prevent, thieves}) = M$$

Similarly, in Figure 1, the orientation of “the lower reach of” with respect with “the yellow river” is determined by *O*(of, the). Suppose both “of” and “the” are unaligned and there are alignments for “reach-xiayou” and “yellow-huanghe”, we will have:

$$O(\text{of, the}) = O(\text{reach, yellow}) = S$$

We believe this orientation is consistent with our intuitions.

More formally, before we determine the orientation of two adjacent target words $O(w_p, w_q)$, we apply the following processing procedure:

While (target word w_p is unaligned) p--;

While (target word w_q is unaligned) q++;

If all words in a target phrase t_i are unaligned, we do not need to consider its orientation since t_i does not trigger any movement along the source words at all. Actually, it will be skipped when we determine the orientation of the previous and next aligned target phrases. (See also the decoding algorithm in Section 4.3)

4.2 The processing of unaligned source words

The processing of Section 4.1 can guarantee that the orientation is determined based on two aligned target words, namely w_p and w_q, which must be continuous or separated by unaligned target words.

Now we introduce the processing of unaligned source words. Before we determine the orientation of two target words $O(w_p, w_q)$, we apply the following procedure to modify the position index of the left most source word aligned to w_p and w_q respectively:

While (the $(lm(w_p)\text{-}1)^{\text{th}}$ source word is unaligned) $lm(w_p)$--;

While (the $(lm(w_q)\text{-}1)^{th}$ source word is unaligned) $lm(w_q)$--;

For the example shown in the Figure 6, initially we have $rm(w_1) = 1$ and $lm(w_4) = 4$. Since the source words w_3 and w_2 are unaligned, our procedure will modify the value of $lm(w_4)$ from 4 to 2. Finally, since $rm(w_1) + 1 = lm(w_4)$, the orientation of the two phrases marked by rectangular boxes in Figure 6 is:

$$O(w_2, w_3) = O(w_1, w_4) = M$$

Again, we believe this result is consistent with our intuition.

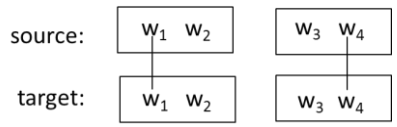


Figure 6. An example of phrases contain unaligned words

Note that during decoding, both the unaligned source and target words are also processed in the same way as in the training step. This makes our lexicalized reordering model consistent.

4.3 Decoding

Now we introduce how to integrate our reordering model into the HPB system during the standard CYK bottom-up decoding.

During decoding, if we just apply a plain phrase, we do not need to consider the orientation at once. It will be triggered when the phrase is used to compose a larger translation hypothesis together with other phrases or rules.

We need to calculate the reordering features whenever we apply a HPB rule or a glue rule during the CYK decoding. Generally, given a rule $r = < s_0 \mathrm{X}_1 s_1 \mathrm{X}_2 s_2 ... \mathrm{X}_n s_n, t_0 \mathrm{X}_1 t_1 \mathrm{X}_2 t_2 ... \mathrm{X}_n t_n, \alpha >$ defined in section 3, we calculate the reordering probability for the span covered by *r* with algorithm 1. In the algorithm, LL(X) represents the lowest rule which covers the left most word of X; LR(X) is the lowest

rule which covers the right most word of X; Both LL(X) and LR(X) can be found by traversing the derivation tree top to down recursively. LI(r) is the index of the last target phrase of rule *r*.

As in the example shown in Figure 3, for the rule $r_{2=}$<X_2 jinru, X_2 breaking in>, the orientation of X_2 and "breaking in" is:

$$O = O_{previous}(\text{breaking in}) = O(\text{from, breaking}) = D$$

The right most target word of X_2 is "from", the lowest rule covering "from" is $r_{3=}$<fangzhi X_4, prevent X_4 from> and the index of the last target phrase of r_3 is 1. So the reordering probability is:

$$prob = P_{previous}(D|r_1,0) \cdot P_{next}(D|r_3,1)$$

Note that, for readability, we use the product of probabilities to demonstrate the decoding process. Actually in practice, we use a linear model which sums the weighted log probabilities.

```
prob=1;
for (int i=1; i<=n; i++)
{
  if ( t_{i-1} is not empty and contains aligned words)
  {
      O = O_next(t_{i-1});
      prob* = P_next(O|r, i-1);
      prob* = P_previous(O|LL(X_i),0);
  }
  if ( t_i is not empty and contains aligned words)
  {
      O = O_previous(t_i);
      prob* = P_next(O|LR(X_i), LI(LR(X_i)));
      prob* = P_previous(O | r, i);
  }
  else if (i<n)
  {
      // X_i and X_{i+1} are continuous
      //or all words between them is unaligned
      rule r_p = LR(X_i);
      rule r_q = LL(X_{i+1});
      t = the first phrase of r_q;
      O = O_previous(t);
      prob* = P_next(O | r_p, LI(r_p))
      prob* = P_previous(O | r_q, 0);
      i++;
  }
}
```

Algorithm 1. Calculating the reordering probability for a span covered by a rule: $r = < s_0 X_1 s_1 X_2 s_2 ... X_n s_n, t_0 X_1 t_1 X_2 t_2 ... X_n t_n, \alpha >$.

As shown in Algorithm 1, the reordering probability depends on the lowest rules which cover the left/right most word. Therefore, we keep the lowest rules which cover the left/right most word for each partial translation. If two partial translations are same in everything but differ in the lowest rule, we need to keep both of them, rather than only keep the one with higher score. This will increase the complexity of the searching.

4.4 Discussion

Orientation can be determined based on word, phrase and hierarchical phrase (Galley and Manning, 2008). What we adopt in this paper is word based orientation. It is based on the following considerations:

- Our baseline is a HPB system, which can capture hierarchical orientation. We use word based orientation with the aim to complement the HPB system.
- Word based orientation is consistent during training and decoding; phrase based orientation is prone to inconsistent between training and decoding.

Galley and Manning (2008) has pointed out an inconsistency in Moses between training and decoding. Here we would like to note that phrase based orientation depends on phrase segmentation. For example, in Figure 1, the orientation of phrase "this is" with respect to next phrase could be either:

- D, if we think the next phrase is "the lower reach of " which is what Figure 1 shows.
- or S, if the next phrase is "the lower reach of the yellow river" which can compose a legal phrase pair with "huanghe xiayou" according to the standard phrase pair extraction algorithm.

The decision to adopt word-based orientation makes our work similar with Hayashi et al. (2010) who proposed a word-based reordering model for HPB system. The difference between our work and Hayashi et al. (2010) is: they adopt the reordering model proposed by Tromble and Eisner (2009) for the preprocessing approach, while we borrow the idea of lexicalized reordering models which are originally proposed for phrase-based machine translation.

5 Experiments

5.1 Experimental settings

Our baseline system is re-implementation of Hiero, a hierarchical phrase-based system (Chiang, 2007). Besides the standard features of a HPB model, there are six reordering features in our reordering model which are M, S and D with respect to the previous and next phrase respectively. They are integrated into the log-linear model of the HPB system. The Minimum Error Rate Training (MERT) (Och, 2003) algorithm is adopted to tune feature weights for translation systems.

We test our reordering model on a Chinese-English translation task. The NIST evaluation set MT06 was used as our development set to tune the feature weights, and the test data are MT04, MT 05 and MT08. We first conduct experiments by using the FBIS parallel training corpus, and then further test the effect of our method on a large scale parallel training corpus.

Word alignment is performed by GIZA++ (Och and Ney, 2000) in both directions with the default setting. The language model is a 4-gram model trained with the Xinhua portion of LDC English Gigaword Version 3.0 and the English part of the bilingual training data. Translation performances are measured with case-insensitive BLEU4 score (Papineni et al., 2002).

5.2 Experimental results on FBIS corpus

We first conduct experiments by using the FBIS parallel corpus to train the model of both the baseline and our lexicalized reordering model. After pre-processing, the statistics of FBIS corpus is shown in table 1.

	#sentences	#words
Chinese	128832	3016570
English	128832	3922816

Table 1. The statistics of FBIS corpus

Table 2 summarizes the translation performance. The first row shows the results of baseline HPB system, and the second row shows the results when we integrated our lexicalized reordering model (LRM). We get 1.2, 0.8 and 0.7 BLEU point improvements over the baseline HPB system on three test sets respectively.

	MT04	MT05	MT08
HPB	33.53	32.97	25.08
HPB+LRM	34.71	33.77	25.84

Table 2. Translation performance on the FBIS corpus.

5.3 Experimental results on large scale corpus

To further test the effect of our reordering model, we use a large scale corpus released by LDC. The catalog number of them is LDC2003E07, LDC2003E14, LDC2005T06, LDC2005T10, LDC2005E83, LDC2006E26, LDC2006E34, LDC2006E85 and LDC2006E92. There are 498K sentence pairs, 12.1M Chinese words and 13.8M English words. Table 3 summarizes the translation performance on the large scale of corpus.

	MT04	MT05	MT08
HPB	38.72	37.59	29.03
HPB+LRM	39.81	38.24	29.63

Table 3. Translation performance on a large scale parallel corpus.

Our model is still effective when we train the translation system on large scale data. We get 1.1, 0.7 and 0.6 BLEU point improvements over the baseline HPB system on three test sets respectively.

6 Conclusion and future work

We proposed a novel lexicalized reordering model for hierarchical phrase based machine translation. The model is compatible with any kind of HPB rules no matter how complex the alignments are. We tested our reordering model on both small and large scale data. On NIST machine translation test sets, our reordering model achieved a 0.6-1.2 BLEU point improvements for Chinese-English translation over a strong baseline hierarchical phrase-based system.

In future work, we will further test our model on other language pairs and compare it with other reordering models for HPB translation.

Acknowledgments

We thank anonymous reviewers for insightful comments. The work of Hailong Cao is sponsored by Microsoft Research Asia Star Track Visiting Young Faculty Program. The work of HIT is also funded by the project of National Natural Science Foundation of China (No. 61173073) and International Science & Technology Cooperation Program of China (No. 2014DFA11350).

Reference

Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion Models for Statistical Machine Translation. In Proceedings of *ACL*.

Colin Cherry, Robert C. Moore and Chris Quirk. 2012. On Hierarchical Re-ordering and Permutation Parsing for Phrase-based Decoding. In Proceedings of *NAACL Workshop on SMT*.

David Chiang. 2007. Hierarchical Phrase-based Translation. Computational Linguistics, 33(2):201–228.

Michel Galley and Christopher D. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. In Proceedings of *EMNLP*.

Katsuhiko Hayashi, Hajime Tsukada, Katsuhito Sudoh, Kevin Duh and Seiichi Yamamoto. 2010. Hierarchical Phrase-based Machine Translation with Word-based Reordering Model. In Proceedings of *COLING*.

Zhongjun He, Qun Liu, Shouxun Lin. 2008. Improving Statistical Machine Translation using Lexicalized Rule Selection. In Proceedings of *COLING*.

Liang Huang, Hao Zhang and Daniel Gildea. 2005. Machine Translation as Lexicalized Parsing with Hooks. In Proceedings of *IWPT*.

Zhongqiang Huang, Martin Čmejrek, and Bowen Zhou. 2010. Soft Syntactic Constraints for Hierarchical Phrase-based Translation Using Latent Syntactic Distributions. In Proceedings of *EMNLP*.

Matthias Huck, Joern Wuebker, Felix Rietig, and Hermann Ney. 2013. A Phrase Orientation Model for Hierarchical Machine Translation. In Proceedings of *ACL Workshop on SMT*.

Philipp Koehn, Hieu Hoang, Alexandra Birch Mayne, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst.. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL* demonstration session.

Masaaki Nagata, Kuniko Saito, Kazuhide Yamamoto and Kazuteru Ohashi. 2006. A Clustered Global Phrase Reordering Model for Statistical Machine Translation. In Proceedings of *ACL*.

Thuylinh Nguyen and Stephan Vogel. 2013. Integrating Phrase-based Reordering Features into Chart-based Decoder for Machine Translation. In Proceedings of *ACL*.

Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In Proceedings of *ACL*.

Franz Josef Och and Hermann Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. Computational Linguistics, 30(4):417–449.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In Proceedings of *ACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In Proceedings of *ACL*.

Christoph Tillmann. 2004. A Unigram Orientation Model for Statistical Machine Translation. In Proceedings of *HLT-NAACL*.

Roy Tromble, Jason Eisner. 2009. Learning Linear Ordering Problems for Better Translation. In Proceedings of *EMNLP*.

Xinyan Xiao, Jinsong Su, Yang Liu, Qun Liu, and Shouxun Lin. 2011. An Orientation Model for Hierarchical Phrase-based Translation. In Proceedings of *IALP*.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum Entropy Based Phrase Reordering Model for Statistical Machine Translation. In Proceedings of *ACL*.

Richard Zens and Hermann Ney. 2006. Discriminative Reordering Models for Statistical Machine Translation. In Proceedings of Workshop on *SMT*.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In Proceedings of *NAACL Workshop on SMT*.

Automatic Corpus Expansion for Chinese Word Segmentation by Exploiting the Redundancy of Web Information

Xipeng Qiu, ChaoChao Huang and Xuanjing Huang
Shanghai Key Laboratory of Intelligent Information Processing
School of Computer Science, Fudan University, Shanghai, China
xpqiu@fudan.edu.cn, superhuang007@gmail.com, xjhuang@fudan.edu.cn

Abstract

Currently most of state-of-the-art methods for Chinese word segmentation (CWS) are based on supervised learning, which depend on large scale annotated corpus. However, these supervised methods do not work well when we deal with a new different domain without enough annotated corpus. In this paper, we propose a method to automatically expand the training corpus for the out-of-domain texts by exploiting the redundant information on Web. We break up a complex and uncertain segmentation by resorting to Web for an ample supply of relevant easy-to-segment sentences. Then we can pick out some reliable segmented sentences and add them to corpus. With the augmented corpus, we can re-train a better segmenter to resolve the original complex segmentation. The experimental results show that our approach can more effectively and stably improve the performance of CWS. Our method also provides a new viewpoint to enhance the performance of CWS by automatically expanding corpus rather than developing complicated algorithms or features.

1 Introduction

Word segmentation is a fundamental task for Chinese language processing. In recent years, Chinese word segmentation (CWS) has undergone great development. The popular method is to regard word segmentation as a sequence labeling problems (Xue, 2003; Peng et al., 2004). The goal of sequence labeling is to assign labels to all elements in a sequence, which can be handled with supervised learning algorithms, such as Maximum Entropy (ME) (Berger et al., 1996), Conditional Random Fields (CRF)(Lafferty et al., 2001).

After years of intensive researches, Chinese word segmentation achieves a quite high precision. However, the performance of segmentation is not so satisfying for the practical demands to analyze Chinese texts. The key reason is that most of annotated corpora are drawn from news texts. Therefore, the system trained on these corpora cannot work well with the out-of-domain texts.

Since these supervised approaches often has a high requirement on the quality and quantity of annotated corpus, which is always not easy to create. As a result, many methods were proposed to utilize the information of unlabeled data.

There are three kinds of methods for domain adaptation problem in CWS.

The first is to use unsupervised learning algorithm to segment texts, like branching entropy (BE) (Jin and Tanaka-Ishii, 2006), normalized variation of branching entropy (nVBE)(Magistry and Sagot, 2012).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1154–1164, Dublin, Ireland, August 23-29 2014.

The second is to use unsupervised or domain-independent features in supervised learning for Chinese word segmentation, such as punctuation and mutual information(MI), word accessory variance (Feng et al., 2004; Zhao and Kit, 2008; Sun and Xu, 2011)

The third is to use semi-supervised learning (Zhu, 2005) in sequence labeling to address the difference in source and target distributions (Jiao et al., 2006; Altun et al., 2006; Suzuki and Isozaki, 2008).

Although these methods improve the performance of out-of-domain texts, the performance is still worse than that of in-domain texts obviously.

We firstly investigate the reasons of lower performance in new domain for state-of-the-art CWS systems and find that most of error segmentation were caused by out-of-vocabulary (OOV) words, also called new words or unknown words (see details in Section 3). It is difficult to devote efforts to building a corpus for out-of-domain texts, since new words are produced frequently as the development of the society, especially the Internet society. It is also impractical to manually maintain an up-to-date corpus to include all geographical names, person names, organization names, technical terms, etc.

In this paper, we propose a method to automatically expand the training corpus for the out-of-domain texts by exploiting the redundant information on Web. When we meet a complex and potentially difficult-to-segment sentence, we do not expect to solve it with more complicated learning algorithm or elaborate features. We assume that there are some relevant sentences that are relatively easy to process. These simple sentences can help to solve the complex one.

For example, the sentence “欧莱雅美宝莲 (L'Oreal, Maybelline)” is difficult to segment if both “欧莱雅 (L'Oreal)” and “美宝莲 (Maybelline)” are unknown words. However, we can always find some easy-to-segment sentences, such as “我使用美宝莲 (I use Maybelline)”, “欧莱雅的产品 (production of L'Oreal)”, and so on. When we use these simple sentences to re-train the segmenter, we can solve the previous complex sentence.

Our method relies on breaking up the complex problems into relevant smaller, simpler problems that can be solved easily. Fortunately, we can resort to the scale and redundancy of the web for an ample supply of simple sentences that are relatively easy to process.

Our method is very easy to implement upon a trainable base segmenter. Given the out-of-domain texts, we firstly choose some uncertain segmentations and select the candidate expansion seeds. Secondly, we use these seeds to get the relevant texts from Web search engine. Then we segment these texts and add the texts with high confidence to training corpus. Finally, we can get a better segmenter with the new corpus.

The rest of the paper is organized as follows: we review the related works in section 2. In section 3, we analyze the influence factor for CWS. Then we describe our method in section 4. Section 5 introduces the base segmenter. Section 6 gives the experimental results. Finally we conclude our work in section 7.

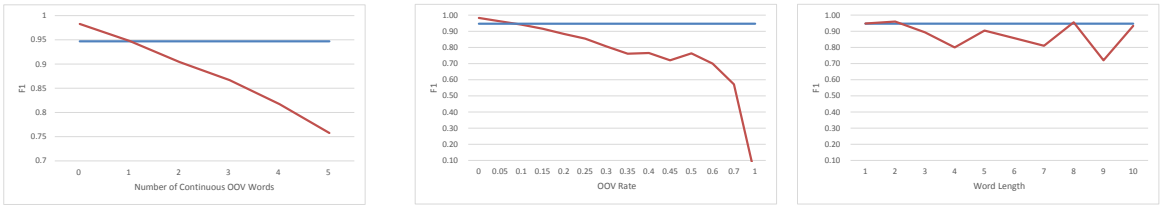
2 Related Works

The idea of exploring information redundancy on Web was introduced in question answering system (Kwok et al., 2001; Clarke et al., 2001; Banko et al., 2002) and the famous information extraction system KNOWITALL(Etzioni et al., 2004). However, this idea is rarely mentioned in Chinese word segmentation.

Nonetheless, there are three kinds of related methods on Chinese word segmentation.

One is active learning. Both (Li et al., 2012) and (Sassano, 2002) try to use active learning method to expand annotated corpus, but they still need to manually label some new raw texts in order to enlarge the training corpus. Different with these methods, our method do not require any manual oracle labeling at all.

Another is self-training, also called bootstrapping or self-teaching (Zhu, 2005). Self-training is a general semi-supervised learning approach. In self-training, a classifier is first trained with the small amount of labeled data. The classifier is then used to classify the unlabeled data.



(a) Number of continuous OOV words (b) OOV rate (c) Word Length

The blue horizontal line is the overall F1 score, and the red line is the F1 scores with different values of the factor.

Figure 1: Analysis of Influence Factors

Typically the most confident unlabeled points, together with their predicted labels, are added to the training set. The classifier is re-trained and the procedure repeated. Note that the classifier uses its own predictions to teach itself. Self-training has been applied to several natural language processing (NLP) tasks, such as word sense disambiguation (Yarowsky, 1995), POS-tagging (Clark et al., 2003; Jiang and Zhai, 2007; Liu and Zhang, 2012), parsing (Steedman et al., 2003; McClosky et al., 2006; Reichart and Rappoport, 2007; Sagae, 2010), information extraction(Etzioni et al., 2004)and so on. It has been proven that self-training can improve system performance on the target domain by simultaneously modeling annotated source-domain data and unannotated target domain data in the training process. However, the data on target domain cannot always help itself (Steedman et al., 2003).

The third is weakly supervised learning. (Li and Sun, 2009; Jiang et al., 2013) utilized the massive manual natural annotations or punctuation information on the Internet to improve the performance of CWS. However, these natural annotations are just partial annotations and their roles depend on the qualities of the selected resource, such as Wikipedia.

In this paper, we wish to propose a method to obtain new fully-annotated data in more aggressive way, which can combine the advantages of the above works.

3 Analysis of Influence Factors for CWS

Before describing our method, we give an analysis of the impact of out-of-vocabulary (OOV) words for segmentation. We first conduct experiments on the Chinese Treebank (CTB6.0) dataset (Xue et al., 2005) (The detailed information of dataset is shown in Section 6).

Table 1 shows the performance of base segmenter. The F1 score of OOV words is significantly lower than that of in-vocabulary (INV) words.

	Precision	Recall	F1
INV	95.86	96.58	96.21
OOV	74.12	66.77	70.25
Total	94.64	94.73	94.69

Table 1: Performances of INV and OOV words

We also investigate the impacts of three different factors: number of continuous OOV words, OOV rate and word length. Figure 1 shows the F1 scores with the changes of the different factors. We find that OOV words significantly improve the difficulty of segmentation, while the word length does not always harm the accuracy.

These findings also indicate that we can improve the performance of CWS if we have a dictionary or annotated corpus including these OOV words. With the redundancy of the Web information, it is not difficult to automatically obtain the expected dictionary or corpus.

4 Our Method

In this section, we describe our method to automatically expand the training corpus.

4.1 Framework of Automatic Corpus Expansion

Our framework of automatic corpus expansion is similar to standard process self-training or active learning for domain adaptation. Given a trainable base segmenter, the texts in out-of-domain, we firstly choose some uncertain segmentations and select the candidate expansion seeds. Secondly, we use these seeds to get the relevant texts from Web search engine. Then we segment these texts and add the texts with high confidence to training corpus. Finally, we can get a better segmenter with the new corpus.

Algorithm 1 illustrates the framework of automatic corpus expansion.

Algorithm 1 Framework of Automatic Corpus Expansion

Input:
Annotated Corpus C_A
Unannotated Corpus in Target domain C_T
Uncertainty Threshold T_u
Seed Extraction Threshold T_{se}
Acceptation Threshold T_a
Maximum Iteration Number: M

Output: Expanded Annotated Corpus C_A

1: **for** $i = 1$ to M **do**
2: Train a basic segmenter using current C_A with base learner
3: Use the basic segmenter to do segmentation for each sentence in C_T and calculate its confidence.
4: Choose out the sentences collection C_{TS}, in which the segmentation confidence of each sentence is less than T_u.
5: Extract the expansion seeds collection C_{seeds} from C_{TS} and use search engine to acquire relevant raw texts C_{RRT}.
6: Segment and calculate the confidence for each sentence in C_{RRT}.
7: Pick the reliable segmentations C_{new} with confidence more than T_a from C_{RRT}.
8: Add C_{new} into C_A.
9: **end for**
10: **return** C_A;

4.2 Uncertainty Sampling

The first key step in our method is to find the uncertain segmentations. There are many proposed uncertainty measures in the literature of active learning (Settles, 2010), such as entropy and query-by-committee (QBC) algorithm.

In our works, we investigate four following uncertainty measures for each sentence x. We use $S_1(x), S_2(x), \cdots, S_N(x)$ to represent the top N scores given by the segmenter.

Normalized Score U_{NS}

The first measures is normalized score by the length of x, the normalized score U_{NS} is calculated by

$$U_{NS} = \frac{S_1(x)}{L} \tag{1}$$

where L is the length of x.

Standard Deviation U_{SD}

The standard deviation is calculate with the top N scores.

$$U_{SD} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(S_i(x) - \mu)^2} \tag{2}$$

where $\mu = \frac{1}{N}\sum_{i=1}^{N} S_i(x)$ is the average or expected value of $S_i(x)$.

Entropy $U_{Entropy}$

Entropy is a measure of unpredictability or information content. Since we use character-based method for word segmentation, each character is labeled as one of {**B**, **M**, **E**, **S**} to indicate the segmentation. {**B**, **M**, **E**} represent *Begin*, *Middle*, *End* of a multi-character segmentation respectively, and **S** represents a *Single* character segmentation.

Given the top N labeled results for a sentence, each labeled sequence consists of the labels {**B**, **M**, **E**, **S**}. We define $l \in \{B, M, E, S\}$ to represent the label variable, and $\mathbf{count}_j(l)$ to be the number of occurrences of l on position j among the top N results. Thus, we can calculate the entropy for the labeling uncertainty of each character.

The entropy $H_j(l)$ for the character on position j is calculated by

$$H_j(l) = -\sum_{l} \frac{\mathbf{count}_j(l)}{N} \log \frac{\mathbf{count}_j(l)}{N}, \tag{3}$$

where $\sum_l \mathbf{count}_j(l) = N$.

The entropy of sentence $U_{Entropy}$ is the sum of the entropies of all the characters in the sentence.

$$U_{Entropy} = \sum_{j=1}^{L} H_j(l). \tag{4}$$

Margin U_{Margin}

Margin is the deviation of top 2 scores, which is often used in machine learning algorithms, such as support vector machine (Cristianini and Shawe-Taylor, 2000) and passive-aggressive algorithm (Crammer et al., 2006).

$$U_{Margin} = S_1(x) - S_2(x) \tag{5}$$

Among the above four measures, the larger the entropy is, the more uncertain the result is. For the rest three factors, the less the score is, the more uncertain the result is.

We test these four uncertainty measures on the development set in order to choose the best one as our confidence measure.

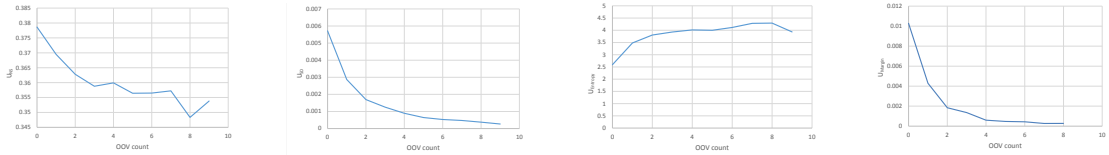
In figure 2, we illustrate the relationship between each uncertainty measure and the OOV count. We assume that the more OOV words are, the more uncertainty is. Meanwhile, a steep learning curve imply a good ability to distinguish whether the result is uncertain.

Obviously, the entropy is not helpful according to our assumption. The normalized score is okay but not good, and both the standard deviation and margin seem to be useful because they can give a better threshold to distinguish uncertain segmentation. Finally, we choose margin as our uncertainty measure.

4.3 Expansion Seeds Extraction

For the uncertain segmentation, not every word is unreliable. We just pick the suspicious fragments. Therefore, we need to extract some seed phrases to get the relevant texts. It is notable that these seed phrases do not need to be words. They can be the combinations of several words or only parts of words.

Take the following sentence for example.



(a) NScore (b) STDEV (c) Entropy (d) Margin

Figure 2: Different Uncertainty Measures

欧莱雅美宝莲兰蔻是很好的品牌
(L'Oreal, Maybelline, Lancome are good brands)

The first fragment "欧莱雅美宝莲兰蔻" is difficult to segment if these words does not appear in training corpus. Conversely, the second fragment is easy to segment since the containing words are very common.

We use base segmenter to get the top five results as follows:

	欧	莱	雅	美	宝	莲	兰	蔻	是	很	好	的	品	牌
1	B	M	M	M	E	B	M	E	S	B	E	S	B	E
2	B	M	M	E	B	E	B	E	S	B	E	S	B	E
3	B	M	E	B	E	B	M	E	S	B	E	S	B	E
4	S	B	M	M	E	B	E	S	S	B	E	S	B	E
5	S	B	M	M	M	E	B	E	S	B	E	S	B	E

(Li et al., 2012) proposed a good way to select the candidate words for active learning with diversity measurement to avoid duplicate annotation. However, their method is not suitable for our work. The reason is that they regarded CWS as a binary classification problem, while our base segmenter uses 1st-order sequence labeling.

In our work, we choose the expansion seeds by calculating the entropy of each character. If the entropy of the character is larger than threshold T_{se}, we say that this character may be in an uncertain context. Thus, we extract the consecutive uncertain characters and their contexts as the expansion seeds.

For the above example, we select the "欧莱雅美宝莲兰蔻 (L'Oreal, Maybelline, Lancome)" and its context "是 (is)" as a seed "欧莱雅美宝莲兰蔻是".

4.4 Collect relevance texts by using Web Search Engines

After obtaining the expansion seeds, we collect the relevant texts on multiple search engines including Google, Baidu and Bing.

For the seed "欧莱雅美宝莲兰蔻是", we can get the following relevance sentence, which is easy to segment.

欧莱雅拥有兰蔻、欧莱雅、美宝莲、薇姿等 500 多个品牌
(L'Oreal owns more than 500 brands, including Lancome, L'Oreal, Maybelline, Vichy, etc.)

In our work, we just get the top 100 relevant texts returned by each search engine without manual intervention. We do not use any search API and directly use the returned webpages by search engine, then extract the snippets and titles. Therefore, we just write a simple program to collect the webpages and clean them.

4.5 Expand Training Corpus

Since the qualities of these relevant texts are spotty, we just pick the reliable texts with high confidence scores. In contrast to uncertainty sampling, we find the certain segmentations from the collecting raw texts and add them to training corpus. Here, we also use a margin to find the reliable ones as new training data.

In our experiments, the number of selected sentence is $1 \sim 5$ for each seed.

Thus, we can re-train a new segmenter on the expanded corpus. After several iteration, we will get a segmenter with the best performance.

5 Base Segmenter

We use discriminative character-based sequence labeling for base word segmentation. Each character is labeled as one of {B, M, E, S} to indicate the segmentation.

We use online Passive-Aggressive (PA) algorithm (Crammer and Singer, 2003; Crammer et al., 2006) to train the model parameters. Following (Collins, 2002), the average strategy is used to avoid the overfitting problem.

6 Experiment

To evaluate our algorithm, we use both CTB6.0 and CTB7.0 datasets in our experiments. CTB is a segmented, part-of-speech tagged, and fully bracketed corpus in the constituency formalism. It is also a popular data set to evaluate word segmentation methods, such as (Sun and Xu, 2011). Since CTB dataset is collected from different sources, such as newswire, magazine, broadcast news and web blogs, it is suitable to evaluate the performance of CWS systems on different domains.

We conduct two experiments on different divisions of datasets.

1. The first experiment is performed on CTB6.0 for comparison with state-of-the-art systems which also utilize the unlabeled data for word segmentation.

2. The second experiment is performed on CTB7.0 for better evaluation on out-of-domain texts. CTB7.0 contains some newer news texts and web blogs texts, which is more suitable to evaluate our method for out-of-domain data.

In our experiments, we set $\mathcal{C} = 0.01$ for PA algorithm. We also try to use the different values of $\mathcal{C}$, and found that larger values of $\mathcal{C}$ imply a more aggressive update step and result to fast convergence, but it has little influence on the final accuracy. The maximum iteration number M' of PA algorithm is set to 50.

The feature templates are $C_iT_0, (i = -1, 0, 1)$,$C_{-1,0}T_0$, $C_{0,1}T_0, C_{-1,1}T_0$, $T_{-1,0}$. C represents a Chinese character, and the subscript of C indicates its position relative to the current character, whose subscript is 0. T represents the character-based tag.

The evaluation measure are reported are precision, recall, and an evenly-weighted F_1.

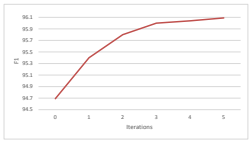
6.1 Experiments on CTB6.0

Train	Dev	Test
81-325, 400-454, 500-554, 590-596, 600-885, 900, 1001-1017, 1019, 1021-1035, 1037-1043, 1045-1059, 1062-1071, 1073-1078, 1100-1117, 1130-1131 1133-1140, 1143-1147, 1149-1151,2000-2139, 2160-2164, 2181-2279,2311-2549, 2603-2774, 2820-3079	41-80, 1120-1129, 2140-2159, 2280-2294, 2550-2569, 2775-2799, 3080-3109	(1-40,901-931 newswire) (1018, 1020, 1036, 1044,1060-1061, 1072, 1118-1119, 1132,1141-1142, 1148 magazine) (2165-2180, 2295-2310, 2570-2602, 2800-2819, 3110-3145 broadcast news)

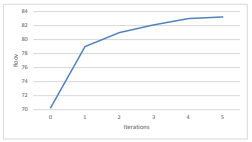
Table 2: CTB6.0 Dataset Division

On CTB 6.0, we divide the training, development and test sets according to (Yang and Xue, 2012). , which are shown in Table 2 The detailed statistical information is shown in Table 3.

Firstly, We use the development set to determine the parameters in Algorithm 1. For T_u, T_{se} and T_a, we have three rounds to determine the parameters. In first round, we find the best value $t1$ in the range to $0 \sim 1$ with the interval of 0.1. In second round, we find the best value $t2$ in range $t1 - 0.1 \sim t1 + 0.1$ with the interval of 0.01. In third round, we find the final best value $t3$

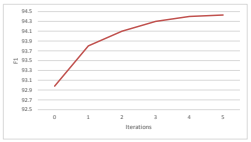


(a) F1 score

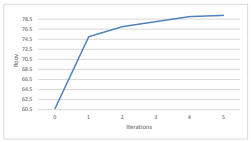


(b) OOV Recall

Figure 3: Iterative Learning Curve on CTB6.0



(a) F1 score



(b) OOV Recall

Figure 4: Iterative Learning Curve on CTB7.0

in the range to $t2 - 0.01 \sim t1 + 0.01$ with the interval of 0.001. The maximum iteration number M is just determined based on convergence with the range $1 \sim 10$.

Finally, we set these parameters as following: uncertainty threshold $T_u = 0.003$, seed extraction threshold $T_{se} = 0.65$, acceptation threshold $T_a = 0.004$ and maximum iteration number $M = 5$.

Figure 3 shows the changing curve of F1 and OOV recall in the process of corpus expansion. The performance of the baseline segmenter is shown at iteration 0. The curve shows that the F1 score and OOV recall have continuous improvement with the increasing of train corpus. The maximum performance is achieved at the $5th$ iteration. The detailed results are shown in Table 4. Compared with the baseline, the expanded corpus leads to a segmenter with significantly higher accuracy. The relative error reductions are 26.37% and 43.63% in terms of the balanced F-score and the recall of OOV words respectively.

Dataset	Sents	Words	Chars	OOV Rate
Train.	22757	639506	1053426	-
Dev.	2003	59764	100038	5.45%
Test	2694	81304	133798	5.58%

Table 3: Corpus Information of CTB 6.0

Test	P	R	F1	R_{oov}
Baseline	94.64	94.73	94.69	70.25
Final	95.66	96.51	96.09	83.23
(Sun and Xu, 2011)	95.86	95.62	95.74	79.28

Table 4: Performance on CTB6.0

6.2 Experiments on CTB7.0

CTB7.0 includes documents from newswire, magazine articles, broadcast news, broadcast conversations, newsgroups and weblogs. The newly added documents contains texts from web blogs, which is very different with news texts. Therefore, we use the documents (No. 4198 4411, weblogs) as test dataset, and the rest as training dataset. The detailed statistical information is shown in Table 5. We can see that the OOV rate is higher than the dataset in the first experiment.

Dataset	Sents	Words	Chars	OOV Rate
Train.	40425	987307	1601142	-
Test	10177	209827	342061	7.09%

Table 5: Corpus Information of CTB 7.0

Test	P	R	F1	R_{oov}
Baseline	93.58	92.40	92.98	60.72
Final	94.47	94.40	94.43	79.24

Table 6: Performance on CTB7.0

Figure 4 shows the changing curve of F1 and OOV recall in the process of corpus expansion. The performance of the baseline segmenter is shown at iteration 0. The curve shows that the F1 score and OOV recall have continuous improvement with the increasing of train corpus. The maximum performance is achieved at iteration 5. The detailed results are shown in Table 6. Compared with the baseline, the expanded corpus leads to a segmenter with significantly higher accuracy. The relative error reductions are 20.66% and 47.15% in terms of the balanced F-score and the recall of OOV words respectively.

6.3 Analysis

The experimental results show that our method is very effective to improve the performance of Chinese word segmentation. Especially, our method gives a significant boost on OOV words.

For the words such as "门兴格拉德巴赫 (Borussia Moenchengladbach)", "过氧化氢酶 (catalase)", "易中天 (Yi ZhongTian, a Chinese person name)" and "黄金档 (prime time)", it is still difficult to segment them correctly even if we can obtain useful features from unlabeled data. When we take advantage of the redundant information from Web, we can easily collect the relevant easy-to-segment sentences to expand the training corpus.

Our method can result to a segmenter significantly better than the systems which finds the informative features derived from unlabeled data, such as (Sun and Xu, 2011). This also suggests that expanding corpus is more effective than developing complicated algorithm or well-design features. Of course, our method is compatible with these technologies, which can further improve the performance of CWS by combining the Web redundancy.

7 Conclusion

In this paper, we propose a method to automatically expand the training corpus for the out-of-domain texts. Given the out-of-domain texts, we first choose some uncertain segmentations as candidate expansion seeds, and use these seeds to get the relevant texts from search engine. Then we segment the texts and add the texts with high confidence to training corpus. We can always obtain some easily-segmented texts due to the large amount of redundancy texts on Web, especially for new words. Our experimental results show that our proposed method can more effectively and stably utilize the unlabeled examples to improve the performance. Our method also provides a new viewpoint to enhance the performance of CWS by expanding corpus rather than developing complicated algorithms or features.

The long term goal of our method is to build an online and constant learning system, which can identify the difficult tasks and seek help from crowdsourcing. Search engines are special cases of crowdsourcing. In the future, we wish to investigate our method for other NLP tasks, such as POS tagging, Named Entity Recognition, and so on.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. This work was funded by NSFC (No.61003091), Science and Technology Commission of Shanghai Municipality (14ZR1403200) and Shanghai Leading Academic Discipline Project (B114).

References

Y. Altun, D. McAllester, and M. Belkin. 2006. Maximum margin semi-supervised learning for structured variables. *Advances in neural information processing systems*, 18:33.

Michele Banko, Eric Brill, Susan Dumais, and Jimmy Lin. 2002. AskMSR: Question answering using the worldwide web. In *Proceedings of 2002 AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases*, pages 7–9.

A.L. Berger, V.J. Della Pietra, and S.A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Stephen Clark, James R Curran, and Miles Osborne. 2003. Bootstrapping POS taggers using unlabelled data. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 49–55. Association for Computational Linguistics.

C.L.A. Clarke, G.V. Cormack, and T.R. Lynam. 2001. Exploiting redundancy in question answering. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 358–365.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*.

K. Crammer and Y. Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

N. Cristianini and J. Shawe-Taylor. 2000. *An introduction to support Vector Machines: and other kernel-based learning methods*. Cambridge Univ Pr.

Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2004. Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 100–110, New York, NY, USA. ACM.

H. Feng, K. Chen, X. Deng, and W. Zheng. 2004. Accessor variety criteria for chinese word extraction. *Computational Linguistics*, 30(1):75–93.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *ACL*, volume 2007, page 22.

Wenbin Jiang, Meng Sun, Yajuan Lü, Yating Yang, and Qun Liu. 2013. Discriminative learning with natural annotations: Word segmentation as a case study. In *ACL*, pages 761–769.

Feng Jiao, Shaojun Wang, Chi-Hoon Lee, Russell Greiner, and Dale Schuurmans. 2006. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 209–216. Association for Computational Linguistics.

Zhihui Jin and Kumiko Tanaka-Ishii. 2006. Unsupervised segmentation of Chinese text by use of branching entropy. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 428–435. Association for Computational Linguistics.

C.C.T. Kwok, O. Etzioni, and D.S. Weld. 2001. Scaling question answering to the web. *Proceedings of the 10th international conference on World Wide Web*, pages 150–161.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.

Zhongguo Li and Maosong Sun. 2009. Punctuation as implicit annotations for chinese word segmentation. *Computational Linguistics*, 35(4):505–512.

Shoushan Li, Guodong Zhou, and Chu-Ren Huang. 2012. Active learning for Chinese word segmentation. In *COLING (Posters)*, pages 683–692.

Yang Liu and Yue Zhang. 2012. Unsupervised domain adaptation for joint segmentation and pos-tagging. In *COLING (Posters)*, pages 745–754.

Pierre Magistry and Benoît Sagot. 2012. Unsupervized word segmentation: the case for mandarin chinese. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 383–387. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*, pages 152–159. Association for Computational Linguistics.

F. Peng, F. Feng, and A. McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. *Proceedings of the 20th international conference on Computational Linguistics*.

Roi Reichart and Ari Rappoport. 2007. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 616–623, Prague, Czech Republic, June. Association for Computational Linguistics.

Kenji Sagae. 2010. Self-training without reranking for parser domain adaptation and its impact on semantic role labeling. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 37–44. Association for Computational Linguistics.

Manabu Sassano. 2002. An empirical study of active learning with support vector machines for japanese word segmentation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 505–512. Association for Computational Linguistics.

Burr Settles. 2010. Active learning literature survey. *University of Wisconsin, Madison.*

Mark Steedman, Miles Osborne, Anoop Sarkar, Stephen Clark, Rebecca Hwa, Julia Hockenmaier, Paul Ruhlen, Steven Baker, and Jeremiah Crim. 2003. Bootstrapping statistical parsers from small datasets. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 331–338. Association for Computational Linguistics.

Weiwei Sun and Jia Xu. 2011. Enhancing Chinese word segmentation using unlabeled data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 970–979. Association for Computational Linguistics.

Jun Suzuki and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *ACL*, pages 665–673. Citeseer.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The Penn Chinese TreeBank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(2):207–238.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing*, 8(1):29–48.

Yaqin Yang and Nianwen Xue. 2012. Chinese comma disambiguation for discourse analysis. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 786–794. Association for Computational Linguistics.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics.

H. Zhao and C. Kit. 2008. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In *The Sixth SIGHAN Workshop on Chinese Language Processing*, pages 106–111. Citeseer.

Xiaojin Zhu. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.

Fast High-Accuracy Part-of-Speech Tagging by Independent Classifiers

Robert C. Moore
Google Inc.
bobmoore@google.com

Abstract

Part-of-speech (POS) taggers can be quite accurate, but for practical use, accuracy often has to be sacrificed for speed. For example, the maintainers of the Stanford tagger (Toutanova et al., 2003; Manning, 2011) recommend tagging with a model whose per tag error rate is 17% higher, relatively, than their most accurate model, to gain a factor of 10 or more in speed. In this paper, we treat POS tagging as a single-token independent multiclass classification task. We show that by using a rich feature set we can obtain high tagging accuracy within this framework, and by employing some novel feature-weight-combination and hypothesis-pruning techniques we can also get very fast tagging with this model. A prototype tagger implemented in Perl is tested and found to be at least 8 times faster than any publicly available tagger reported to have comparable accuracy on the standard Penn Treebank Wall Street Journal test set.

1 Introduction

Part-of-speech (POS) tagging remains an important basic task in natural-language processing, often being used as an initial step in addressing more complex problems such as parsing (e.g., McDonald et al., 2005) or named-entity recognition (e.g., Florian et al., 2003). State-of-the-art-taggers typically employ discriminatively-trained models with hidden tag-sequence features. These models include features of the observable input sequence, plus hidden features consisting of tag sequences up to some fixed length.

With a tag-sequence model, the highest scoring tagging for an input sentence can be found by the Viterbi algorithm, but exact search can be slow with a large tag set. If tri-tag features are used, the full search space is $O(|T|^3 n)$, where $|T|$ is the size of the tag set and n is the length of the sentence. For the English Penn Treebank (Marcus et al., 1993) , $|T| = 45$, hence $|T|^3 = 91125$. For efficiency, some form of approximate search is normally used. For example, both Shen et al. (2007) and Huang et al. (2012) use approximate search in both training and tagging. Shen et al. use a specialized bi-directional beam search in which the search order is learned at training time and applied at tagging time, along with the model. Huang et al. use a more conventional left-to-right beam search, but they explore various special variants of the perceptron algorithm to cope with search errors during model training. These two taggers represent the current state of the art on the Penn Treebank Wall Street Journal (WSJ) corpus, for models trained using no additional resources, as measured on the standard training/development/test data split introduced by Collins (2002a): 2.67% per tag error for Shen et al., and 2.65% for Huang et al.

Alternatively, one may omit hidden tag-sequence features, enrich the set of observable features, and treat tagging each token as an independent multi-class classification problem. Toutanova et al. (2003) were the first to note that such models could achieve fairly high accuracy for POS tagging, reporting per-tag error of 3.43% on the standard WSJ development set. Liang et al. (2008) report 3.2% error on the standard WSJ test set (using a slightly smaller than standard training set), which as far as we know is the current state of the art for WSJ POS tagging by independent classifiers. The independent classifier approach has the advantage of a simple model structure with a search space for tagging of $O(|T|n)$. On the other hand, while Liang et al.'s result would have been state-of-the-art before Collins (2002a), today

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1165–1176, Dublin, Ireland, August 23-29 2014.

it represents an error rate about 20% higher than Huang et al.'s best result for tri-tag-based POS tagging, under similar training conditions.

In the first part of this paper, we introduce new features for tagging by independent classifiers. We introduce case-insensitive versions of several standard types of features, which enables our models to generalize over different casings of the same underlying word. We also cluster the vocabulary of the annotated training set, preserving as much information as possible about the tag probabilities for each word, and use sequences of the resulting classes to approximate the contextual information provided by hidden tri-tag features. With the further addition of another set of word-class features based on distributional similarity over a large corpus of unnanotated data, we obtain a model with a WSJ test set error of 2.66% (97.34% accuracy).

In the remainder of the paper, we show how to perform fast tagging with this model. Even with the simple structure of an independent multiclass classifer, tagging can be slow with a rich model and a large tag set, simply because feature extraction and model scoring take so much time. We address this in two ways. First we effectively reduce the number of features that have to be considered for a given token by combining the feature weights for more general features into those for more specific features. For example, if a word is in the training set vocabulary, none of its sublexical features need to be extracted or scored, if the weights of those features have already been combined into the weights for the corresponding "whole word" feature. Second, we limit the number of tags considered for each token by a pruning method that refines Ratnaparkhi's (1996) tag dictionary, employing a Kneser-Ney-smoothed probability distribution over the possible tags for each word, and applying a threshold tuned to reduce the number of tags considered while minimizing loss of accuracy. We have implemented a prototype tagger in Perl using these methods, which we find to be at least 8 times faster than any of the publicly available taggers reported to have comparable accuracy on the standard WSJ test set.

2 Models for Tagging by Independent Classifiers

We formulate the POS-tagging task as a linear multiclass classification problem defined by a set of tags $\mathcal{T}$ and a set of indicator features $\mathcal{F}$. Each training example consists of a set of features $\mathbf{f} \subseteq \mathcal{F}$ present in that example and a correct tag $t \in \mathcal{T}$. The feature set $\mathbf{f}$ for a particular example consists of observable properties of the token to be tagged and the tokens surrounding it. A model is a vector $\mathbf{w} \in \Re^{|\mathcal{T}|\times|\mathcal{F}|}$ indexed by feature-tag pairs. We refer to the coordinates $w_{(f,t)}$ of $\mathbf{w}$ as *feature weights*. A model $\mathbf{w}$ maximizes the sum of relevant feature weights to predict a tag $t(\mathbf{f}, \mathbf{w})$:

$$t(\mathbf{f}, \mathbf{w}) = \arg\max_{t \in \mathcal{T}} \sum_{f \in \mathbf{f}} w_{(f,t)} \tag{1}$$

In the remainder of this section we explain the feature sets we use and our method of training feature weights, and we evaluate the accuracy of the resulting models on the usual Wall Street Journal corpus from Penn Treebank III (Marcus et al., 1993).

2.1 Lexical Features

As noted above, the current state of the art for tagging by independent classifiers seems to be the results presented by Liang et al. (2008). Their best model uses the following set of base features for each word:

Whether the first character of the word is a capital letter
Prefixes of the word up to three characters
Suffixes of the word up to three characters
Two "shape" features described below
The full word

For each base feature, Liang et al. define three expanded features: whether the token being tagged has the base feature, whether the preceding token has the base feature, and whether the following token has the base feature. The shape features were first introduced by Collins (2002b) for named-entity recognition. What we will call the "Shape 1" feature is a generalization of the spelling of the word with all capital

letters treated as equivalent, all lower-case letters treated as equivalent, and all digits treated as equivalent. All other characters are treated as distinct. In the "Shape 2" feature, all sequences of capital letters, all sequences of lower case letters, and all sequences of digits are treated as equivalent, regardless of the length of the sequence or the identity of the upper case letters, lower case letters, or digits.

With this feature set as our starting point, and partially drawing from the feature sets of Ratnaparkhi (1996) and Collins (2002a), we settled on the following set of base features through experimentation on the WSJ development set:

Whether the word contains a capital letter
Whether the word contains a digit
Whether the word contains a hyphen
Lower-cased prefixes of the word up to four characters
Lower-cased suffixes of the word up to four characters
The Shape 1 feature for the word
The Shape 2 feature for the word
The full lower-cased word
The full word
A distributional-similarity-based class for the full word

In *all* these features we ignore distinctions among digits (rather than just in the shape features, as Liang et al. do). For the last feature, we used 256 word classes derived by unsupervised clustering for the most frequent 999996 distinct tokens (ignoring distinctions among digits) in 121.6 billion tokens of English-language newswire, using the method of Uszkoreit and Brants (2008). A 257th class was added for tokens not found in this set. We use Liang et al.'s mapping of all base features into expanded features for the token being tagged, the preceding token, and the following token. For the first token of a sentence we include a beginning-of-sentence feature in place of the preceding-token features, and for the last token of a sentence we include an end-of-sentence feature in place of the following-token features.

2.2 Word-Class-Sequence Features

In a hidden tri-tag model, the prediction for a particular tag t_i is linked to the predictions for the preceding tag t_{i-1}, the following tag t_{i+1}, the preceding tag pair $\langle t_{i-2}, t_{i-1} \rangle$, the following tag pair $\langle t_{i+1}, t_{i+2} \rangle$, and the surrounding tag pair $\langle t_{i-1}, t_{i+1} \rangle$. In tagging by independent classifiers, we do not have access to information regarding predictions for these nearby tags and tag combinations.

To substitute for these missing features, we carry out supervised clustering of the distinct words in the training set (again ignoring distinctions among digits) into 50 classes, attempting to maximize the information carried by each class regarding the tag probabilities for the words in the class. From these classes, we construct the features

$c(w_{i-1})$
$c(w_{i+1})$
$\langle c(w_{i-2}), c(w_{i-1}) \rangle$
$\langle c(w_{i+1}), c(w_{i+2}) \rangle$
$\langle c(w_{i-1}), c(w_{i+1}) \rangle$

The type of clustering we use here differs from the unsupervised clustering described previously. In assigning each word to a cluster, the unsupervised clustering algorithm looks only at adjacent words in unannoted data, while the supervised clustering algorithm looks only at the tags the word receives in the annotated data. The unsupervised clustering tells us what known words a large number of unknown words are simliar to, but the supervised clustering carries much more information about what tags the known words are likely to receive.

2.2.1 Clustering Algorithm

Our supervised clustering algorithm is based on the method presented by Dhillon et al. (2003). This is similar to the well-known Lloyd algorithm for k-means clustering, but uses KL-divergence between

probability distributions, instead of Euclidian distance, to assign items to clusters. In our application of this algorithm, we simply keep moving each word to the cluster that has the most similar probability distribution over tags, and then re-estimating the tag probability distributions for the clusters, until the clustering converges. At a high-level, our algorithm is:

- For each unique word w in the training set, estimate a smoothed probability distribution $p(T|w)$ over tags given w.
- Select k seed words, and initialize k clusters for clustering 0, with one seed word per cluster.[1]
- Set $i = 0$.
- Repeat until the assignment of words to clusters in clustering i is the same as in clustering $i-1$, returning clustering i:
  - For each cluster c in clustering i, compute a probability distribution $p(T|c)$ over tags given c, such that
$$p(t|c) = \sum_{w \in c} p(w|c)p(t|w)$$
  - For each word w, find the cluster c that minimizes the KL-divergence $D_{KL}(p(T|w)||p(T|c))$, and assign w to cluster c in clustering $i+1$.
  - Set $i = i + 1$.

As indicated, the probability distributions $p(T|c)$ over tags for a given cluster are computed as the word-frequency-weighted mean of probability distributions $p(T|w)$ over tags given the words in the cluster. The $p(T|w)$ distributions are estimated based on the relative frequencies of each tag for a given word, smoothed using the interpolated Kneser-Ney method (Chen and Goodman, 1999) widely used in statistical language modeling. (See Section 3.2 for more discussion of this smoothing method applied to POS tag prediction.)

2.2.2 Cluster Initialization

Our clustering algorithm is identical to that of Dhillon et al., except for the method of initializing the clusters. Their initialization method would assign all words with the same most likely tag to the same initial cluster. Instead, we initialize the clusters using a set of seed words with the property that conflating any two of them would result in a large loss of information about tag probabilities.

We define the distance between a pair of words (w_1, w_2) as the total decrease resulting from treating w_1 and w_2 as indistinguishable, in the estimated log probability, based on $p(T|W)$, of the reference tagging of the training data. Letting n_1 be the number of occurrences in the training data of w_1, and similarly for n_2 and w_2, we compute the distance between w_1 and w_2 as

$$n_1 D_{KL}(p(T|w_1)||p_{w_1 w_2}) + n_2 D_{KL}(p(T|w_2)||p_{w_1 w_2})$$

where $p_{w_1 w_2} = p(T|w_1 \vee w_2)$, computed as

$$p_{w_1 w_2}(t) = \frac{n_1}{n_1 + n_2} p(t|w_1) + \frac{n_2}{n_1 + n_2} p(t|w_2)$$

We select a set S of k seed words as follows:

- Choose a maximal subset V of the training data vocabulary, such that every word in V has a different distribution of observed POS tags.
- Choose a random ordering of V.
- Initialize S to contain the first k words of V.

[1] Note that most words in the training set are not assigned to any initial cluster.

- Find the minimum distance d between any two words in S.
- Taking each remaining word w of V in order:
  - Find the minimum distance d' between w and any word in S.
  - If $d' > d$,
    - Select from S a pair of words (w', w'') separated by d.
    - Find the minimum distance d'_2 between w' and any word in S other than w''.
    - Find the minimum distance d''_2 between w'' and any word in S other than w'.
    - If $d'_2 < d''_2$, remove w' from S, otherwise remove w'' from S.
    - Add w to S.
    - Recompute the minimum distance d between any two words in S.

2.2.3 Random restarts

The clustering we find depends on the set of seed words, which in turn depends on the order in which the words in V are enumerated to select the seed words. To ensure that we find a good clustering, we try multiple runs of the algorithm based on different random enumerations of V, returning the clustering yielding the lowest entropy for predicting the training set tags from the clusters.

We noticed in preliminary experiments that a poor clustering on the first iteration of the algorithm seldom leads to a good final clustering, so we save the training set tag entropy for the first iteration of the best clustering found so far, and we abandon a run of the algorithm if it results in higher training set tag entropy on its first iteration than the best previously observed final clustering had on its first iteration. We continue trying different random enumerations until a fixed number of runs has passed since the current best clustering was found.

2.2.4 Classes for unknown words

Note that this clustering method assigns classes only to words observed in the training data. All words (ignoring distinctions among digits) not seen in the training data are assigned to an additional class. In training the tagging model, however, we treat each word that has a single occurrence in the training data as a member of this unknown-word class, so that features based on that class will be seen in training; but at tagging time, we give all words seen in the training data the class they are assigned by the clustering algorithm, and apply the unknown-word class only to words not seen in the training data.

2.3 Feature Weight Training

Our models are trained by optimizing the multiclass SVM hinge loss objective (Crammer and Singer, 2001) using stochastic subgradient descent as described by Zhang (2004). We use a small, constant learning rate of 2^{-8}, which early in our experiments we found generally to be a good value, given the size of our training set and the sorts of feature sets we were using. We did not re-optimize the learning rate as we experimented with different feature sets. We do not use a numerical regularizer (such as L_1 or L_2), but we avoid over-fitting by using early stopping, and averaging as Collins (2002a) does with the averaged perceptron. To determine the stopping point, we evaluate the model on the development set after each pass through the training data. We continue iterating until we have made 10 consecutive passes through the training data without reducing the development set error, and we return the model from the iteration with the lowest error.

2.4 Evaluation of Tagging Accuracy

We evaluate the tagging accuracy of three models: our new model with all the features discussed above, our new model minus the unsupervised distributional clustering features (to give a "no additional resources" measurement), and the Liang et al. model that was our starting point. Our data is the ususal Wall Street Journal corpus from Penn Treebank III (Marcus et al., 1993), split into standard training (sections 0–18), development (sections 19–21), and test (sections 22-24) sets.

Table 1 shows WSJ development and test set error rates for all tokens and for unknown-word (OOV) tokens for all three models. Our full model has an overall test set tag error rate of 2.66%, or 97.34%

Tagging Model	Dev Set All Tag Error %	Dev Set OOV Tag Error %	Test Set All Tag Error %	Test Set OOV Tag Error %
Our full feature set	2.69	9.40	2.66	8.93
Our features minus unsupervised classes	2.83	10.45	2.77	10.14
Liang et al. feature set	3.23	12.47	3.17	11.92

Table 1: WSJ development and test set error rates for different feature sets

accuracy. Omitting unsupervised word-class features results in a relative increase in the error rate of 4.1% overall and 13.5% on unknown words. The model trained on the Liang et al. feature set gives results consistent with their reported 3.2% test set error, but the error is 19.2% higher than the model using our full feature set, and 14.4% higher than our model without unsupervised word-class features.

3 Efficient Tag Inference

Although the complexity of tag inference with our model is only $O(|T|n)$, with a rich feature set and many possible tags, the simple summation of feature weights and comparison of sums implied by Equation 1 can still be slow. With our full model, a given token occurrence can have up to 53 features present, and on the WSJ development set, we measured the average number of features present with a non-zero weight for at least one tag to be 38.0. Given 45 possible tags in the Penn Treebank tag set and our full model, the average number of relevant non-zero feature weights per token on the WSJ development set is 1215.0. We reduce computational costs in two ways. First, we introduce a method of combining feature weights that effectively reduces the number of features per token by a factor of 8. Then we introduce a refined version of a tag dictionary that reduces the number of tags considered per token by a factor of more than 12 without noticeably affecting tagging accuracy. The combination of these techniques reduces the number of non-zero feature weights used per token by a factor of 75, which, in our Perl implementation, speeds up tagging by a factor of 45.

3.1 Combining Feature Weights

The base lexical feature types in our model form a natural hierarchy as follows:

1. Original case tokens
   1.1. Unsupervised distributional word clusters
   1.2. Lower-cased tokens
      1.2.1. Lower-cased 4-character prefixes
         1.2.1.1. Lower-cased 3-character prefixes
            1.2.1.1.1. Lower-cased 2-character prefixes
               1.2.1.1.1.1. Lower-cased 1-character prefixes
      1.2.2. Lower-cased 4-character suffixes
         1.2.2.1. Lower-cased 3-character suffixes
            1.2.2.1.1. Lower-cased 2-character suffixes
               1.2.2.1.1.1. Lower-cased 1-character suffixes
   1.3. Shape 1 features
      1.3.1. Shape 2 features
         1.3.1.1. Contains upper case token
         1.3.1.2. Contains digit
         1.3.1.3. Contains hyphen

The significance of the hierarchy is that the occurrence of a base feature of any of these types fully determines which features of the types below it in the hierarchy also occur. For example, given a whole token with its original casing, the corresponding features of all the other feature types in the hierarchy are completely determined. Given just the lower-cased version of the token, the lower-cased prefixes

and suffixes are determined, but the distributional word cluster and the shape features are not completely determined, because they depend on capitalization.[2]

We use this hierarchy to perform a simple transformation on the trained tagging model. For every base lexical feature f found in the training data, we add to the value of each feature weight associated with that base feature, the value of all corresponding feature weights for base features below f in the hierarchy. For instance, to the feature weight for the 3-character suffix *ion*, the tag *NN*, and the position -1 (i.e, the word preceding the word being tagged), we add the value of feature weight for the 2-character suffix *on*, the tag *NN*, and the position -1, plus the value of the feature weight for the 1-character suffix *n*, the tag *NN*, and the position -1.

To use this transformed model, we make a corresponding modification to feature extraction in the tagger. We carry out feature extraction top-down with respect to the base feature hierarchy, and whenever we find a base feature f for which there are any corresponding feature weights in the model, we skip the extraction of all the base features below f in the hierarchy. We can do that because the model has been transformed to incorporate the weights for all the skipped features into the corresponding feature weights associated with f. The weights for the skipped features are still kept in the model, so that they can be used when we encounter an unknown feature of the same type as f, such as an unknown whole word, or an unknown 4-character suffix, when we have seen the corresponding 3-character suffix.

The word-class-sequence features are arranged into a similar hierarchy, which is used in a similar way.

1. $\langle c(w_{i-2}), c(w_{i-1}), c(w_{i+1}), c(w_{i+2})\rangle$
   - 1.1. $\langle c(w_{i-2}), c(w_{i-1})\rangle$
   - 1.2. $\langle c(w_{i+1}), c(w_{i+2})\rangle$
   - 1.3. $\langle c(w_{i-1}), c(w_{i+1})\rangle$
     - 1.3.1. $c(w_{i-1})$
     - 1.3.2. $c(w_{i+1})$

Note that in this hierarchy, we have introduced a new feature type that does not actually exist in the trained model, the combination of the word-class bigrams preceding and following the word being tagged. The weights for the features of this type are constructed from the sums of the weights of other features lower in the hierarchy. To keep the size of the transformed model from exploding, we limit the instances of this feature type to those seen at least twice in the training data. We found this covered about 80% of the tagging decisions for the WSJ development set. We also included in the transformed model all possible instances (including those not observed in the training data) of the feature type 1.3 for word-class bigrams surrounding the word being tagged, which allows us to drop the feature weights for the lowest two feature types 1.3.1 and 1.3.2 after their feature weights have been added to the weights for the word-class-bigram features.

Altogether, these transformations increase the size of our full model from 151,174 features with 861,111 non-zero feature weights to 392,318 features with 17,047,515 non-zero feature weights. While this may be a substantial relative increase in size, the resulting model is still not particularly large in absolute terms.

Feature Weight Combination	Features per Token	Weights per Token	Tokens per Second	All Tag Error %	OOV Tag Error %
No	38.0	1215.0	1100	2.69	9.40
Yes	4.7	194.0	6400	2.69	9.40

Table 2: WSJ development set speeds and error rates without and with feature weight combination

In Table 2, we show the effect of these transformations on the speed of tagging the WSJ development set while considering all possible labels for each token. As expected, feature weight combination has no effect on tagging error, since it results in the same tagging decisions as the original model and feature extraction method. The "Features per Token" column shows the average number of features used for

[2]Note that we could have placed the "contains digit" or "contains hyphen" features under "lower-cased tokens" instead of "Shape 2"; our choice here was arbitrary.

each tagging decision, without and with the model and feature extraction feature-weight-combination transformations. The transformations reduce this number by a factor of 8.13. The "Weights per Token" column is the corresponding number of non-zero feature weights used for each tagging decision. Feature weight combination reduces this number by a factor of 6.26.

The "Tokens per Second" measurements are rounded to two significant digits due to the limited precision of our observations. Time was measured to the nearest second, and for each tagger, the data set was replicated enough times for the total tagging time to fall in the range of 100 to 200 seconds. The times reported include only reading the sentence-split tokenized text, extracting features, and predicting tags; time to read in model parameters and initialize the corresponding data structures is not inlcuded. Times are for a single-threaded implementation in Perl on a Linux workstation equipped with Intel Xeon X5550 2.67 GHz processors. In this implementation, feature weight combination increases tagging throughput by a factor of 5.82.

3.2 Pruning Possible Tags

It has long been standard practice to prune the set of possible tags considered for each word, in order to speed up tagging. Ratnaparkhi (1996) may have been the first to use the common heuristic of defining a tag dictionary allowing any tag for unknown words, but restricting each known word to the tags it was observed with in the training data. In addition, the tag dictionary for known words is sometimes further pruned (e.g., Banko and Moore, 2004; Giménez and Màrquez, 2004, 2012) according to the relative frequency of tags for each word. Tags observed in the training data with less than some fixed proportion of the occurrences of a particular word are not considered as possible tags for that word in test data.

In our experiments, we find these heuristics produce fast tagging, but lead to a noticable loss of accuracy, because known words are never allowed to be labeled with tags they were not observed with in the training data. This is similar to the problem of unseen n-grams in statistical language modeling, so we apply methods developed in that field to the problem of dictionary pruning for POS tagging. We construct our tag dictionary based on a "bigram" model of the probability $p(t|w)$ of a tag t given a word w, estimated from the annotated training data. The probabilities for tags that have never been seen with a given word, as well as all the tag probabilities for unknown words, are estimated by interpolation with a "unigram" distribution over the tags.

To estimate the probabilities of tags given words, we use the same interpolated-Kneser-Ney-smoothed (Chen and Goodman, 1999) model that we used in Section 2.2.1 in our supervised word-clustering procedure. In this model, we estimate the probabilty $p(t|w)$ by interpolating a discounted relative frequency estimate with a lower-order estimate of $p(t)$. The lower-order estimates are based on "diversity counts", taking the count of a tag t to be the number of distinct words ever observed with that tag. This has the desirable property for POS tagging that closed-class tags receive a very low estimated probabilty of being assigned to a rare or unknown word, even though they occur very frequently with a small number of frequent words. We use a single value for the discount parameter in the Kneser-Ney formula, chosen to maximize the estimated probability of the reference tagging of the development set. These probabilities are estimated ignoring distinctions among digit characters, just as in the features of our tagging model.

We construct our tag dictionary by setting a threshold on the value of $p(t|w)$. Whenever $p(t|w)$ is less than or equal to the threshold, the tag t is considered not to be a possible POS tag for the word w. Our preferred threshold ($p(t|w) > 0.0005$) is set to prune as agressively as possible while maintaining tagging accuracy on the WSJ development set. This threshold is applied to both known and unknown words, which produces 24 possible tags for unknown words by applying the threshold to the lower-order probability estimate $p(t)$. Note that the probabilities we use for pruning can be viewed as posteriors of a very simple POS tagging model, which makes inferring a tag dictionary an instance of coarse-to-fine inference with posterior pruning (Charniak et al., 2006; Weiss and Taskar, 2010).

The standard tag dictionary pruning heuristics can be viewed as a application of the same approach, but with the $p(t|w)$ probabilities being unsmoothed relative-frequency estimates for known words and a uniform distribution for unknown words. The original Ratnaparkhi heuristic amounts to thresholding these probabilities at 0, with a higher threshold being applied when using additional pruning.

Pruning Method	Tags per Token	Weights per Token	Tokens per Second	All Tag Error %	OOV Tag Error %	Seen Tag Error %	Unseen Tag Error %	OOV Mean Tags	Seen Mean Tags	Unseen Mean Tags
None	45.0	194.0	6400	2.69	9.40	2.19	52.0	45.0	45.0	45.0
Ratnaparkhi	3.7	19.0	47000	2.81	9.40	2.07	100.0	45.0	2.3	1.3
Ratnaparkhi+	2.9	14.3	56000	2.81	9.40	2.07	100.0	45.0	1.4	1.2
Kneser-Ney	3.5	16.1	49000	2.69	9.45	2.18	55.5	21.3	2.8	10.2
Kneser-Ney+	1.8	6.1	67000	2.81	9.74	2.14	83.8	10.6	1.4	2.8

Table 3: WSJ development set speeds and error rates for different tag dictionary pruning methods

In Table 3 we compare these methods of tag dictionary pruning on the WSJ development set, when combined with our feature-weight-combination technique. The "Tags per Token" column shows the average number of tags considered for each tagging decision, depending on the tag pruning method used. "Weights per Token", "Tokens per Second", "All Tag Error %", and "OOV Tag Error %" are as in Table 2. The first line repeats the experiment with no tag dictionary pruning from Table 2. The next line gives results for Ratnaparkhi's dictionary pruning method, and the next line, "Ratnaparkhi+", gives results for the maximum additional pruning by thresholding based on unsmoothed relative frequencies that does not increase overall tagging error ($p(t|w) > 0.005$). We see that these taggers are much faster than the unpruned tagger, but noticeably less accurate.

The final two lines of Table 3 are for our tag dictionary pruning method, with different pruning thresholds. The "Kneser-Ney" line represents our preferred threshold, set to prune as agressively as possible without noticeably degrading the overall tagging error on the WSJ development set. This produces a lower error rate than either Ratnaparkhi or Ratnaparkhi+ pruning, but Ratnaparkhi+ pruning results in faster tagging. However, if we increase the pruning threshold until we match the Ratnaparkhi+ error rate, as shown in the final "Kneser-Ney+" line, our method is faster than Ratnaparkhi+.

The remaining columns of Table 3 provide some insight as to why Kneser-Ney-smoothed pruning with our preferred threshold results in lower error than Ratnaparkhi and Ratnaparkhi+ pruning. The column labeled "Seen Tag Error %" is the error rate for examples with word/tag pairs seen in training. The column labeled "Unseen Tag Error %" is the error rate for examples with word/tag pairs not seen in training, but with a word that was seen in training. There are 660 of the latter examples in the WSJ development set, which amounts to 0.5% of that data set. By construction, the error rate of the Ratnaparkhi and Ratnaparkhi+ pruning methods on this subset of the data is 100%, but both the unpruned tagger and the tagger with Kneser-Ney-smoothed pruning correctly tag nearly half of these examples.

The Ratnaparkhi and Ratnaparkhi+ pruning methods are somewhat more accurate than the Kneser-Ney-smoothed pruning method on the seen word/tag pairs and the unknown words, but not enough to overcome the losses on the unseen word/tag pairs with known words. In absolute numbers on the WSJ development set, both the Ratnaparkhi and Ratnaparkhi+ pruning methods make 131 fewer errors on the seen word/tag pairs and 2 fewer errors on the unknown words, but 294 more errors on the unseen word/tag pairs with known words, compared to Kneser-Ney-smoothed pruning method with our preferred threshold. The final three columns of Table 3 show the mean number of tags allowed by each dictionary for these three categories of examples. Compared to Ratnaparkhi and Ratnaparkhi+ pruning, our preferred threshold for Kneser-Ney-smoothed pruning slightly increases the number of tags considered for seen word/tag pairs, substantially reduces the number of tags considered for unknown words, and substantially increases the number of tags considered for unseen word/tag pairs with known words.

4 Comparison to Other Taggers

We compared our tagger to several publicly available taggers, on the standard WSJ POS tagging test set. As far as we know, six taggers have been reported to have an error rate of less than 2.7% (accuracy greater than 97.3%) on this test set. Three of these are publicly available: the Stanford tagger (Toutanova et al., 2003; Manning, 2011), the Prague COMPOST tagger (Spoustová, et al., 2009), and the UPenn

bidirectional tagger (Shen et al., 2007).[3] We tested two versions of the Stanford tagger, one based on their most accurate model "wsj-0-18-bidirectional-distsim", and one based on the much faster, but less accurate model "english-left3words-distsim" recommended for practical use on the Stanford tagger website. The UPenn tagger is run with a beam width of 3, which is the setting that gave their best reported results.

These taggers were all tested on on the same Linux workstation as our Perl tagger. To obtain comparable speed measurements omitting time for initialization, we performed two runs with each tagger. one on the first 1000 sentences of the test set, and another with those 1000 sentences followed by the entire test set replicated enough times to produce a difference in total time of at least 100 seconds. The tagging speed was inferred from the difference in these two times. The Stanford tagger reports tagging times directly, and these agreed with our measurements to two significant digits, which is the precision limit of our measurements.

We also report on the SVMTool tagger of Giménez and Màrquez (2004). Giménez recently provided us with benchmarks, which he obtained with a somewhat faster processor than ours, the Intel Xeon X5660 2.80 GHz. We give results for two versions of this tagger, one in Perl and one in C++, both with a combination of left-to-right and right-to-left tagging, which gives higher accuracy with this tagger than either direction by itself.

Tagger	Implementation Language	WSJ Tokens per Second	WSJ All Tag Error %	WSJ OOV Tag Error %	Brown Tokens per Second	Brown All Tag Error %	Brown OOV Tag Error %
This work	Perl	51000	2.66	9.02	40000	3.46	10.64
Stanford fast	Java	80000	3.13	10.31	50000	4.47	12.62
Stanford accurate	Java	5900	2.67	7.90	1600	3.86	11.21
COMPOST	C	2600	2.57	10.03	2700	3.36	12.16
UPenn	Java	270	2.67	10.39	290	3.90	12.96
SVMTool	Perl	1340	2.86	11.37			
SVMTool	C++	7700	2.86	11.37			

Table 4: WSJ test set and Brown corpus speeds and error rates compared to publicly available taggers

Results on the WSJ test set are shown in Table 4. We include a column giving the implementation language of each tagger to help interpret the results. Generally, we would expect an algorithm implemented in Perl to be slower than the same algorithm implemented in Java, which in turn would probably be slower than the same algorithm implemented in C/C++; although depending on the libraries used and the degree of optimization in the compilers, Java can sometimes be competitive with C/C++ (See, for example, `http://blog.famzah.net/2010/07/01/cpp-vs-python-vs-perl-vs-php-performance-benchmark/`).

"This work" refers to our tagger with feature weight combination and Kneser-Ney-smoothed dictionary pruning, with the pruning threshold set to maximize pruning without decreasing overall tagging accuracy on the WSJ development set. The fast Stanford tagger is the fastest overall by a wide margin, but it is also the least accurate. Our tagger is both the second fastest and the second most accurate, having an error rate relatively 3.9% higher (absolutely 0.09% higher) than the COMPOST tagger. But our tagger is almost 20 times faster than COMPOST, and more than 8 times faster than the accurate Stanford tagger, the second fastest tagger of equivalent or better accuracy. This is despite the fact that our tagger is written in Perl, while the other high-accuracy taggers are written either in Java or C.

As a final, out-of-domain evaluation, we ran the five taggers that we had direct access to on the Brown Corpus subset (3279 sentences, 83769 tokens) from the Penn Treebank. As might be expected, tagging was in general both slower and less accurate than on in-domain data. Our tagger maintained its relative position with respect to both speed and accuracy compared to all the other taggers. The only qualitative change in position of any tagger is that on the Brown Corpus data, the accurate Stanford tagger is slower than COMPOST, which actually runs faster than it does on the WSJ test set.

[3] A fourth tagger, the semi-supervised condensed nearest neighbor tagger of Søgaard (2011), has some released source code, but not a complete tagger nor detailed instructions on how to build the tagger Søgaard evaluates.

5 Conclusions

We have shown that a feature-rich model for POS tagging by independent classifiers can reach tagging accuracies comparable to several state-of-the art taggers, and we have introduced implementation strategies that result in much faster tagging than any other high-accuracy tagger we are aware of, despite these other taggers being implemented in faster programming languages.

A number of the techniques introduced here may have applications to other tasks. The sort of word-class-sequence models derived by supervised clustering described in Section 2.2 may be useful for other sequence labeling tasks, such as named-entity recognition. Our method of pruning the tag dictionary with smoothed probability distributions could also be used for label pruning for other problems with large label sets. Finally, the feature-weight-combination technique of Section 3.1 can be applied to any rich feature space in which the features have the kind of hierarchical structure we see in POS tagging. Such feature spaces are common in NLP, since we are almost always dealing with lexical items and their sublexical features.

Acknowledgements

Thanks to Chris Manning, John Bauer, and Jenny Liu for help with the Stanford tagger; to Johanka Spoustová and Jan Hajič, for help with the COMPOST tagger; to Jesús Giménez, for benchmarking the SVMTool tagger; and to Kuzman Ganchev and Dan Bikel, for valuable comments on earlier versions of this paper.

References

Michele Banko and Robert C. Moore. 2004. Part of speech tagging in context. In *Proceedings of the 20th International Conference on Computational Linguistics*, August 23–27, Geneva, Switzerland, 556–561.

Eugene Charniak, Mark Johnson, Micha Elsner, Joseph Austerweil, David Ellis, Isaac Haxton, Catherine Hill, R. Shrivaths, Jeremy Moore, Michael Pozar, and Theresa Vu. 2006. Multilevel coarse-to-ne PCFG parsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, June 4–9, New York, New York, USA, 168–175.

Stanley F. Chen and Joshua T. Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–393.

Michael Collins. 2002a. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, July 6–7, Philadelphia, Pennsylvania, USA, 1–8.

Michael Collins. 2002b. Ranking algorithms for named-entity extraction: boosting and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics*, July 7–12, Philadelphia, Pennsylvania, USA, 489–486.

Koby Crammer and Yoram Singer. 2001. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292.

Radu Florian, Abe Ittycheriah, Honyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of CoNLL-2003*, May 31–June 1, Edmonton, Alberta, Canada.

Inderjit S. Dhillon, Subramanyam Mallela, and Rahul Kumar. 2003. A divisive information-theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3:1265–1287.

Jesús Giménez and Lluís Màrquez. 2004. SVMTool: a general POS tagger generator based on support vector machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, May 26–28, Lisbon, Portugal, 43–46.

Jesús Giménez and Lluís Màrquez. 2012. SVMTool Technical Manual v1.4. http://www.lsi.upc.edu/~nlp/SVMTool/SVMTool.v1.4.pdf

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology*, June 3–8, Montreal, Quebec, Canada, 142–151.

Percy Liang, Hal Daumé III, and Dan Klein. 2008. Structure compilation: trading structure for features. In *Proceedings of the 25th International Conference on Machine Learning*, July 5–9, Helsinki, Finland, 592–599.

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In Alexander Gelbukh (ed.), *Computational Linguistics and Intelligent Text Processing, 12th International Conference, CICLing 2011, Proceedings, Part I. Lecture Notes in Computer Science 6608*, Springer, 171–189.

Mitchell P. Marcus, Beatrice Santorini, and Mary A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313-330.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association of Computational Linguistics*, June 25–30, Ann Arbor, Michigan, USA, 91–98.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, May 17–18, Philadelphia, Pennsylvania, USA, 133–142.

Libin Shen, Giorgio Satta, and Aravind K. Joshi. 2007. Guided learning for bidirectional sequence classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, June 23–30, Prague, Czech Republic, 760–767.

Drahomíra "johanka" Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta. 2009. Semi-supervised training for the averaged perceptron POS tagger. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, March 30–April 3, Athens, Greece, 763–771.

Anders Søgaard. 2011. Semisupervised condensed nearest neighbor for part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics*, June 19–24, Portland, Oregon, USA, 48–52.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, May 27–June 1, Edmonton, Alberta, Canada, 173–180.

David Weiss and Ben Taskar. 2010. Structured prediction cascades. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, May 13–15, Chia Laguna Resort, Sardinia, Italy, 916-923.

Tong Zhang. 2004. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the 21st International Conference on Machine Learning*, July 4–8, Banff, Alberta, Canada, 919–926.

Morfessor FlatCat: An HMM-Based Method for Unsupervised and Semi-Supervised Learning of Morphology

Stig-Arne Grönroos[1]
stig-arne.gronroos@aalto.fi

Sami Virpioja[2]
sami.virpioja@aalto.fi

Peter Smit[1]
peter.smit@aalto.fi

Mikko Kurimo[1]
mikko.kurimo@aalto.fi

[1]Department of Signal Processing and Acoustics, Aalto University
[2]Department of Information and Computer Science, Aalto University

Abstract

Morfessor is a family of methods for learning morphological segmentations of words based on unannotated data. We introduce a new variant of Morfessor, FlatCat, that applies a hidden Markov model structure. It builds on previous work on Morfessor, sharing model components with the popular Morfessor Baseline and Categories-MAP variants. Our experiments show that while unsupervised FlatCat does not reach the accuracy of Categories-MAP, with semi-supervised learning it provides state-of-the-art results in the Morpho Challenge 2010 tasks for English, Finnish, and Turkish.

1 Introduction

Morphological analysis is essential for automatic processing of compounding and highly-inflecting languages, for which the number of unique word forms may be very large. Apart from rule-based analyzers, the task has been approached by machine learning methodology. Especially unsupervised methods that require no linguistic resources have been studied widely (Hammarström and Borin, 2011). Typically these methods focus on morphological segmentation, i.e., finding *morphs*, the surface forms of the morphemes.

For language processing applications, unsupervised learning of morphology can provide decent-quality analyses without resources produced by human experts. However, while morphological analyzers and large annotated corpora may be expensive to obtain, a small amount of linguistic expertise is more easily available. A well-informed native speaker of a language can often identify the different prefixes, stems, and suffixes of words. Then the question is how many annotated words makes a difference. One answer was provided by Kohonen et al. (2010), who showed that already one hundred manually segmented words provide significant improvements to the quality of the output when comparing to a linguistic gold standard.

The semi-supervised approach by Kohonen et al. (2010) was based on Morfessor Baseline, the simplest of the Morfessor methods by Creutz and Lagus (2002; 2007). The statistical model of Morfessor Baseline is simply a categorical distribution of morphs—a unigram model in the terms of statistical language modeling. As the semi-supervised Morfessor Baseline outperformed all unsupervised and semi-supervised methods evaluated in the Morpho Challenge competitions (Kurimo et al., 2010a) so far, the next question is how the approach works for more complex models.

Another popular variant of Morfessor, Categories-MAP (CatMAP) (Creutz and Lagus, 2005), models word formation using a hidden Markov model (HMM). The context-sensitivity of the model improves the precision of the segmentation. For example, it can prevent splitting a single *s*, a common English suffix, from the beginning of a word. Moreover, it can disambiguate between identical morphs that are actually surface forms of different morphemes. Finally, separation of stems and affixes in the output makes it simple to use the method as a stemmer.

In contrast to Morfessor Baseline, the lexicon of CatMAP is *hierarchical*: a morph that is already in the lexicon may be used to encode the forms of other morphs. This has both advantages and drawbacks.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1177–1185, Dublin, Ireland, August 23-29 2014.

One downside is that it mixes the prior and likelihood components of the cost function, so that the semi-supervised approach presented by Kohonen et al. (2010) is not usable.

1.1 Hierarchical versus flat lexicons

From the viewpoint of data compression and following the two-part Minimum Description Length principle (Rissanen, 1978), Morfessor tries to minimize the number of bits needed to encode both the model parameters and the training data. Equivalently, the cost function L can be derived from the Maximum a Posteriori (MAP) estimate:

$$\hat{\theta} = \arg\max_{\theta} \mathrm{P}(\theta \mid \boldsymbol{D}) = \arg\min_{\theta} \big(-\log \mathrm{P}(\theta) - \log \mathrm{P}(\boldsymbol{D} \mid \theta) \big) = \arg\min_{\theta} L(\theta, \boldsymbol{D}), \qquad (1)$$

where θ are the model parameters, $\boldsymbol{D}$ is the training corpus, $\mathrm{P}(\theta)$ is the prior of the parameters and $\mathrm{P}(\boldsymbol{D} \mid \theta)$ is the data likelihood.

In context-independent models such as Morfessor Baseline, the parameters include only the forms and probabilities of the morphs in the lexicon of the model. Morfessor Baseline and Categories-ML (CatML) (Creutz and Lagus, 2004) use a flat lexicon, in which the forms of the morphs are encoded directly as strings: each letter requires a certain number of bits to encode. Thus longer morphs are more expensive. Encoding a long morph is worthwhile only if the morph is referred to frequently enough from the words in the training data. If a certain string, let us say *segmentation*, is common enough in the training data, it is cost-effective to have it as a whole in the lexicon. Splitting it into two items, *segment* and *ation*, would double the number of pointers from the data, even if those morphs were already in the lexicon. The undersegmentation of frequent words becomes evident especially if the training data is a corpus instead of a list of unique word forms.

In contrast, Morfessor CatMAP applies a hierarchical lexicon, which makes use of the morphs that are already in the lexicon. Instead of encoding the form of *segmentation* by its 12 letters, we could just encode the form with two references to the forms of the morphs *segment* and *ation*. This may also cause errors, for example encoding *station* with *st* and *ation*.

The lexicon of Morfessor CatMAP allows but does not force hierarchical encoding for the forms: each morph has an extra parameter that indicates whether it has a hierarchical representation or not. The problem of oversegmentation, as in *st* + *ation*, is solved using the morph categories. The categories, which are states of the HMM, include stem, prefix, suffix, and a special non-morpheme category. The non-morpheme category is intended to catch segments that do not fit well into the three proper morph categories because they are fragments of a larger morph. In our example, the morph *st* cannot be a suffix as it starts the word, it is unlikely to be a prefix as it directly precedes a common suffix *ation*, and it is unlikely to be a stem as it is very short. Thus the algorithm is likely to use the non-morpheme state. The hierarchy is expanded only up to the level in which there are no non-morphemes, so the final analysis is still *station*. Without the hierarchy, the non-morphemes have to be removed heuristically, as in CatML (Creutz and Lagus, 2004).

A hierarchical lexicon presents some challenges to model training. For a standard unigram or HMM model, if you know the state and emission sequence of the training data, you can directly derive the maximum likelihood (ML) parameters of the model: a probability of a morph is proportional to the number of times it is referred to, conditional on the state in the HMM. But if the lexicon is partly hierarchical, also the references *within* the lexicon add to the reference counts, and there is no direct way to find the ML parameters even if the encoding of the training data is known. Similarly, semi-supervised learning cannot be accomplished simply by adding the counts from an annotated data set, as it is not clear when to use hierarchy instead of segmenting a word directly in the data.

Moreover, for a flat lexicon, the cost function divides into two parts that have opposing optima: the cost of the data (likelihood) is optimal when there is minimal splitting and the lexicon consists of the words in the training data, whereas the cost of the model (prior) is optimal when the lexicon is minimal and consists only of the letters. In consequence, the balance of precision and recall of the segmentation boundaries can be directly controlled by setting a weight for the data likelihood. Tuning this hyperparameter is a very simple form of supervision, but it has drastic effects on the segmentation results

(Kohonen et al., 2010). A direct control of the balance may also be useful for some applications: Virpioja et al. (2011) found that the performance of the segmentation algorithms in machine translation correlates more with the precision than the recall. The weighting approach does not work for hierarchical lexicons, for which changing the weight does not directly affect the decision whether to encode the morph with hierarchy or not.

1.2 Morfessor FlatCat

In this paper, we introduce a new member to the Morfessor family, Morfessor FlatCat. As indicated by its name, FlatCat uses a flat lexicon. Our hypothesis is that enabling semi-supervised learning is effective in compensating for the undersegmentation caused by the lack of hierarchy. In particular, semi-supervised learning can improve modeling of suffixation. In the examined languages, suffixes tend to serve syntactic purposes, such as marking case, tense, person or number. Examples are the suffix *s* marking tense and person in *she writes* and number in *stations*. Thus the suffix class is closed and has only a small number of morphemes compared to the prefix and stem categories. As a consequence, a large coverage of suffixes can be achieved already with a relatively small annotated data set.

The basic model of morphotactics in FlatCat is the same as in the CatML and CatMAP variants: a hidden Markov model with states that correspond to a word boundary and four morph categories: stem, prefix, suffix, and non-morpheme. As in CatML, we apply heuristics for removal of non-morphemes from the final segmentation. However, because FlatCat uses MAP estimation of the parameters, these heuristics are not necessary during the training for controlling the model complexity, but merely used as a post-processing step to get meaningful categories.

Modeling of morphotactics improves the segmentation of compound words, by allowing the overall level of segmentation to be increased without increasing the number of correct morphs used in incorrect positions. As the benefits of semi-supervised learning and improved morphotactics are likely to complement each other, we can expect improved performance over the semi-supervised Morfessor Baseline method. By experimental comparison to the previous Morfessor variants, we are able to shed more light on the effects of using an HMM versus unigram model for morphotactics, using a hierarchical versus flat lexicon, and exploiting small amounts of annotated training data.

2 FlatCat model and algorithms

Morfessor FlatCat uses components from the older Morfessor variants. Instead of going through all the details, we refer to the previous work and highlight only the differences. Common components between Morfessor methods are summarized in Table 1.

As a generative model, Morfessor FlatCat describes the joint distribution $\mathrm{P}(\boldsymbol{A}, \boldsymbol{W} \mid \theta)$ of words and their analyses. The words $\boldsymbol{W}$ are observed, but their analyses, $\boldsymbol{A}$, is a latent variable in the model. An analysis of a word contains its morphs and morph categories: prefix, stem, suffix, and non-morpheme.

As marginalizing over all possible analyses is generally infeasible, point estimates are used during the training. The likelihood conditioned on the current analyses is

$$\mathrm{P}(\boldsymbol{D} \mid \boldsymbol{A}, \theta) = \prod_{j=1}^{|\boldsymbol{D}|} \mathrm{P}(\boldsymbol{A}_j \mid \theta). \tag{2}$$

If m_i are the morphs in $\boldsymbol{A}_j$, c_i are the hidden states of the HMM corresponding to the categories of the morphs, and $\#$ is the word boundary, $\mathrm{P}(\boldsymbol{A}_j \mid \theta)$ is

$$\mathrm{P}(c_1 \mid \#) \prod_{i=1}^{|\boldsymbol{A}_j|} \left[\mathrm{P}(m_i \mid c_i)\, \mathrm{P}(c_{i+1} \mid c_i) \right] \mathrm{P}(\# \mid c_{|\boldsymbol{A}_j|}). \tag{3}$$

Morfessor FlatCat applies an MDL-derived prior designed to control the number of non-zero parameters. The prior is otherwise the same as in Morfessor Baseline, but it includes the usage properties from Morfessor CatMAP: the length of the morph and its right and left perplexity. The perplexity measures describe the predictability of the contexts in which the morph occurs. The emission probability of

	Morfessor method			
Component	Baseline	CatMAP	CatML	**FlatCat**
Lexicon type	Flat	Hierarchy	Flat	Flat
Morphotactics	Unigram	HMM	HMM	HMM
Estimation	MAP	MAP	ML	MAP
Semi-supervised	Implemented	Not implemented	Not implemented	Implemented

Table 1: Overview of similarities and differences between Morfessor methods.

a morph conditioned on the morph category, $\mathrm{P}(m \mid c)$, is calculated from the properties of the morphs similarly as in CatMAP.

2.1 Training algorithms

The parameters are optimized using a local search. Only a part of the parameters are optimized in each step: the parameters that are used in calculating the likelihood of a certain part, *unit*, of the corpus. Units vary in complexity, from all occurrences of a certain morph to the occurrences of a morph bigram whose context fits to certain criteria.

The algorithm tries to simultaneously find the optimal segmentation for the unit and the optimal parameters consistent with that segmentation:

$$(\boldsymbol{A}, \theta) = \underset{\textsc{Op}(\boldsymbol{A},\theta)}{\arg\min} \big\{ L(\theta, \boldsymbol{A}, \boldsymbol{D}) \big\}. \tag{4}$$

The training operators OP define the units changed by the local search and the alternative segmentations tried for each unit. There are three training operators: *split, join* and *resegment*, analogous to the similarly named stages in CatMAP.

The split operator is applied first. It targets all occurrences of a specific morph in the corpus simultaneously, attempting to split it into two parts. The whole corpus is processed by sorting the current morphs by length from shortest to longest.

The second operator attempts to join morph bigrams, grouped by the position of the bigram in the word. The position grouped bigram counts are sorted by frequency, from most to least common.

Finally, resegmenting uses the generalized Viterbi algorithm to find the currently optimal segmentation for one whole word at a time. This operator targets each corpus word in increasing order of frequency.

The heuristics used in FlatCat to remove non-morphemes from the final segmentation are the following: All consequent non-morphemes are joined together. If the resulting morph is longer than 4 characters, it is accepted as a stem. All non-morphemes preceded by a suffix and followed by only suffixes or other non-morphemes are recategorized as suffixes without joining with their neighbors. If any short non-morphemes remain, they are joined either to the preceding or following morphs (the latter only for those in the initial position).

2.2 Semi-supervised learning

Kohonen et al. (2010) found that semi-supervised learning of Morfessor models was not effective by only fixing the values of the analysis $\boldsymbol{A}$ for the annotated samples $\boldsymbol{D}_A$. Their solution was to introduce corpus likelihood weights α and β, one for the unannotated data set and one for the annotated data set. Thus, instead of optimizing the MAP estimate, Kohonen et al. (2010) minimize the cost

$$L(\theta, \boldsymbol{A}, \boldsymbol{D}, \boldsymbol{D}_A) = -\log \mathrm{P}(\theta) - \alpha \log \mathrm{P}(\boldsymbol{D} \mid \boldsymbol{A}, \theta) - \beta \log \mathrm{P}(\boldsymbol{D}_A \mid \boldsymbol{A}, \theta). \tag{5}$$

The weights can be tuned on a development set. We use the same scheme for FlatCat.

The likelihood of the annotated data is calculated using the same HMM that is used for the unannotated data. The morph properties are estimated only from the unannotated data. To ensure that the morphs required for the annotated data can be emitted, a copy of each word in the annotations is added to the

(a) English.

Method	α	β	Pre	Rec	F
U Baseline	1.0	–	.88	.59	.71
U CatMAP	–	–	.89	.51	.65
U FlatCat	1.0	–	.90	.57	.69
W Baseline	0.7	–	.83	.62	.71
W FlatCat	0.5	–	.84	.60	.70
SS Baseline	1.0	3000	.83	**.77**	.80
SS FlatCat	0.9	2000	.86	.76	.81
SS CRF+FlatCat	0.9	2000	.87	**.77**	**.82**
S CRF	–	–	**.92**	.73	.81

(b) Finnish.

Method	α	β	Pre	Rec	F
U Baseline	1.0	–	.84	.38	.53
U CatMAP	–	–	.76	.51	.61
U FlatCat	1.0	–	.84	.38	.52
W Baseline	.02	–	.62	.54	.58
W FlatCat	.015	–	.66	.52	.58
SS Baseline	.1	15000	.75	.72	.73
SS FlatCat	.2	1500	.79	.71	.75
SS CRF+FlatCat	.2	2500	.82	**.76**	.79
S CRF	–	–	**.88**	.74	**.80**

Table 2: Boundary Precision and Recall results in comparison to gold standard segmentation. Abbreviations have been used for Unsupervised (U), likelihood weighted (W), semi-supervised (SS) and fully supervised (S) methods. Best results for each measure have been hilighted using boldface.

unannotated data. This unannotated copy is loosely linked to the annotated word: operations that would result in the removal of a morph required for the annotations from the lexicon cannot be selected, as such an operation would have infinite cost.

3 Experiments

We compare Morfessor FlatCat[1] to two previous Morfessor methods and a fully supervised discriminative segmentation method. The Morfessor methods used as references are the CatMAP[2] and Baseline[3] implementations by Creutz and Lagus (2005) and Virpioja et al. (2013), respectively. Virpioja et al. (2013) implements the semi-supervised method described by Kohonen et al. (2010). For a supervised discriminative model, we use a character-level conditional random field (CRF) implementation by Ruokolainen et al. (2013)[4].

We use the English, Finnish and Turkish data sets from Morpho Challenge 2010 (Kurimo et al., 2010b). They include large unannotated word lists, one thousand annotated words for training, 700–800 annotated words for parameter tuning, and 10×1000 annotated words for testing.

For evalution, we use the BPR score by Virpioja et al. (2011). The score calculates the precision (Pre), recall (Rec), and F_1-score (F) of the predicted morph boundaries compared to a linguistic gold standard. In the presence of alternative gold standard analyses, we weight each alternative equally.

We also report the mean average precision from the English and Finnish information retrieval (IR) tasks of the Morpho Challenge. The Lemur Toolkit (Ogilvie and Callan, 2001) with Okapi BM25 ranking was used. The Finnish data consists of 55K documents, 50 test queries and 23K binary relevance assessments. The English data consists of 170K documents, 50 test queries and 20K binary relevance assessments. The domain of both data sets is short newspaper articles. All word forms in both the corpora and the queries were replaced by the morphological segmentation to be evaluated.

Morfessor FlatCat is a pipeline method that refines an initial segmentation given as input. We try two different initializations for the semi-supervised setting: initializing with the segmentation produced by semi-supervised Morfessor Baseline, and initializing with the CRF segmentation. All unsupervised and likelihood-weighted results are initialized with the corresponding Baseline output.

All methods were trained using word types. The weight and perplexity threshold parameters were optimized separately for each method, using a grid search with the held-out data set. The supervised CRF method was trained using the one thousand word annotated training data set.

[1]Available at `https://github.com/aalto-speech/flatcat`
[2]Available at `http://www.cis.hut.fi/projects/morpho/morfessorcatmap.shtml`
[3]Available at `https://github.com/aalto-speech/morfessor`
[4]Available at `http://users.ics.aalto.fi/tpruokol/`

Method	α	β	Pre	Rec	F
U Baseline	1.0	–	.85	.36	.51
U CatMAP	–	–	.83	.50	.62
U FlatCat	1.0	–	.87	.36	.51
W Baseline	0.1	–	.71	.41	.52
W FlatCat	0.3	–	.88	.38	.53
SS Baseline	0.4	2000	.86	.60	.71
SS FlatCat	0.8	2666	.87	.59	.70
SS CRF+FlatCat	1.0	3000	.87	**.61**	**.72**
S CRF	–	–	**.89**	.58	.70

Table 3: Boundary Precision and Recall results in comparison to gold standard segmentation for Turkish. Abbreviations have been used for Unsupervised (U), likelihood weighted (W), semi-supervised (SS) and fully supervised (S) methods. Best results for each measure have been hilighted using boldface.

3.1 Comparison to linguistic gold standards

The results of the BPR evaluations are shown in Tables 2 (English, Finnish) and 3 (Turkish). Semi-supervised FlatCat initialized using CRF achieves the highest F-score for both the English and Turkish data sets. The difference between the highest and second-highest scoring methods is statistically significant for Finnish and Turkish, but not for English (Wilcoxon signed-rank test, $p < 0.01$).

Table 4 shows BPR for subsets of words consisting of different morph category patterns. Each subset consists of 500 words from the English or Finnish gold standard, with one of five selected morph patterns as the only valid analysis. The subsets consist of words with the following morph patterns: words that should not be segmented (STM), compound words consisting of exactly two stems (STM + STM), a prefix followed by a stem (PRE + STM), a stem followed by a single suffix (STM + SUF) and a stem and exactly two suffixes (STM + SUF + SUF). For the STM pattern only precision is reported, as recall is not defined for an empty set of true boundaries.

The fact that semi-supervised FlatCat compares well against CatMAP in recall, for all morph patterns and for the test set as a whole, indicates that supervision indeed is effective in compensating for the undersegmentation caused by the lack of hierarchy in the lexicon. The benefit of modeling morphotactics can be seen in improved precision for the STM + STM (for English and Finnish) and PRE + STM (for Finnish) patterns when comparing against semi-supervised Baseline. The more aggressive segmentation of Baseline gives better results for the English PRE + STM subset than for Finnish due to the shortness of the English prefixes (on average 3.6 letters for the English and 5.3 for the Finnish subset). While not directly observable in Table 4, a large part of the improvement over semi-supervised Baseline is explained by that FlatCat does not use suffix-like morphs in incorrect positions.

Initializing the FlatCat model with CRF segmentation improves the F-scores in all subsets compared to the initialization with Morfessor Baseline. While FlatCat cannot keep the accuracy of the suffix boundaries at as high level as CRF, it clearly improves the stem splitting.

3.2 Information retrieval

Stemming has been shown to improve IR results (Kurimo et al., 2009), by removing inflection that is often not relevant to the query. The morph categories make it possible to simulate stemming by removing morphs categorized as prefixes or suffixes. As longer affixes are more likely to be meaningful, we limited the affix removal to morphs of at most 3 letters. For methods that use morph categories, we report two IR results: the first using all the data and the second with short affix removal (SAR) applied.

In the IR results, we include the topline methods from Morpho Challenge: Snowball Porter stemmer (Porter, 1980) for English and "TWOL first" for Finnish. The latter selects the lemma from the first of the possible analyses given by the morphological analyzer FINTWOL (Lingsoft, Inc.) based on the

(a) English.

	STM	STM + STM			PRE + STM			STM + SUF			STM + SUF + SUF		
Method	Pre	Pre	Rec	F	Pre	Rec	F	Pre	Rec	F	Pre	Rec	F
U CatMAP	**.90**	.94	.63	.75	**.91**	.64	.75	.87	.45	.59	.90	.51	.65
SS Baseline	.64	.93	.77	.84	.82	**.74**	**.77**	.83	.86	.84	.91	.79	.85
SS FlatCat	.68	.94	.65	.77	.78	.62	.69	.86	.88	.87	.94	.79	.86
SS CRF+FlatCat	.68	**.95**	**.78**	**.86**	.78	.66	.72	.87	.89	.88	.94	.80	.87
S CRF	.78	.94	.72	.81	.85	.59	.69	**.92**	**.91**	**.91**	**.95**	**.82**	**.88**

(b) Finnish.

	STM	STM + STM			PRE + STM			STM + SUF			STM + SUF + SUF		
Method	Pre	Pre	Rec	F	Pre	Rec	F	Pre	Rec	F	Pre	Rec	F
U CatMAP	**.77**	.90	**.97**	**.94**	.88	**.96**	**.92**	.67	.46	.54	.68	.38	.49
SS Baseline	.50	.82	.88	.85	.73	.83	.78	.64	.85	.73	.76	.78	.77
SS FlatCat	.49	**.91**	.95	.93	.80	.89	.85	.67	.84	.75	.77	.75	.76
SS CRF+FlatCat	.53	**.91**	.96	**.94**	.84	.94	.88	.71	.88	.79	.80	.79	.79
S CRF	.68	.88	.91	.89	**.90**	.91	.91	**.83**	**.91**	**.87**	**.91**	**.85**	**.88**

Table 4: Results of BPR experiments with different morph category patterns. Best results for each measure have been hilighted using boldface.

two-level model by Koskenniemi (1983). As baseline results we also include unsegmented word forms and truncating each word after the first five letters (First 5).

The results of the IR experiment are shown in Table 5. FlatCat provides the highest score for Finnish. The English scores are similar to those of the semi-supervised Baseline. FlatCat performs better than CRF for both languages. This is explained by the higher level of consistency in the segmentations produced by FlatCat, which makes the resulting morphs more useful as query terms. The number of morphs in the lexicons of FlatCat initialized using CRF are 108 391 (English), 46 123 (Finnish) and 74 193 (Turkish), which is much smaller than the respective morph lexicon sizes counted from the CRF segmentation: 339 682 (English), 396 869 (Finnish) and 182 356 (Turkish). This decrease in lexicon size indicates a more structured segmentation.

The IR performance of semi-supervised FlatCat benefits from the removal of short affixes for English when initialized by CRF, and Finnish for both initializations. It also improves the results of unsupervised FlatCat and CatMAP for Finnish, but lowers the precision for English. A possible explanation is that the unsupervised methods do not analyze the suffixes with a high enough accuracy.

4 Conclusions

We have introduced a new variant of the Morfessor method, Morfessor FlatCat. It predicts both morphs and their categories based on unannotated data, but also annotated training data can be provided. It was shown to outperform earlier Morfessor methods in the semi-supervised learning task for English, Finnish and Turkish.

The purely supervised CRF-based segmentation method proposed by Ruokolainen et al. (2013) outperforms FlatCat for Finnish and reaches the same level for English. However, we show that a discriminative model such as CRF gives inconsistent segmentations that do not work as well in a practical application: In English and Finnish information retrieval tasks, FlatCat clearly outperformed the CRF-based segmentation.

We see two major directions for future work. Currently Morfessor FlatCat, like most Morfessor methods, assumes that words in a sentence occur independently. Making use of the sentence context in which words occur would, however, allow making Part-Of-Speech -like distinctions. These distinctions could

(a) English.

Rank		Method	SAR	MAP
1	–	Snowball Porter	–	0.4092
2	SS	Baseline	–	0.3855
3	**SS**	**FlatCat**	**No**	**0.3837**
4	**SS**	**FlatCat**	**Yes**	**0.3821**
5	**SS**	**CRF+FlatCat**	**Yes**	**0.3810**
6	**SS**	**CRF+FlatCat**	**No**	**0.3788**
7	S	CRF	–	0.3771
8	W	Baseline	–	0.3761
9	U	Baseline	–	0.3695
10	U	CatMAP	No	0.3682
11	U	CatMAP	Yes	0.3653
12	**W**	**FlatCat**	**No**	**0.3651**
13	–	(First 5)	–	0.3648
14	**W**	**FlatCat**	**Yes**	**0.3606**
15	**U**	**FlatCat**	**No**	**0.3486**
16	**U**	**FlatCat**	**Yes**	**0.3451**
17	–	(Words)	–	0.3303

(b) Finnish.

Rank		Method	SAR	MAP
1	**W**	**FlatCat**	**No**	**0.5057**
2	**W**	**FlatCat**	**Yes**	**0.5029**
3	**SS**	**FlatCat**	**Yes**	**0.4987**
4	–	TWOL first	–	0.4973
5	**SS**	**CRF+FlatCat**	**Yes**	**0.4912**
6	U	CatMAP	Yes	0.4884
7	U	CatMAP	No	0.4865
8	**SS**	**CRF+FlatCat**	**No**	**0.4826**
9	**SS**	**FlatCat**	**No**	**0.4821**
10	–	(First 5)	–	0.4757
11	SS	Baseline	–	0.4722
12	S	CRF	–	0.4660
13	W	Baseline	–	0.4582
14	U	Baseline	–	0.4378
15	**U**	**FlatCat**	**Yes**	**0.4349**
16	**U**	**FlatCat**	**No**	**0.4334**
17	–	(Words)	–	0.3483

Table 5: Information Retrieval results. Results of the method presented in this paper are hilighted using boldface. Mean Average Precision is abbreviated as MAP. Short affix removal is abbreviated as SAR.

help disambiguate inflections of different lexemes that have the same surface form but should be analyzed differently (Can and Manandhar, 2013).

The second direction is removal of the assumption that a morphology consists only of concatenative processes. Introducing transformations to model allomorphy in a similar manner as Kohonen et al. (2009) would allow finding the shared abstract morphemes underlying different allomorphs. This could be especially beneficial in information retrieval and machine translation applications.

Acknowledgments

This research has been supported by European Community's Seventh Framework Programme (FP7/2007–2013) under grant agreement n°287678 and the Academy of Finland under the Finnish Centre of Excellence Program 2012–2017 (grant n°251170) and the LASTU Programme (grants n°256887 and 259934). The experiments were performed using computer resources within the Aalto University School of Science "Science-IT" project. We thank Teemu Ruokolainen for his help with the experiments.

References

Burcu Can and Suresh Manandhar. 2013. Dirichlet processes for joint learning of morphology and PoS tags. In *Proceedings of the International Joint Conference on Natural Language Processing*, pages 1087–1091, Nagoya, Japan, October.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In Mike Maxwell, editor, *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30, Philadelphia, PA, USA, July. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2004. Induction of a simple morphology for highly-inflecting languages. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology*, pages 43–51, Barcelona, Spain, July. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In Timo Honkela, Ville Könönen, Matti Pöllä, and Olli Simula, editors, *Proceedings of AKRR'05,*

International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning, pages 106–113, Espoo, Finland, June. Helsinki University of Technology, Laboratory of Computer and Information Science.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1):3:1–3:34, January.

Harald Hammarström and Lars Borin. 2011. Unsupervised learning of morphology. *Computational Linguistics*, 37(2):309–350, June.

Oskar Kohonen, Sami Virpioja, and Mikaela Klami. 2009. Allomorfessor: Towards unsupervised morpheme analysis. In *Evaluating Systems for Multilingual and Multimodal Information Access: 9th Workshop of the Cross-Language Evaluation Forum, CLEF 2008, Aarhus, Denmark, September 17–19, 2008, Revised Selected Papers*, volume 5706 of *Lecture Notes in Computer Science*, pages 975–982. Springer Berlin / Heidelberg, September.

Oskar Kohonen, Sami Virpioja, and Krista Lagus. 2010. Semi-supervised learning of concatenative morphology. In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 78–86, Uppsala, Sweden, July. Association for Computational Linguistics.

Kimmo Koskenniemi. 1983. *Two-level morphology: A general computational model for word-form recognition and production*. Ph.D. thesis, University of Helsinki.

Mikko Kurimo, Sami Virpioja, Ville T. Turunen, Graeme W. Blackwood, and William Byrne. 2009. Overview and results of Morpho Challenge 2009. In *Working Notes for the CLEF 2009 Workshop*, Corfu, Greece, September.

Mikko Kurimo, Sami Virpioja, Ville Turunen, and Krista Lagus. 2010a. Morpho Challenge 2005-2010: Evaluations and results. In Jeffrey Heinz, Lynne Cahill, and Richard Wicentowski, editors, *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 87–95, Uppsala, Sweden, July. Association for Computational Linguistics.

Mikko Kurimo, Sami Virpioja, and Ville T. Turunen. 2010b. Overview and results of Morpho Challenge 2010. In *Proceedings of the Morpho Challenge 2010 Workshop*, pages 7–24, Espoo, Finland, September. Aalto University School of Science and Technology, Department of Information and Computer Science. Technical Report TKK-ICS-R37.

Paul Ogilvie and James P Callan. 2001. Experiments using the Lemur toolkit. In *TREC*, volume 10, pages 103–108.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Jorma Rissanen. 1978. Modeling by shortest data description. *Automatica*, 14:465–471.

Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja, and Mikko Kurimo. 2013. Supervised morphological segmentation in a low-resource learning setting using conditional random fields. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 29–37, Sofia, Bulgaria, August. Association for Computational Linguistics.

Sami Virpioja, Ville T Turunen, Sebastian Spiegler, Oskar Kohonen, and Mikko Kurimo. 2011. Empirical comparison of evaluation methods for unsupervised learning of morphology. *Traitement Automatique des Langues*, 52(2):45–90.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for Morfessor Baseline. Report 25/2013 in Aalto University publication series SCIENCE + TECHNOLOGY, Department of Signal Processing and Acoustics, Aalto University.

Japanese Word Reordering Integrated with Dependency Parsing

Kazushi Yoshida[1,a] **Tomohiro Ohno**[2,b] **Yoshihide Kato**[3,c] **Shigeki Matsubara**[1,d]
[1]Graduate School of Information Science, Nagoya University, Japan
[2]Information Technology Center, Nagoya University, Japan
[3]Information & Communications, Nagoya University, Japan
[a)]yoshida@db.ss.is.nagoya-u.ac.jp [b)]ohno@nagoya-u.jp
[c)]yoshihide@icts.nagoya-u.ac.jp [d)]matubara@nagoya-u.jp

Abstract

Although Japanese has relatively free word order, Japanese word order is not completely arbitrary and has some sort of preference. Since such preference is incompletely understood, even native Japanese writers often write Japanese sentences which are grammatically well-formed but not easy to read. This paper proposes a method for reordering words in a Japanese sentence so that the sentence becomes more readable. Our method can identify more suitable word order than conventional word reordering methods by concurrently performing dependency parsing and word reordering instead of sequentially performing the two processing steps. As the result of an experiment on word reordering using newspaper articles, we confirmed the effectiveness of our method.

1 Introduction

Japanese has relatively free word order, and thus Japanese sentences which make sense can be written without having a strong awareness of word order. However, Japanese word order is not completely arbitrary and has some sort of preference. Since such preference is incompletely understood, even native Japanese writers often write Japanese sentences which are grammatically well-formed but not easy to read. The word reordering of such sentences enables the readability to be improved.

There have been proposed some methods for reordering words in a Japanese sentence so that the sentence becomes easier to read (Uchimoto et al., 2000; Yokobayashi et al., 2004). In addition, there exist a lot of researches for estimating appropriate word order in various languages (Filippova and Strube, 2007; Harbusch et al., 2006; Kruijff et al., 2001; Ringger et al., 2004; Shaw and Hatzivassiloglou, 1999). Although most of these previous researches used syntactic information, the sentences they used there were what had been previously parsed. It is a problem that word reordering suffers the influence of parsing errors. Furthermore, as the related works, there are various researches on word reordering for improving the performance of statistical machine translation (Goto et al., 2012; Elming, 2008; Ge, 2010; Christoph and Hermann, 2003; Nizar, 2007). These researches consider information as to both a source language and a target language to handle word order differences between them. Therefore, their problem setting is different from that for improving the readability of a single language.

This paper proposes a method for reordering words in a Japanese sentence so that the sentence becomes easier to read for revision support. Our proposed method concurrently performs dependency parsing and word reordering for an input sentence of which the dependency structure is still unknown. Our method can identify more suitable word order than conventional word reordering methods because it can concurrently consider the preference of both word order and dependency. An experiment using newspaper articles showed the effectiveness of our method.

2 Word Order and Dependency in Japanese Sentences

There have been a lot of researches on Japanese word order in linguistics (for example, Nihongo Kijutsu Bunpo Kenkyukai, 2009; Saeki, 1998), which have marshalled fundamental contributing factors which

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1186–1196, Dublin, Ireland, August 23-29 2014.

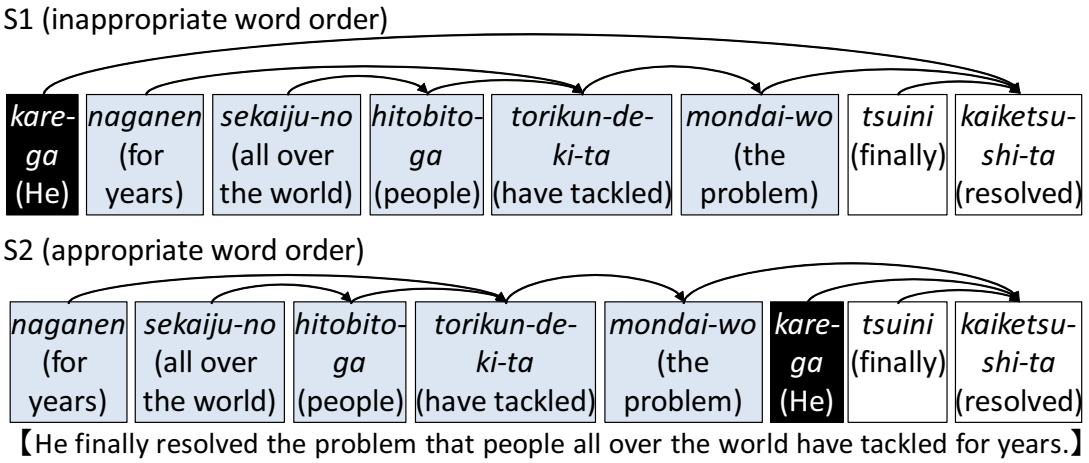


Figure 1: Example of inappropriate/appropriate word order

decide the appropriate word order in detail. In a Japanese sentence, a predicate of the main clause is fundamentally placed in last position, and thus, case elements, adverbial elements, or subordinate clauses are located before it. In addition, case elements are basically placed in the order of a nominative, a dative and an accusative. However, the basic order of case elements is often changed by being influenced from grammatical and discourse factors. For example, it is pointed out that a long case element has strong preference to be located at the beginning of a sentence even if the element is not nominative, as shown in Figure 1.

In Figure 1, a box and an arrow express a *bunsetsu*[1] and a dependency relation respectively. Both the sentences S1 and S2 have the same meaning which is translated as "He finally resolved the problem that people all over the world have tackled for years" in English. The difference between S1 and S2 is just in their word orders in Japanese.

The word order of S1 is more difficult to read than that of S2 because the distance between the bunsetsu "*kare-ga* (He)" and its modified bunsetsu "*kaiketsu-shi-ta* (resolved)" is large and thus the loads on working memory become large. This example suggests that if the dependency structure of S1 is identified, that information is useful to reorder the word order of S1 to that of S2 so that it becomes easier to read. In fact, most of the conventional word reordering methods have reordered words using the previously parsed dependency structure. However, the word order of S1 is thought to be more difficult to parse than that of S2 because dependency parsers are usually trained on syntactically annotated corpora in which sentences have the appropriate word order such as that in S2. This is why it is highly possible that dependency parsing can achieve a higher accuracy by changing the word order of S1 to that of S2 in advance.

The above observations indicate that word reordering and dependency parsing depend on each other. Therefore, we consider it is more desirable to concurrently perform the two processings than to sequentially perform them.

3 Word Reordering Method

In our method, a sentence, on which morphological analysis and bunsetsu segmentation have been performed, is considered as the input[2]. We assume that the input sentence might have unsuitable word order,

[1] *Bunsetsu* is a linguistic unit in Japanese that roughly corresponds to a basic phrase in English. A bunsetsu consists of one independent word and zero or more ancillary words. A dependency relation in Japanese is a modification relation in which a modifier bunsetsu depends on a modified bunsetsu. That is, the modifier bunsetsu and the modified bunsetsu work as modifier and modifyee, respectively.

[2] In order to focus attention on the comparison between our method and the conventional method, we assumed the input on which the lower layer processings than dependency parsing have been performed. Even if morphological analysis and bunsetsu segmentation are automatically performed on input sentences which have unsuitable word order, we can expect the accuracies

which is not easy to read but grammatically well-formed. Our method identifies the suitable word order which is easy to read by concurrently performing dependency parsing.

The simultaneous performing of dependency parsing and word reordering is realized by searching for the maximum-likelihood pattern of word order and dependency structure for the input sentence. Note our method reorders bunsetsus in a sentence without paraphrasing and does not reorder morphemes within a bunsetsu.

3.1 Probabilistic Model for Word Reordering

When a sequence of bunsetsus in an input sentence $B = b_1 \cdots b_n$ is provided, our method identifies the structure S which maximizes $P(S|B)$. The structure S is defined as a tuple $S = \langle O, D \rangle$ where $O = \{o_{1,2}, o_{1,3}, \cdots, o_{1,n}, \cdots, o_{i,j}, \cdots, o_{n-2,n-1}, o_{n-2,n}, o_{n-1,n}\}$ is the word order pattern after reordering and $D = \{d_1, \cdots, d_{n-1}\}$ is dependency structure. Here, $o_{i,j}$ $(1 \leq i < j \leq n)$ expresses the order between b_i and b_j after reordering. $o_{i,j}$ is 1 if b_i is located before b_j, and is 0 otherwise. In addition, d_i expresses the dependency relation whose modifier bunsetsu is b_i.

$P(S|B)$ for a $S = \langle O, D \rangle$ is calculated as follows.

$$\begin{aligned} P(S|B) &= P(O, D|B) \\ &= \sqrt{P(O|B) \times P(D|O, B) \times P(D|B) \times P(O|D, B)} \end{aligned} \quad (1)$$

Formula (1) is obtained for the product of the following two formulas. According to the probability theory, the calculated result of Formula (1) is equal to those of Formulas (2) and (3). However, in practice, since each factor in the formulas is estimated based on the corpus used for training, the calculated results of these formulas are different from each other. We use Formula (1) to estimate $P(S|B)$ by using both values of $P(D|O, B)$ and $P(O|D, B)$. In fact, we pre-experimentally confirmed that the calculated result of Formula (1) was better than those of the others.

$$P(O, D|B) = P(O|B) \times P(D|O, B) \quad (2)$$

$$P(O, D|B) = P(D|B) \times P(O|D, B) \quad (3)$$

Assuming that order $o_{i,j}$ between two bunsetsus is independent of that between other two bunsetsus and that each dependency relation d_i is independent of the others, each factor in Formula (1) can be approximated as follows:

$$P(O|B) \cong \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} P(o_{i,j}|B) \quad (4)$$

$$P(D|O, B) \cong \prod_{i=1}^{n-1} P(d_i|O, B) \quad (5)$$

$$P(D|B) \cong \prod_{i=1}^{n-1} P(d_i|B) \quad (6)$$

$$P(O|D, B) \cong \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} P(o_{i,j}|D, B) \quad (7)$$

where $P(o_{i,j}|B)$ is the probability that the order between b_i and b_j is $o_{i,j}$ when B is provided, $P(d_i|O, B)$ is the probability that the dependency relation whose modifier bunsetsu is b_i is d_i when the sentence generated by reordering B according to O is provided, $P(d_i|B)$ is the probability that the dependency relation whose modifier bunsetsu is b_i is d_i when B is provided, and $P(o_{i,j}|D, B)$ is the probability that the order between b_i and b_j is $o_{i,j}$ when B where the dependency relation is D is provided. These probabilities are estimated by the maximum entropy method.

remain comparatively high. This is because their processings use mainly local information.

To estimate $P(d_i|O, B)$, we used the features used in Uchimoto et al. (1999) except when eliminating features about Japanese commas (called *toten*, which is a kind of punctuation) and quotation marks. To estimate $P(d_i|B)$, we used the features which can be obtained without information about the order of input bunsetsus among the features used in estimating $P(d_i|O, B)$. To estimate $P(o_{i,j}|D, B)$, if b_i and b_j modifies the same bunsetsu, we used the features used in Uchimoto et al. (2000), except when eliminating features about parallel relations and semantic features. Otherwise, we used the features left after eliminating features about modified bunsetsus from those used in the above-mentioned case. To estimate $P(o_{i,j}|B)$, we used the features which can be obtained without dependency information among the features used to estimate $P(O_{i,j}|D, B)$.

3.2 Search Algorithm

Since there are a huge number of the structures $S = \langle O, D\rangle$ which are theoretically possible for an input sentence B, an efficient algorithm is desired. However, since O and D are dependent on each other, it is difficult to find the optimal structure efficiently. In our research, we extend CYK algorithm used in conventional dependency parsing to efficiently find the suboptimal $S = \langle O, D\rangle$ which maximizes $P(S|B)$ efficiently.

Our research assumes that an input sentence, which is grammatically well-formed, is reordered without changing the meaning so that the sentence becomes much easier to read. From this assumption, we can use following conditions for efficient search:

1. The dependency structure of an input sentence should satisfy the following Japanese syntactic constraints under the input word order:
   - No dependency is directed from right to left.
   - Dependencies don't cross each other.
   - Each bunsetsu, except the last one, depends on only one bunsetsu.
2. Even after the words are reordered, the dependency structure should satisfy the above-mentioned Japanese syntactic constraints under the changed word order.
3. The dependency structures of a sentence before and after reordering should be identical.

Using the condition 1 and the condition 3, we can narrow down the search space of D to dependency structures that satisfy Japanese syntactic constraints under the input word order. Furthermore, the search space of O can be narrowed down to the word order patterns derived from the above narrowed dependency structures based on the conditions 2 and 3. That is, after dependency structures possible for an input sentence are narrowed down, we just have to find the word order patterns after reordering so that each of the dependency structures is maintained and satisfies the Japanese syntactic constraints even under the changed word order.

On the other hand, it is well known that CYK algorithm can efficiently find the optimal dependency structure which satisfies Japanese syntactic constraints. Therefore, in our research, we have extended the CYK algorithm for the conventional dependency parsing so that it can find the suboptimal D and O from among the dependency structures and word order patterns which satisfy the conditions 1, 2 and 3.

3.2.1 Word Reordering Algorithm

Algorithm 1 shows our word reordering algorithm. In our algorithm, the $n \times n$ triangular matrix $M_{i,j}(1 \leq i \leq j \leq n)$ such as the left-side figure in Figure 2 is prepared for an input sentence consisting of n numbers of bunsetsus. $M_{i,j}$, the element of the triangular matrix M in the i-th row and j-th column, is filled by $argmax_{S_{i,j}} P(S_{i,j}|B_{i,j})$, which is the maximum-likelihood structure for an input subsequence $B_{i,j} = b_i \cdots b_j$. In this section, for convenience of explanation, we represent $S_{i,j}$ as a sequence of dependency relations $d_x(i \leq x \leq j)$. For example, $S_{i,j} = d_i d_{i+1} \cdots d_j^0$ means that the first bunsetsu is b_i, the second is b_{i+1}, $\cdots$, the last is b_j, and the dependency structure is $\{d_i, d_{i+1}, \cdots, d_{j-1}\}$. Here, if we need to clearly specify the modified bunsetsu, we represent the dependency relation that bunsetsu

Algorithm 1 word reordering algorithm

1: **input** $B_{1,n} = b_1 \cdots b_n$ // input sentence
2: **set** $M_{i,j}$ $(1 \leq i \leq j \leq n)$ // triangular matrix
3: **set** $C_{i,j}$ $(1 \leq i \leq j \leq n)$ // set of structure candidates
4: **for** $i = 1$ **to** n **do**
5: $\quad M_{i,i} = d_i^0$
6: **end for**
7: **for** $d = 1$ **to** $n - 1$ **do**
8: $\quad$ **for** $i = 1$ **to** $n - d$ **do**
9: $\quad\quad j = i + d$
10: $\quad\quad$ **for** $k = i$ **to** $j - 1$ **do**
11: $\quad\quad\quad C_{i,j} = C_{i,j} \cup ConcatReorder(M_{i,k}, M_{k+1,j})$
12: $\quad\quad$ **end for**
13: $\quad\quad M_{i,j} = argmax_{S_{i,j} \in C_{i,j}} P(S_{i,j}|B_{i,j})$
14: $\quad$ **end for**
15: **end for**
16: **return** $M_{1,n}$

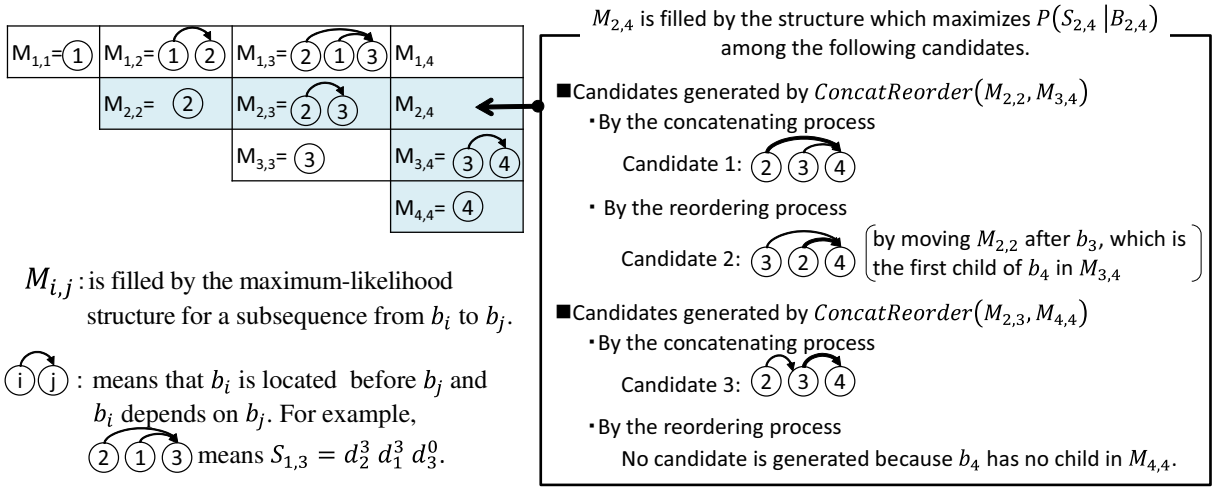


Figure 2: Execution example of our search algorithm

b_x modifies b_y as d_x^y. In addition, d_j^0 means that the last bunsetsu of the subsequence don't modify any bunsetsu.

First, the statements of the lines 4 to 6 fill each of diagonal elements $M_{i,i}$ $(1 \leq i \leq n)$ with d_i^0. Next, the statements of the lines 7 to 15 fill $M_{i,j}$ in turn toward the upper right $M_{1,n}$ along the diagonal line, starting from the diagonal elements $M_{i,i}$. The maximum-likelihood structure which should fill an $M_{i,j}$ is found as follows:

The statements of the lines 10 to 12 repeat the process of generating candidates of the maximum-likelihood structure from $M_{i,k}$ and $M_{k+1,j}$ by the function $ConcatReorder$, and adding them to the set of structure candidates $C_{i,j}$. The function $ConcatReorder$ takes two arguments of $M_{i,k}$ and $M_{k+1,j}$ and returns the set of candidates of the maximum-likelihood structure which should fill $M_{i,j}$. The function $ConcatReorder$ is composed of two processes: concatenating process and reordering process. First, the concatenating process generates a candidate by simply concatenating $M_{i,k}$ and $M_{k+1,j}$ in turn about the word order and connecting $M_{i,k}$ and $M_{k+1,j}$ by the dependency relation between the last bunsetsus of them about the dependency structure, without changing the internal structure of each of them. For example, when $M_{i,k} = d_i d_{i+1} \cdots d_{k-1} d_k^0$ and $M_{k+1,j} = d_{k+1} d_{k+2} \cdots d_{j-1} d_j^0$ are given as the argument, the concatenating process generates "$d_i d_{i+1} \cdots d_{k-1} d_k^j d_{k+1} d_{k+2} \cdots d_{j-1} d_j^0$."

Second, the reordering process generates candidates by reordering words in the candidate generated by the concatenating process. The reordering is executed on the following conditions. The first condition is that the dependency structure is maintained and satisfies the Japanese syntactic constraints even under the changed word order. The second condition is that the order of any two words within each of $M_{i,k}$ and $M_{k+1,j}$ is maintained. Concretely, the first reordered candidate is generated by moving $M_{i,k}$ after the first (leftmost) child[3] of the last bunsetsu of $M_{k+1,j}$ among the children in $M_{k+1,j}$. Then, the second reordered candidate is generated by moving $M_{i,k}$ after the second child. The reordering process is continued until the last reordered candidate is generated by moving $M_{i,k}$ after the last child. That is, the number of candidates generated by the reordering process is equal to the number of children of the last bunsetsu in $M_{k+1,j}$. For example, when $M_{i,k} = d_i d_{i+1} \cdots d_{k-1} d_k^0$ and $M_{k+1,j} = d_{k+1}^j d_{k+2}^j \cdots d_{j-1}^j d_j^0$, which means all bunsetsus except the last one depend on the last one, are given, the reordering process generates the following $j-k-1$ candidates: "$d_{k+1}^j d_i d_{i+1} \cdots d_{k-1} d_k^j d_{k+2}^j d_{k+3}^j \cdots d_{j-1}^j d_j^0$," "$d_{k+1}^j d_{k+2}^j d_i d_{i+1} \cdots d_{k-1} d_k^j d_{k+3}^j d_{k+4}^j \cdots d_{j-1}^j d_j^0$," ..., and "$d_{k+1}^j d_{k+2}^j \cdots d_{j-1}^j d_i d_{i+1} \cdots d_{k-1} d_k^j d_j^0$." Therefore, in this case, the function $ConcatReorder$ finally returns the set of candidates of which size is $j-k$, which includes the candidates generated by the reordering process and a candidate generated by the concatenating process. Next, in the line 13, our algorithm fills in $argmax_{S_{i,j} \in C_{i,j}} P(S_{i,j}|B_{i,j})$ which is the maximum-likelihood structure for a subsequence $B_{i,j}$ on $M_{i,j}$.

Finally, our algorithm outputs $M_{1,n}$ as the maximum-likelihood structure of word order and dependency structure for the input sentence.

Note that if the function *ConcatReorder* is changed to the function *Concat* in the line 11, our algorithm becomes the same as CYK algorithm used in the conventional dependency parsing. The function $Concat$ takes two arguments of $M_{i,k}$ and $M_{k+1,j}$ and generates a candidate of the maximum-likelihood structure which should fill $M_{i,j}$ by the same way as the concatenating process in the function $ConcatReorder$. Then, the function $Concat$ returns the set which has the generated candidate as a element, of which size is 1.

3.2.2 Execution Example of Word Reordering Algorithm

Figure 2 represents an example of execution of our word reordering algorithm in $n = 4$. The left side of Figure 2 represents the triangle diagram which has 4×4 dimensions. The elements of the triangle diagram $M_{1,1}, M_{2,2}, M_{3,3}, M_{4,4}, M_{1,2}, M_{2,3}, M_{3,4}$, and $M_{1,3}$ have already been filled in turn, and $M_{2,4}$ is being filled. The right side of Figure 2 shows the process of calculating the maximum-likelihood structure which should fill $M_{2,4}$. First, in the loop from the line 10 to the line 12 in Algorithm 1, two structure candidates are generated by $ConcatReorder(M_{2,2}, M_{3,4})$. The candidate 1 is generated by the concatenating process, that is, by simply concatenating $M_{2,2}$ and $M_{3,4}$ and connecting the last bunsetsu of $M_{2,2}$ and that of $M_{3,4}$. The candidate 2 is generated by the reordering process, that is, by moving $M_{2,2}$ after b_3, which is the first child of b_4 in $M_{3,4}$. Second, the candidate 3 is generated by the concatenating process in $ConcatReorder(M_{2,3}, M_{4,4})$. On the other hand, the reordering process in $ConcatReorder(M_{2,3}, M_{4,4})$ generates no candidates because b_4 has no child in $M_{4,4}$. Among the three structures generated in the above way, the structure which maximizes $P(S_{2,4}|B) = P(O_{2,4}, D_{2,4}|B_{2,4})$ fills $M_{2,4}$.

4 Experiment

To evaluate the effectiveness of our method, we conducted an experiment on word reordering by using Japanese newspaper articles.

4.1 Outline of Experiment

In the experiment, as the test data, we used sentences generated by only changing the word order of newspaper article sentences in Kyoto Text Corpus (Kurohashi and Nagao, 1998), maintaining the dependency structure. That is, we artificially generated sentences which made sense but were not easy to read,

[3]When b_i depends on b_j, we call b_i as a child of b_j. Furthermore, if b_j has more than or equal to one child, the children are numbered from left to right based on their positions.

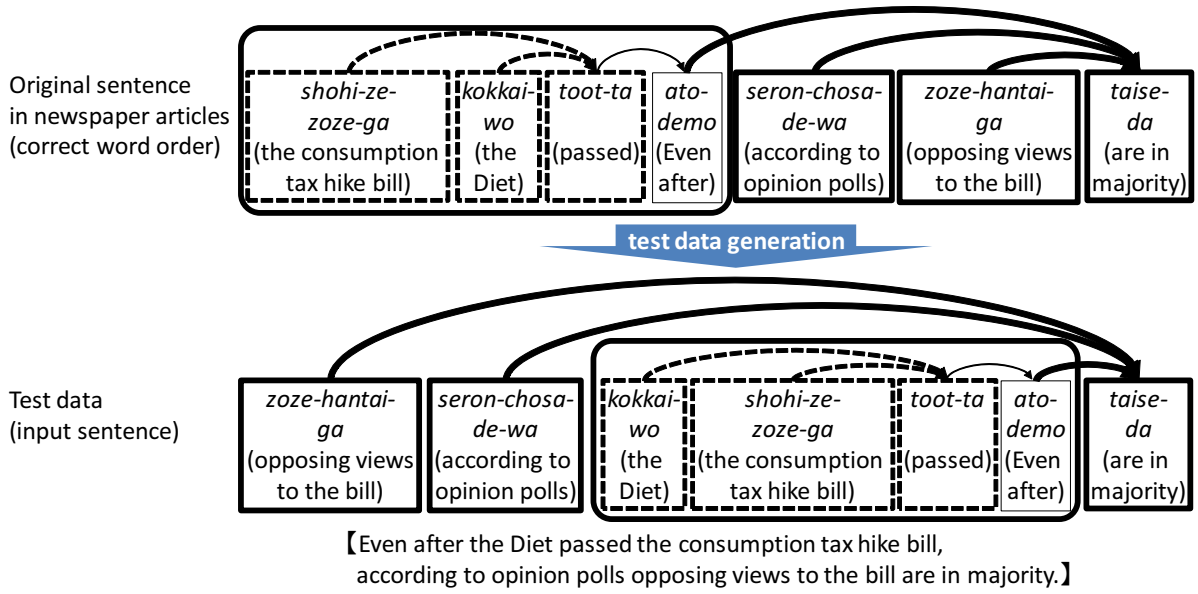


Figure 3: Example of test data generation

in order to focus solely on problems caused by unsuitable word order. Figure 3 shows an example of the test data generation. The generation procedure is as follows:

1. Find a bunsetsu modified by multiple bunsetsus from the sentence end.
2. Change randomly the order of the sub-trees which modify such bunsetsu.
3. Iterate 1 and 2 until reaching the beginning of the sentence.

In Figure 3, the bunsetsus "*taise-da* (are in the majority)" and "*toot-ta* (passed)" are found as bunsetsus modified by multiple bunsetsus. For example, when "*toot-ta* (passed)" is found, the order of "*shohi-ze-zoze-ga* (the consumption tax hike bill)" and "*kokkai-wo* (the Diet)" is randomly changed. In this experiment, all Japanese commas (*toten*) in a sentence, and sentences which have quotation marks were removed.

In this way, we artificially generated 865 sentences (7,620 bunsetsus) from newspaper articles of Jan. 9 in Kyoto Text Corpus and used them as the test data. As the training data, we used 7,976 sentences in 7 days' newspaper articles (Jan. 1, 3-8). Here, we used the maximum entropy method tool (Zhang, 2008) with the default options except "-i 1000."

In the evaluation of word reordering, we obtained the following two measurements, which are defined by Uchimoto et al. (2000):

- **complete agreement:** the percentage of the sentences in which all words' order completely agrees with that of the original sentence.
- **pair agreement:** the percentage of the pairs of bunsetsus whose word order agrees with that in the original sentence. (For example, in Figure 3, if the word order of the input sentence is not changed after reordering, the pair agreement is 52.4% ($= 11/{}_7C_2$) because the 11 pairs out of the ${}_7C_2$ pairs are the same as those in the original sentence.)

In the evaluation of dependency parsing, we obtained the **dependency accuracy** (the percentage of correctly analyzed dependencies out of all dependencies) and **sentence accuracy** (the percentage of the sentences in which all the dependencies are analyzed correctly), which are defined by Sekine et al. (2000).

For comparison, we established two baselines. Both of the baselines execute the dependency parsing primarily, and then, perform the word reordering by using the conventional word reordering method

Table 1: Experimental results (word reordering)

	pair agreement	complete agreement
our method	77.3% (30,190/38,838)	25.7% (222/865)
baseline 1	75.4% (29,279/38,838)*	23.8% (206/865)
baseline 2	74.8% (29,067/38,838)*	23.5% (203/865)
no reordering	61.5% (23,886/38,838)*	8.0% (69/865)*

Note that the agreements followed by * differ significantly from those of our method ($p < 0.05$).

Table 2: Experimental results (dependency parsing)

	dependency accuracy	sentence accuracy
our method	78.4% (5,293/6,755)	35.3% (305/865)
baseline 1	79.2% (5,350/6,755)	31.6% (273/865)*
baseline 2	81.2% (5,487/6,755)*	32.1% (278/865)*

Note that the accuracies followed by * differ significantly from those of our method ($p < 0.05$).

(Uchimoto et al., 1999). The difference between the two is the method of dependency parsing. The baselines 1 and 2 use the dependency parsing method proposed by Uchimoto et al. (2000) and the dependency parsing tool CaboCha (Kudo and Matsumoto, 2002), respectively. The features used for the word reordering in both the baselines are the same as those used to estimate $P(o_{i,j}|D, B)$ in our method. Additionally, the features used for the dependency parsing in the baseline 1 are the same as those used to estimate $P(d_i|O, B)$ in our method.

4.2 Experimental Results

Table 1 shows the experimental results on word reordering of our method and the baselines. Here, the last row shows the agreements measured by comparing the input word order with the correct word order. The agreements mean the values which can be achieved with no reordering[4]. The pair and complete agreements of our method were highest among all. The pair agreement of our method is significantly different from those of both the baselines ($p < 0.05$) although there is no significant difference between the complete agreements of them.

Next, Table 2 shows the experimental results on dependency parsing. The sentence accuracy of our method is significantly higher than those of both the baselines ($p < 0.05$). On the other hand, the dependency accuracy of our method is significantly lower than that of the baseline 2 although there is no significant difference between the dependency accuracies of our method and the baseline 1 ($p > 0.05$). Here, if the input sentences had the correct word order, the dependency accuracies of the baselines 1 and 2 were 86.4% (5,835/6,755) and 88.1% (5,950/6,755), respectively. We can see that the unsuitable word order caused a large decrease of the accuracies of the conventional dependency parsing methods. This is why the word order agreements of the baselines were decreased.

Figure 4 shows an example of sentences of which all bunsetsus were correctly reordered and the dependency structure was correctly parsed only by our method. We can see that our method can achieve the complicated word reordering. On the other hand, Figure 5 shows an example of sentences incorrectly reordered and parsed by our method. In this example, our method could not identify the correct modified bunsetsu and the appropriate position of the bunsetsu "*arikata-wo* (whole concept)." This is because the dependency probability between the bunsetsu "*arikata-wo* (whole concept)" and the bunsetsu "*fukume*

[4]Some input sentences were in complete agreement with the original ordering. There were some cases that the randomly reordered sentences accidentally have the same word order as the original ones. In addition, there were some sentences in which all bunsetsus except the last one depend on the next bunsetsu. The word order of such sentences is not changed by the test data generation procedure because the procedure is executed on condition of maintaining the dependency structure.

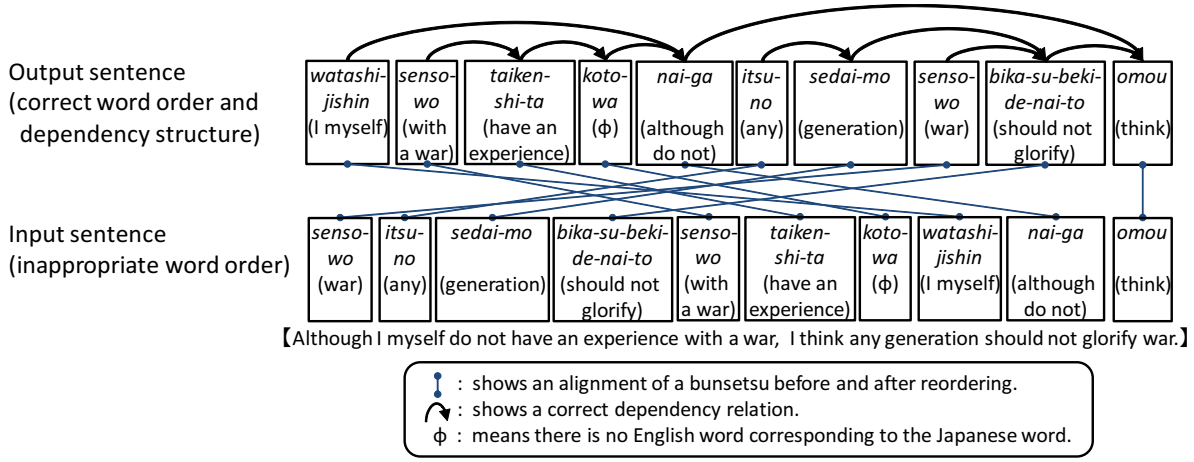


Figure 4: Example of sentences correctly reordered and parsed by our method

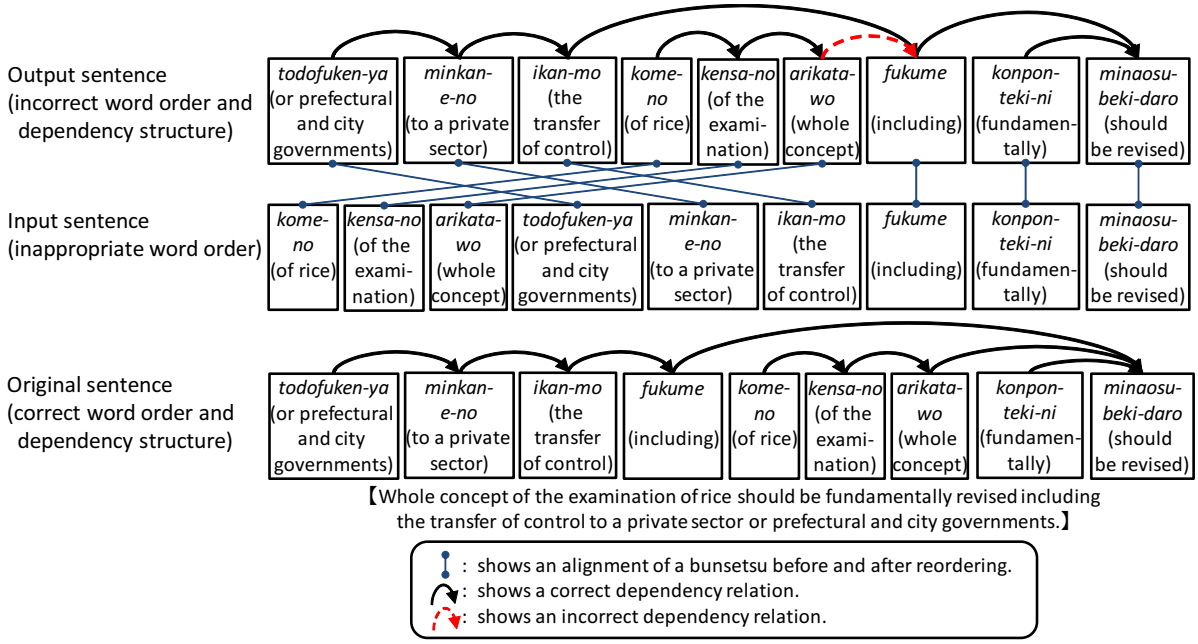


Figure 5: Example of sentences incorrectly reordered and parsed by our method

(including)" is higher than the one between the bunsetsu "*arikata-wo* (whole concept)" and the bunsetsu "*minaosu-beki-daro* (should be revised)", and the probability that the bunsetsu "*arikata-wo* (whole concept)" is located at the left side of "*fukume* (including)" is higher than that of the right side. Since the word order of the output sentence has a strong probability of causing a wrong interpretation like "The transfer of control to a private sector or prefectural and city governments should be fundamentally revised including whole concept of the examination of rice.", this reordering has a harmful influence on the comprehension. We need to study techniques for avoiding the word order which causes the change of meanings in an input sentence.

From the above, we confirmed the effectiveness of our method on word reordering and dependency parsing of a sentence of which the word order is not easy to read.

5 Conclusion

This paper proposed the method for reordering bunsetsus in a Japanese sentence. Our method can identify suitable word order by concurrently performing word reordering and dependency parsing. Based on the

idea of limiting the search space using the Japanese syntactic constraints, we made the search algorithm by extending the CYK algorithm used in the conventional dependency parsing, and found the optimal structure efficiently. The result of the experiment using newspaper articles showed the effectiveness of our method.

In our future works, we would like to collect sentences written by Japanese subjects who do not have much writing skills, to conduct an experiment using those sentences. In addition, we would like to conduct a subjective evaluation to investigate whether the output sentences are indeed more readable than the input ones.

Acknowledgments

This research was partially supported by the Grant-in-Aid for Young Scientists (B) (No.25730134) and Challenging Exploratory Research (No.24650066) of JSPS.

References

Tillmann Christoph and Ney Hermann. 2003. Word reordering and a dynamic programming beam search algorithm for statistical machine translation. *Computational Linguistics*, 29(1):97–133.

Jakob Elming. 2008. Syntactic reordering integrated with phrase-based SMT. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING2008)*, pages 209–216.

Katja Filippova and Michael Strube. 2007. Generating constituent order in German clauses. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL2007)*, pages 320–327.

Niyu Ge. 2010. A direct syntax-driven reordering model for phrase-based machine translation. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT2010)*, pages 849–857.

Geert-Jan M. Kruijff, Ivana Kruijff-Korbayová, John Bateman, and Elke Teich. 2001. Linear order as higher-level decision: Information structure in strategic and tactical generation. In *Proceedings of the 8th European Workshop on Natural Language Generation (ENLG2001)*, pages 74–83.

Isao Goto, Masao Utiyama, and Eiichiro Sumita. 2012. Post-ordering by parsing for Japanese-English statistical machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL2012)*, pages 311–316.

Karin Harbusch, Gerard Kempen, Camiel van Breugel, and Ulrich Koch. 2006. A generation-oriented workbench for performance grammar: Capturing linear order variability in German and Dutch. In *Proceedings of the 4th International Natural Language Generation Conference (INLG2006)*, pages 9–11.

Nihongo Kijutsu Bunpo Kenkyukai, editor. 2009. *Gendai nihongo bunpo 7 (Contemporary Japanese Grammar 7)*, pages 165–182. *Kuroshio Shuppan*. (In Japanese).

Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proceedings of the 6th Conference on Computational Natural Language Learning (CoNLL2002)*, pages 63–69.

Sadao Kurohashi and Makoto Nagao. 1998. Building a Japanese parsed corpus while improving the parsing system. In *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC '98)*, pages 719–724.

Habash Nizar. 2007. Syntactic preprocessing for statistical machine translation. In *Proceedings of the 11th Machine Translation Summit (MT SUMMIT XI)*, pages 215–222.

Eric Ringger, Michael Gamon, Robert C. Moore, David Rojas, Martine Smets, and Simon Corston-Oliver. 2004. Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING2004)*, pages 673–679.

Tetsuo Saeki. 1998. *Yosetsu nihonbun no gojun (Survey: Word Order in Japanese Sentences). Kuroshio Shuppan*. (In Japanese).

Satoshi Sekine, Kiyotaka Uchimoto, and Hitoshi Isahara. 2000. Backward beam search algorithm for dependency analysis of Japanese. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING2000)*, volume 2, pages 754–760.

James Shaw and Vasileios Hatzivassiloglou. 1999. Ordering among premodifiers. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*, pages 135–143.

Kiyotaka Uchimoto, Satoshi Sekine, and Hitoshi Isahara. 1999. Japanese dependency structure analysis based on maximum entropy models. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL '99)*, pages 196–203.

Kiyotaka Uchimoto, Masaki Murata, Qing Ma, Satoshi Sekine, and Hitoshi Isahara. 2000. Word order acquisition from corpora. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING2000)*, volume 2, pages 871–877.

Hiroshi Yokobayashi, Akira Suganuma, and Rin-ichiro Taniguchi. 2004. Generating candidates for rewriting based on an indicator of complex dependency and it's application to a writing tool. *Journal of Information Processing Society of Japan*, 45(5):1451–1459. (In Japanese).

Le Zhang. 2008. Maximum entropy modeling toolkit for Python and C++. `http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html`. [Online; accessed 1-March-2008].

Query-focused Multi-Document Summarization: Combining a Topic Model with Graph-based Semi-supervised Learning

Yanran Li and **Sujian Li***
Key Laboratory of Computational Linguistics,
Peking University, MOE, China
{liyanran,lisujian}@pku.edu.cn

Abstract

Graph-based learning algorithms have been shown to be an effective approach for query-focused multi-document summarization (MDS). In this paper, we extend the standard graph ranking algorithm by proposing a two-layer (i.e. sentence layer and topic layer) graph-based semi-supervised learning approach based on topic modeling techniques. Experimental results on TAC datasets show that by considering topic information, we can effectively improve the summary performance.

1 Introduction

Query-focused multi-document summarization (MDS) can facilitate users to grasp the main idea of the documents according to the users' concern. In query-focused summarization, one query is firstly proposed at the beginning of the documents. Then according to the given query and its influence on sentences, a ranking score is assigned to each of the sentences and higher ranked sentences are picked into a summary.

Among existing approaches, graph-based semi-supervised learning algorithms have been shown to be an effective way to impose a query's influence on sentences (Zhou et al, 2003; Zhou et al, 2004; Wan et al, 2007). Specifically, a weighted network is constructed where each sentence is modeled as a node and relationships between sentences are modeled as directed or undirected edges. With the assumption that a query is the most important node, initially, a positive score is assigned to the query and zero to the remaining nodes. All nodes then spread their ranking scores to their nearby neighbors via the weighted network. This spreading process is repeated until a global stable state is achieved, and all nodes obtain their final ranking scores.

The primary disadvantage of existing learning method is that sentences are ranked without considering topic level information. As we know, a collection of related documents usually covers a few different topics. For example, the specific event "Quebec independence" may involve the topics such as "leader in independence movement", "referendum", "related efforts in independence movement" and so on. It is important to discover the latent topics when summarizing a document collection, because sentences in an important topic would be more important than those talking about trivial topics (Hardy et al, 2002; Harabagiu and Lacatusu, 2005; Otterbacher et al, 2005; Wan and Yang, 2008).

The topic models (Blei et al, 2003) offer a good opportunity for the topic-level information modeling by offering clear and rigorous probabilistic interpretations over other existing clustering techniques. So far, LDA has been widely used in summarization task by discovering topics latent in the document collections (Daume and Marcu, 2006; Haghighi and Vanderwende, 2009; Jin et al, 2010; Mason and Charniak, 2011; Delort and Alfonseca, 2012). However, as far as we know, how to combine topic information and semi-supervised learning into a unified framework has seldom been exploited.

In this paper, inspired by the graph-based semi-supervised strategy and topic models, we propose a two-layer (i.e. sentence layer and topic layer) graph-based semi-supervised learning approach for

*correspondence author

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1197–1207, Dublin, Ireland, August 23-29 2014.

query-focused MDS. By using two revised versions of LDA topic model (See Section 2), our approach naturally models the relations between topics and sentences, and further use these relations to construct the two-layer graph. Experiments on the TAC datasets demonstrate that we can improve summarization performance under the framework of two-layer graph-based semi-supervised learning.

The rest of this paper is organized as follows: Section 2 describes our LDA based topic models, W-LDA and S-LDA. Section 3 presents the construction of the two-layer graph and the semi-supervised learning and the experimental results are provided in Section 4. Then, Section 5 describes related work on query-focused multi-document summarization and topic modeling techniques and we conclude this paper in Section 6.

2 Topic Modeling

2.1 Model Description

As discussed in Section 1, a collection of documents often involves different topics related to a specific event. The basic idea of our summarization approach is to discover the latent topics and cluster sentences according to the topics. Inspired by (Chemudugunta et al, 2006) and (Li et al, 2011), we find 4 types of words in the text: (1) Stop words that occur frequently in the text. (2) Background words that describe the general information about an event, such as "Quebec" and "independence". (3) Aspect words talking about topics across the corpus. (4) Document-specific words that are local to a single document and do not appear across different corpus. Similar ideas can also be found in many LDA based summarization techniques (Haghighi and Vanderwende, 2009; Li et al, 2011; Delort and Alfonseca, 2012).

Stop words can easily be filtered out by a standard list of stopwords. We use a background word distribution ϕ_B to model vocabularies commonly used in the document collection. We assume that there are K aspect topics shared across corpus and each topic is associated with a topic-word distribution $\phi_k,\ k \in [1, K]$. For each document m, there is a document-specific word distribution $\phi_m,\ m \in [K + 1, K + M]$. Each word w is modeled as a mixture of background topics, document-specific topics or aspect topics. We use a latent parameter y_w to denote whether it is a background word, a document-specific word or an aspect word. y_w is sampled from a multinomial distribution with parameter π.

2.2 W-LDA and S-LDA

We describe two models: a word level model W-LDA and a sentence level S-LDA. Their difference only lies in whether the words within a sentence are generated from the same topic.

W-LDA: Figure 1 and Figure 3 show the graphical model and generation process of W-LDA, which is based on Chemudugunta et al's work (2007). Using the Gibbs sampling technique, in each iteration two latent parameters y_w and z_w are sampled simultaneously as follows:

$$P(y_w = 0) \propto \frac{N_{m0,-w} + \gamma}{N_{m,-w} + 3\gamma} \frac{E_B^w + \lambda}{\sum_{w'} E_B^{w'} + V\lambda} \tag{1}$$

$$P(y_w = 1) \propto \frac{N_{m1,-w} + \gamma}{N_{m,-w} + 3\gamma} \frac{E_m^w + \lambda}{\sum_{w'} E_m^{w'} + V\lambda} \tag{2}$$

$$P(y_w = 2, z_w = k) \propto \frac{N_{m2,-w} + \gamma}{N_{m,-w} + 3\gamma} \times \frac{C_m^k + \alpha}{\sum_{k'} C_m^k + K\alpha} \frac{E_k^w + \lambda}{\sum_{w'} E_k^{w'} + V\lambda} \tag{3}$$

where $N_{m0,-w}, N_{m1,-w}$ and $N_{m2,-w}$ denote the number of words assigned to background, document-specific and aspect topic in current document. $N_{m,-w}$ denotes the total number of words in current document. E_B^w, E_m^w and E_k^w are the number of times that word w appears in background topic, document-specific topic and aspect topic k. C_m^k denotes the number of words assigned to topic k in current document.

With one Gibbs sampling, we can make the following estimation:

$$\phi_k^w = \frac{E_k^w + \lambda}{\sum_{w'} E_k^{w'} + V\lambda} \tag{4}$$

Then, the probability that a sentence s is generated from topic k is computed based on the probability that each of its aspect words is generated from topic k:

$$P(s|z_s = k) = \prod_{w \in s, y_w = 2} \phi_k^w \tag{5}$$

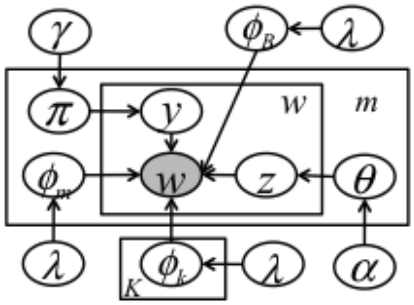


Figure 1: Graphical model for W-LDA

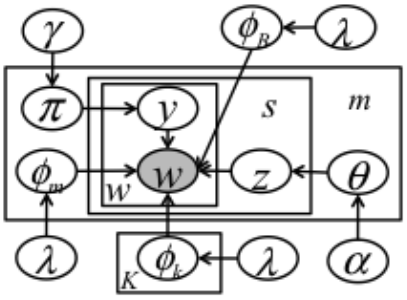


Figure 2: Graphical model for S-LDA

1. Draw background distribution $\phi_B \sim Dir(\lambda)$
2. For each document m:
   draw doc proportion vector $\theta_m \sim Dir(\alpha)$
   draw doc proportion vector $\pi_m \sim Dir(\gamma)$
   draw doc specific distribution $\phi_m \sim Dir(\lambda)$
3. For each topic k:
   draw topic distribution $\phi_k \sim Dir(\lambda)$
4. **For each word w in document m:**
   (a) draw $y_w \sim Multi(\pi_m)$
   (b) if $y_w = 0$: draw $w \sim \phi_B$
   if $y_w = 1$: draw $w \sim \phi_m$
   if $y_w = 2$:
   draw $z_w \sim Multi(\theta_m)$
   $w \sim Multi(\phi_{z_w})$

Figure 3: Generation process for W-LDA

1. Draw background distribution $\phi_B \sim Dir(\lambda)$
2. For each document m:
   draw doc proportion vector $\theta_m \sim Dir(\alpha)$
   draw doc proportion vector $\pi_m \sim Dir(\gamma)$
   draw doc specific distribution $\phi_m \sim Dir(\lambda)$
3. For each topic k:
   draw topic distribution $\phi_k \sim Dir(\lambda)$
4. **For each sentence s in document m:**
   4.1 draw $z_s \sim Multi(\theta_m)$
   4.2 for each word in sentence s:
   (a) draw $y_w \sim Multi(\pi_m)$
   (b) if $y_w = 0$: draw $w \sim \phi_B$
   if $y_w = 1$: draw $w \sim \phi_m$
   if $y_w = 2$: draw $w \sim Multi(\phi_{z_w})$

Figure 4: Generation process of S-LDA

S-LDA: In S-LDA, each sentence is treated as a whole and words within a sentence are generated from the same topic (Gruber et al., 2007). Its graphical model and generated process are shown in Figure 2 and Figure 4. In S-LDA, we firstly sample the topic z_s for each sentence as follows:

$$\begin{aligned} P(z_s = k|z_{-s}, y, w) &\propto \frac{\Gamma(\sum_{w'} E_k^{w'} + V\lambda)}{\Gamma(\sum_{w'} E_k^{w'} + N_s^A + V\lambda)} \\ &\times \prod_{w \in s, y_w = 2} \frac{\Gamma(E_k^w + N_s^w + \lambda)}{\Gamma(E_k^w + \lambda)} \cdot \frac{C_m^k + \alpha}{\sum_{k'} C_m^{k'} + K\alpha} \end{aligned} \tag{6}$$

C_m^k denotes the number of sentences in document m assigned to topic k. N_s^A denotes the number of aspect words in current sentence. Then y_w is sampled.

In our experiments, we set hyperparameters $\alpha = 1,\ \beta = 0.5,\ \lambda = 0.01$. We run 500 burn-in iterations through all documents in the collection to stabilize the distribution of z and y before collecting samples.

3 Graph-based Semi-supervised Learning

As stated before, the consideration of higher level information (i.e. topics) would be helpful for sentence ranking in summarization. In our two-layer graph, the upper layer is composed of topic nodes and the lower layer is composed of sentences nodes, among which there is one node representing the query.

Formally, given a document set D, let $G =< V_s, V_t, E >$ be the two-layer graph, where $V_s = \{s_1, s_2, ..., s_N\}$ denotes the set of all the sentence nodes and s_1 is the query. $V_t = \{z_1, z_2, ..., z_K\}$ corresponds to all the topic nodes. The collection of edges E in the graph consists of the relations within layers and between layers. And the edge weights are measured according to the similarities between nodes, which are computed based on the topic distribution from our two topic model extensions. Specifically, we introduce four edge weight matrices $\hat{W}_{N*K}$, $\bar{W}_{K*N}$, U and P to describe the sentence-to-topic relations, the topic-to-sentence relations, the sentence-to-sentence relations and the topic-to-topic relations respectively.

Firstly, the row-normalized edge weight matrices $\hat{W}_{N*K}$ and $\bar{W}_{K*N}$ denotes the similarity matrix between sentences and topics,

$$\hat{W}_{i,j} = \frac{sim(s_i, z_k)}{\sum_{k'} sim(s_i, z_{k'})} \quad \bar{W}_{i,j} = \frac{sim(s_i, z_k)}{\sum_j sim(s_j, z_k)} \tag{7}$$

where $sim(s_i, z_k) = p(s_i|z_{s_i} = z_k)$ is the probability that the sentence is generated from that topic calculated in Equation (5).

The edge weight matrix U describe the sentence-to-sentence relations. In the same way, the similarity between two sentences is the cosine similarity between their topic distributions, $sim(s_i, s_j) = \frac{1}{C_1} \sum_k p(s_i|z_{s_i} = k) \cdot p(s_j|z_{s_j} = z_k)$, where $C_1 = \sqrt{\sum_k p^2(s_i|z_{s_i} = k)} \sqrt{\sum_k p^2(s_j|z_{s_j} = k)}$ is the normalized factor. Since the row-normalization process will make the sentence-to-sentence relation matrix asymmetric, we adopt the following strategy: let $Sim(s)$ denote the similarity matrix between sentences, where $Sim(s)(i, j) = sim(s_i, s_j)$ and D denotes the diagonal matrix with (i, i)-element equal to the sum of the i^{th} row of $Sim(s)$. Edge weight matrix between sentences U is calculated as follows:

$$U = D^{-\frac{1}{2}} Sim(s) D^{-\frac{1}{2}} \tag{8}$$

Then, the edge weight matrix between topics P is the normalized symmetric matrix of the similairty matrix between two topics. The cosine similarity between two topics is calculated according to word-topic distribution.

$$sim(z_i, z_j) = \frac{1}{C_2} \sum_w p(w|z_i) p(w|z_j) = \frac{1}{C_2} \sum_w \phi_{z_i}^w \phi_{z_j}^w \tag{9}$$

where $C_2 = \sqrt{\sum_w p^2(w|z_i)} \cdot \sqrt{\sum_w p^2(w|z_j)}$ is the normalized factor.

We further transform the task to an optimizing problem based on the assumption that closely related nodes (sentences and topics) tend to have similar scores. So we would give more penalty for the difference between closely related nodes with regard to edge weight matrices $\hat{W}_{N*K}$, $\bar{W}_{K*N}$, U and P. This motivates the following optimization function $\Omega(f, g)$ in Equation (10) similar to the graph harmonic function(Zhu et al, 2003). f denotes the sentence score vector and g denotes the topic score vector. Intuitively, $\Omega(f, g)$ measures the sum of difference between graph nodes; the more they differ, the larger $\Omega(f, g)$ would be.

$$\begin{aligned} \Omega(f, g) = & \, a \sum_{0 \le i,j \le N} U_{i,j} (f_i - f_j)^2 + a \sum_{0 \le i,j \le K} P_{i,j} (g_i - g_j)^2 \\ & + (1 - a) \sum_{0 \le i \le N} \sum_{0 \le j \le K} \hat{W}_{ij} (f_i - g_j)^2 \\ & + (1 - a) \sum_{0 \le i \le N} \sum_{0 \le j \le K} \bar{W}_{ij} (g_i - f_j)^2 \end{aligned} \tag{10}$$

The score vectors can be achieved by minimizing the function in Equation (10). That is, (f, g)=argmin$_{f,g} \Omega(f, g)$. We can get the following equations (details are shown in Appendix).

$$\begin{aligned} f &= aUf + \frac{1}{2}(1 - a)(\hat{W} + \bar{W}^T) g \\ g &= aPg + \frac{1}{2}(1 - a)(\hat{W}^T + \bar{W}) f \end{aligned} \tag{11}$$

Equation (11) conforms to our intuition: (1) A sentence would be important if it is heavily connected with many important sentences and a topic would be important if it is closely related to other important topics. (2) A sentence would be important if it is expressing an important topic, and in turn a topic would be important if it is referred by an important sentence. Based on Equation (11), the ranking algorithm is designed in a semi-supervised way, where the score of the labeled query is fixed to the largest score of 1 during each iteration, as shown in Figure 5. Then, our algorithm iteratively calculates the score of topics and sentences until convergence[1].

Input: The sentence set $\{s_1, s_2, ..., s_N\}$, topic set $\{z_1, z_2, ..., z_K\}$, edge weight matrix $\hat{W}$, $\bar{W}$, U and P. s_1 is the query.
Output: Sentence score vector f and topic score vector g.
BEGIN
1. Initialization, k=0:
$f^0 = (1, 0, 0, ..., 0)^T$, $g^0 = (0, 0, ..., 0)^T$
2. Update sentence score vector
$f^{k+1} = aUf^k + \frac{1}{2}(1-a)(\hat{W} + \bar{W}^T)g^k$
3. Update topic score vector
$g^{k+1} = aPg^k + \frac{1}{2}(1-a)(\hat{W}^T + \bar{W})f^k$
4. fix the score of query in f^{k+1} to 1.
5. k=k+1 Go to Step 2 until convergence.
END

Figure 5: Sentence Ranking Algorithm

Input: The sentence set $S = \{s_1, s_2, ..., s_N\}$, sentence score vector f
Output: Summary Y.
BEGIN:
1. Initialization: $Y = \Phi$, $X = \{S - s_1\}$.
2. while word num is less than 100:
(a) $s_m = \arg\max_{s_i \in X} f(s_i)$
(b) If $sim(s_m, s) < Th_{sem}$, for all $s \in Y$:
$Y = Y + \{s_m\}$
(c) $X = x - \{s_m\}$
END

Figure 6: Sentence Selection Algorithm

3.1 Summary Generation

Sentence compression can largely improve summarization quality (Zajic et al, 2007; Peng et al, 2011). Since sentence compression is not the main task in this paper, we just use the revised sentence compression techniques in (Li et al, 2011).Here, we remove the redundant modifiers such as adverbials, relative clause modifiers, abbreviations, participials and infinitive modifiers for each sentence.

As for the sentence selection process, sentences with higher ranking score are selected into the summary. Then Maximum Marginal Relevance (MMR)(Goldstein et al, 1999) is further used for redundancy removal. We just apply a simple greedy algorithm for sentence selection as shown in Figure 6. We use Y to denote the summary set which contains the selected summary sentences. The algorithm first initializes Y to Φ and X as the set $\{S - s_1\}$. During each iteration, we select the highest ranked sentence s_j from the sentence set X. We need to assure that the value of semantic similarity between two sentences is less than Th_{sem}. Th_{sem} denotes the threshold for the cosine similarity between two sentences and is set to 0.5 in our model.

4 Experiments

The query-focused MDS task defined in TAC (Text Analysis Conference) evaluations requires generating a concise and well organized summary for a collection of related documents according to a given query. The query usually consists of a narrative/question sentence. Our experiment data is composed of TAC (2008-2009) data[2], which contain 48 and 44 document collections respectively. We use docset-A data sets in TAC which has 10 documents per collection. The average numbers of sentences per document in TAC2008 and TAC2009 are 252 and 243 respectively, and the system-generated summary is limited to 100 words. It is noted that the corpus of TAC2008 and TAC2009 are similar. In our experiment, we apply the optimal topic number trained on TAC2008 dataset to TAC2009 dataset.

[1]In our experiments, if $|f_i^k - f_1^{k+1}| \leq 0.0001 (1 \leq i \leq N)$ and $|g_i^k - g_1^{k+1}| \leq 0.0001 (1 \leq i \leq T)$, iteration stops.

[2]TAC data sets are for the update summarization tasks, where the summarization for docset-A can be seen the query-focused summarization task referred in this paper.

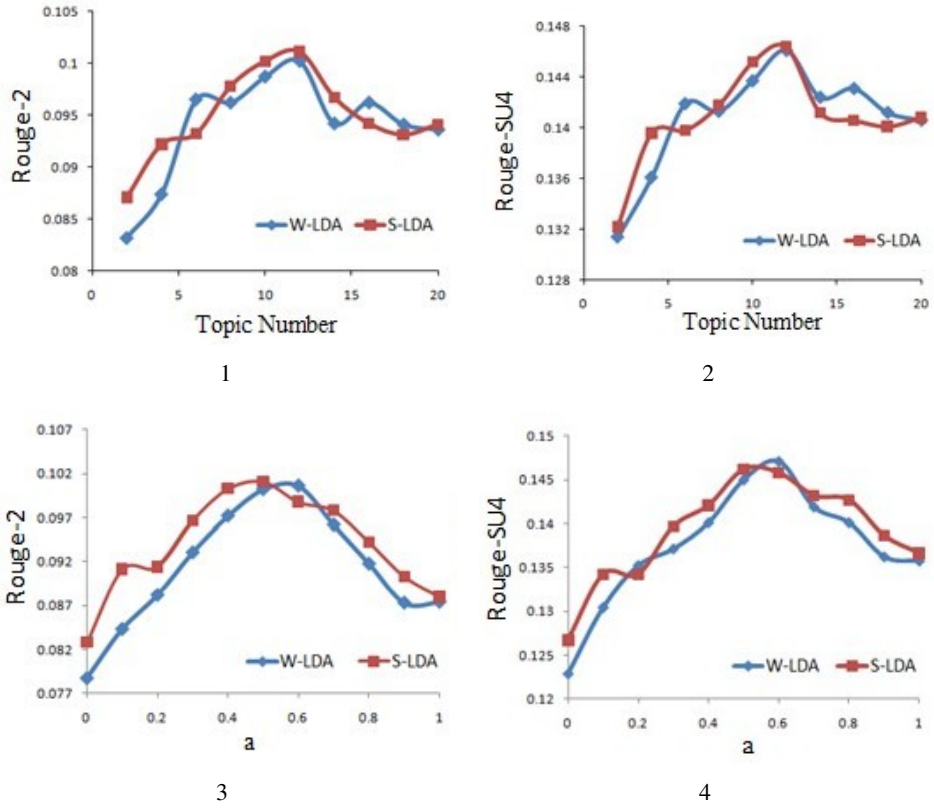


Figure 7: ROUGE score via (1)(2) topic number and (3)(4) parameter a on TAC2008.

As for evaluation metrics, we use ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin, 2004) measures. ROUGE measures summary quality by counting overlapping units such as the n-gram, word sequences and word pairs between the candidate summary and the reference summary. We report ROUGE-1, ROUGE-2, and ROUGE-SU4[3] scores and their corresponding 95% confidential intervals, to evaluate the performance of the system-generated summaries. As a preprocessing step, stopwords are firstly removed with a list of 598 stop words and the remaining words are then stemmed using PorterStemmer.[4].

4.1 Parameter Tuning

There are two parameters to tune in our model. The first parameter is a in Equation (11) that controls the tradeoff between influence from topics and from sentences. The second one is the topic number K in LDA topic model. The combination of the two factors makes it hard to find a global optimized solution. So we apply a gradient search strategy. At first, parameter a is fixed to a given value. Then the performance of using different topic numbers is evaluated. After that, we fix the topic number to the value which has achieved the best performance, and conduct experiments to find an appropriate value for a. Here, we use TAC2008 as training data and test our model on TAC2009.

First, a is set to 0.5, then we change topic number K from 2 to 20 at the interval of 2. The ROUGE score reaches their peaks when the topic number is around 12, as shown in Figure 7(1) and Figure 7(2). Then we fix the number of K to 12 and change the value of parameter a from 0 to 1 with the interval of 0.1. When the value of a is set to 0, the model degenerates into a one-layer graph ranking algorithm where topic clustering information is neglected. As we can see from Figure 7(3) and Figure 7(4) , the ROUGE scores reach their peaks around 0.6 and then drop afterwards. Thus, the topic number is set to 12 and a is set to 0.6 in the test dataset.

[3]Jackknife scoring for ROUGE is used in order to compare with the human summaries.
[4]http://tartarus.org/martin/PorterStemmer/

4.2 Baseline Comparison

We firstly compare W-LDA and S-LDA with other clustering approaches. To be fair, we use the identical sentence compression techniques and preprocessing methods for all baselines. Summaries are truncated to the same length of 100 words.

Standard-LDA: A simplified version of W-LDA without considering the background or document-specific information.

K-means: Using the K-means clustering algorithm for graph construction. We firstly randomly select K sentences as initial centroid for clusters and then iteratively assign a sentence to each cluster. The centroid is recomputed until convergence. The similarity between nodes in the graph (sentence or cluster) is computed using the standard cosine measure based on the tf-idf information. K is set to 12, the same as topic number in LDA.

Agglomerative: a bottom-up hierarchical clustering algorithm and starts with the sentences as individual clusters and, at each step, merges the most similar or closest pair of clusters, until the number of the clusters reduces to the desired number $K = 12$.

Divisive: a top-down hierarchical clustering algorithm and starts with one, all-inclusive cluster and, at each step, splits the largest cluster until the number of clusters increases to the desired number K, $K = 12$.

Approach	Rouge-1	Rouge-2	Rouge-SU4
W-LDA	0.3791 (0.3702-0.3880)	0.1092 (0.1047-0.1135)	0.1382 (0.1350-0.1414)
S-LDA	**0.3802 (0.3721-0.3883)**	**0.1109 (0.1061-0.1157)**	**0.1398 (0.1342-0.1454)**
Standard LDA	0.3702 (0.3614-0.3790)	0.1012 (0.0960-0.1064)	0.1292 (0.1242-0.1344)
K-means	0.3658 (0.3582-0.3734)	0.1046 (0.0992-0.1080)	0.1327 (0.1263-0.1391)
Agglomerative	0.3681 (0.3612-0.3750)	0.1042 (0.091-0.1093)	0.1319 (0.1266-0.1272)
Divisive	0.3676 (0.3610-0.3742)	0.1021 (0.0981-0.1061)	0.1320 (0.1275-0.1365)

Table 1: Comparison with other clustering baselines.

Table 1 presents the performance of different clustering algorithms for summarization. Traditional clustering algorithms such as K-means, Agglomerative and Divisive clustering achieve comparative results. Compared with traditional clustering algorithms, LDA based models (W-LDA, S-LDA, Standard-LDA) achieve better results. This can be explained by the clear and rigorous probabilistic interpretation of topic models. Background information and document-specific information would influence the performance of topic modeling (Chemudugunta et al, 2006), that is why S-LDA and W-LDA achieve better ROUGE performance than the standard LDA. We can also see that S-LDA is slightly better than W-LDA in regard with ROUGE performance. The reason can be explained as follows: The aim of topic modeling in this task is to cluster sentences according to their topics. So treating sentence as a unit in topic modeling would be better than treating it as a set of independent words. In addition, forcing the words in one sentence to share the same aspect topic can ensure semantic cohesion of the mined topics.

Next, we compare our model with the following widely used summarization approaches.

Manifold: One-layer graph-based semi-supervised approach developed by Wan et al.(2008). Sentence relations are calculated according to $tf - idf$ and topic information is neglected.

LexRank: An unsupervised graph-based summarization approach(Erkan and Radev, 2004), which is a revised version of the famous web ranking algorithm PageRank.

KL-Divergence: The approach developed by (Lin et al, 2006) by using a KL-divergence based sentence selection strategy.

$$KL(P_s||Q_d) = \sum_w P(w) log \frac{P(w)}{Q(w)} \tag{12}$$

where P_s is the unigram distribution of candidate summary and Q_d denotes the unigram distribution of document collection. Since this approach is designed for general summarization, query influence is not considered.

Hiersum: A LDA based approach proposed by (Haghighi and Vanderwende, 2009), where unigram distribution is calculated from LDA topic model in Equation (12).

MEAD: A centroid based summary algorithm by (Radev et al, 2004). Cluster centroids in MEAD consists of words which are central not only to one article in a cluster, but to all the articles. Similarity is measured by using $tf - idf$.

Approach	Rouge-1	Rouge-2	Rouge-SU4
W-LDA	0.3891 (0.3802-0.3980)	0.1192 (0.1147-0.1235)	0.1482 (0.1450-0.1514)
S-LDA	**0.3902 (0.3821-0.3983)**	**0.1209 (0.1161-0.1257)**	**0.1498 (0.1442-0.1554)**
Manifold	0.3581 (0.3508-0.3656)	0.1007 (0.0952-0.1062)	0.1267 (0.1214-0.1320)
LexRank	0.3442 (0.3381-0.3502)	0.0817 (0.0782-0.0852)	0.1106 (0.1064-0.1148)
KL-divergence	0.3468 (0.3410-0.3526)	0.0820 (0.0782-0.0858)	0.1117 (0.1073-0.1161)
Hiersum	0.3599 (0.3526-0.3672)	0.1004 (0.0956-0.1052)	0.1280 (0.1221-0.1339)
MEAD	0.3451 (0.3390-0.3512)	0.0862 (0.0817-0.0907)	0.1131 (0.1080-0.1182)

Table 2: Performance comparison with baselines

Performance is presented at Table 2. We can find that ROUGE performance of one-layer graph ranking algorithms such as Manifold and LexRank, where topic information is neglected, achieve worse results than all two-layer models where topic information is considered (See Table 1). This verifies our previous claim (Hardy et at., 2002; Harabagiu and Lacatusu, 2005; Wan and Yang, 2008) that the consideration of topic information will improve summarization performance. S-LDA and W-LDA achieve better performance than KL-divergence and Hiersum. This is because the sentence selection strategy for KL-divergence and Hiersum tries to select sentence best representing the document as shown in Equation (12), but do not consider the influence of query.

4.3 Manual Evaluation

W-LDA and S-LDA get comparative ROUGE scores. To obtain a more accurate measure to decide which approach is better, we perform a simple user study concerning the following aspects on 40 randomly selected topics in TAC2009: (1) Overall quality. (2) Focus: Whether the summary contains less irrelevant content? (3) Responsiveness: Whether the summary is responsive to the query. (4) Non-Redundancy: Whether the summary is non-redundant. Each respect is rated from 1 (very poor) to 5 (very good). Four native speakers who are Ph.D. students in computer science (none are authors) performed the task.

The average score and standard deviation for W-LDA and S-LDA are displayed in Table 3. We can see that the two models almost tie in foucs and non-redundancy. This is because two models use the same sentence selection strategy based on MMR for redundancy removal and propagation model to impose the query's influence on sentences. S-LDA outperforms W-LDA in overall ranking and responsiveness ranking. This implies that treating sentence as a unit in topic modeling would be preferable to just treating it as a series of independent words.

	S-LDA	W-LDA
Overall	3.98 ± 0.52	3.58 ± 0.55
Focus	3.65 ± 0.54	3.35 ± 0.61
Responsiveness	3.73 ± 0.43	3.38 ± 0.46
Non-Redundancy	3.48 ± 0.51	3.45 ± 0.48

Table 3: Manual evaluation for S-LDA and W-LDA.

5 Related Work

Graph-based ranking approaches have been hot these days for both generic and query-focused summarization (Zhou et al, 2003; Zhou et al, 2004; Erkan and Radev, 2004; Wan et al, 2007; Wei et al, 2008). Commonly used graph-based ranking algorithms are mainly inspired by the link analysis algorithm in web research such as PageRank (Page et al, 1999). (Wan et al, 2007) proposed the approach that treated

the task of query-focused MDS as a semi-supervised learning task, in which the query is treated as a labeled node, and sentences as unlabeled nodes. Then the scores of sentences are determined from the manifold learning algorithm proposed by (Zhou et al, 2003) or the harmonic approach proposed by (Zhu et al, 2003).

It is worthy of noting that researchers have found that by considering topic level information, the summarization performance can be effectively improved (Hardy et al, 2002; Wan and Yang, 2008; Harabagiu and Lacatusu, 2005). For example, (Otterbacher et al, 2005) models documents as a stochastic graph and calculates sentence ranking scores with a topic-sensitive version of PageRank. (Wan and Yang, 2008) developed a two-layer graph by clustering sentences by using standard clustering algorithms such as K-means or agglomerate clustering. However, his algorithm is for general summarization where the influence of query is not considered.

A significant portion of recent work incorporates LDA topic models (Blei et al, 2008) in summarization tasks for their clear and rigorous probabilistic topic interpretations (Daume and Marcu, 2006; Titov and McDonald, 2008; Haghighi and Vanderwende, 2009; Mason and Charniak, 2011; Li et al, 2013a; Li et al, 2013b). (Haghighi and Vanderwende, 2009) introduced a LDA based model called Hiersum to find the subtopics or aspects by combining KL-divergence criterion for selecting relevant sentences. AYESSUM (Daume and Marcu, 2006) and the Special Words and Background model (Chemudugunta et al, 2006) are very similar to Hiersum. In the same way, (Delort and Alfonseca, 2012) tried to use LDA to model different levels of information for novelty detection in update summarization. Furthermore, (Paul and Dredze, 2013) extends their f-LDA to jointly model combinations of drug, aspect and route of administration as an exploratory tool for extractive summarization.

6 Conclusions and Future Work

In this paper, we propose a two-layer graph-based semi-supervised algorithm for query-focused MDS. Topic modeling techniques are used for sentence clustering and further graph construction. By considering different kinds of information such as background or document-specific information, our two LDA topic model extensions achieve better results than traditional clustering algorithms.

One primary disadvantage of our models is that it is hard to decide the topic number K in LDA models and how to define topic number is still a open problem in LDA topic models. From Figure 7, we can see that summarization performance is sensitive to topic number. We train the value of topic number on TAC2008 dataset and test the model on TAC2009. Such process makes sense because the corpus sizes and contents of two datasets are similar. But it would be hard to extend optimal topic number in TAC2008 to other datasets. Using non-parametric topic modeling techniques where topic number does not have to be predefined is one of our future works.

Acknowledgements

Thanks Jiwei Li for the insightful reviews and careful polishment. We also thank the three anonymous reviewers for their helpful comments. This work was partially supported by National High Technology Research and Development Program of China (No. 2012AA011101), National Key Basic Research Program of China (No. 2014CB340504), National Natural Science Foundation of China (No. 61273278), and National Key Technology R&D Program (No: 2011BAH10B04-03).

References

Edoardo M. Airoldi, Blei D M, Fienberg S E, et al. Mixed membership stochastic blockmodels[J]. In *The Journal of Machine Learning Research*, 2008, 9(1981-2014): 3.

David Blei, Andrew Ng and Micheal Jordan. 2003. Latent dirichlet allocation. In *The Journal of Machine Learning Research*.

Chaltanya Chemudugunta, Padhraic Smyth and Mark Steyers. Modeling General and Specific Aspects of Documents with a Probabilistic Topic Model.. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference.*

Hal Daume and Daniel Marcu H. 2006. Bayesian Query-Focused Summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 305-312.

Jean-Yves Delort and Enrique Alfonseca. DualSum: a topic-model based approach for update summarization. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics.*

Gune Erkan and Dragomir Radev. 2004. Lexrank: graph-based lexical centrality as salience in text summarization. In *Journal of Artificial Intelligence Research*.

Jade Goldstein, Mark Kantrowitz, Vibhu Mittal and Jaime Carbonell. 1999. Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*.

Amit Gruber, Yair Weiss and Michal Rosen-Zvi. Hidden topic Markov models. In *International Conference on Artificial Intelligence and Statistics.* 2007

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362370.

Sanda Harabagiu and Finley Lacatusu. 2005. Topic themes for multi-document summarization. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*.

Hilda Hardy, Nobuyuki Shimizu, Tomek Strzakowski, Liu Ting, Xinyang Zhang and Bowden Wize. 2002. Cross-document summarization by concept classification. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*.

Feng Jin, Minlie Huang, and Xiaoyan Zhu. 2010. The summarization systems at tac 2010. In *Proceedings of the third Text Analysis Conference*, TAC-2010.

Jiwei Li and Sujian Li. 2013. Evolutionary Hierarchical Dirichlet Process for Timeline Summarization. In *ACL* 2013.

Jiwei Li and Claire Cardie. 2014. Timeline Generation: Tracking individuals on Twitter. In *WWW* 2014.

Peng Li, Yinglin Wang, Wei Gao and Jiang Jing. 2011. Generating aspect-oriented multi-document summarization with event-aspect model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.*

Chin-Yew Lin, Guihong Cao, Jianfeng Gao, and Jian-Yun Nie. 2006. An information-theoretic approach to automatic evaluation of summaries. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics.*

Chin-Yew Lin. Improving summarization performance by sentence compression: a pilot study. In *Proceedings the sixth international workshop on Information retrieval with Asian languages*.

Rebecca Mason and Eugene Charniak. 2011. Extractive multi-document summaries should explicitly not contain document-specific content. In *proceedings of ACL HLT*.

Jahna Otterbacher, Gne Erkan, and Dragomir R. Radev. Using random walks for question-focused sentence retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2005.

Lawrence Page, Sergey Brin, Rajeev Motwani and Terry Winograd. 1999. The Pagerank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Libraries.

Michael J. Paul and Mark Dredze. Drug extraction from the web: Summarizing drug experiences with multi-dimensional topic models. In *Proceedings of NAACL-HLT.* 2013.

Wei-Ting Peng, Wei-Ta Chu, Chia-Han Chang, et al. Editing by viewing: automatic home video summarization by viewing behavior analysis[J]. In *Multimedia, IEEE Transactions on*, 2011, 13(3): 539-550.

Dragomir Radev, Allison T, Blair-Goldensohn S, et al. MEAD-a platform for multidocument multilingual text summarization[C]. In *Proceedings of the 4th International Conference on Language Resources and Evaluation,* 2004.

Ivan Titov and Ryan McDonald. 2008. Modeling on- line reviews with multi-grain topic models. In *International World Wide Web Conference*.

Xiaojun Wan, Jianwu Yang and Jianguo Xiao. 2007. Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of International Joint Conference on Artificial Intelligence*.

Xiaojun Wan and Jianwu Yang. 2008. Multi-document Summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*.

Furu Wei, Wenjie Li, Qin Lu, and Yanxiang He. 2008. Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*.

David Zajic, et al. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. In *Information Processing & Management* 43.6 (2007): 1549-1570.

Dengzhong Zhou, Jason Weston, Arthur Gretton, Olivier Bousquet and Bernhard Schlkopf. 2003. Ranking on Data Manifolds. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*.

Dengyou Zhou, Olivier Bousquet, Thomas Navin and JasonWeston. 2004. Learning with Local and Global Consistency. In *Proceedings of Advances in neural information processing systems*.

Xiaojin Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and functions. In *Proceedings of the 20th International Joint Conference on Machine Learning*, 2003.

APPENDIX

To optimize $\Omega(f, g)$, shown in Equation (10), we set the partial derivative with respect to f_m to 0, for $m \in [1, N]$. Let δ_{mn} denote the index function as follows:

$$\delta_{mn} = \begin{cases} 1 & if\ m = n \\ 0 & if\ m \neq n \end{cases}$$

$$\begin{aligned}
0 &= \frac{\partial \Omega(f,g)}{f_t} \\
&= 2a \sum_{i,j} U_{i,j}(f_i - f_j)(\delta_{it} - \delta_{jt}) + 2(1-a) \\
&\times \sum_{i,j} \hat{W}_{ij}(f_i - g_j)\delta_{it} - 2(1-a) \sum_{i,j} \bar{W}_{ij}(g_i - f_j)\delta_{jt} \\
&= 2(1-a) \sum_j \hat{W}_{tj}(f_t - g_j) + 2(1-a) \sum_i \bar{W}_{it}(g_i - f_t) \\
&+ 2a \sum_j U_{tj}(f_t - f_j) + 2a \sum_i U_{it}(f_i - f_t) \\
&= f_t[4a \sum_j U_{tj} + 2(1-a) \sum_j \hat{W}_{tj} + 2(1-a) \sum_j \bar{W}_{jt}] \\
&- 4a \sum_j U_{tj} f_j - 2(1-a) \sum_j \hat{W}_{tj} g_j - 2(1-a) \sum_j \bar{W}_{jt} g_j \\
&\sum_j U_{tj} = 1 \qquad \sum_j \hat{W}_{tj} = 1 \qquad \sum_j \bar{W}_{jt} = 1 \\
f_t &= a \sum_j U_{tj} f_j + \frac{1}{2}(1-a)[\sum_j (\hat{W}_{tj} + \bar{W}_{jt}) g_j]
\end{aligned}$$

So we have:

$$f = aUf + \frac{1}{2}(1-a)(\hat{W} + \bar{W}^T)g$$

A similar approach is used to obtain the second part of Equation (11).

Ranking Multidocument Event Descriptions for Building Thematic Timelines

Kiem-Hieu Nguyen[†], **Xavier Tannier**[†$], **and Veronique Moriceau**[†$]
[†]LIMSI-CNRS
[$]Univ. Paris-Sud
Orsay, France
{nguyen,xtannier,moriceau}@limsi.fr

Abstract

This paper tackles the problem of timeline generation from traditional news sources. Our system builds thematic timelines for a general-domain topic defined by a user query. The system selects and ranks events relevant to the input query. Each event is represented by a one-sentence description in the output timeline.

We present an inter-cluster ranking algorithm that takes events from multiple clusters as input and that selects the most salient and relevant events. A cluster, in our work, contains all the events happening in a specific date. Our algorithm utilizes the temporal information derived from a large collection of extensively temporal analyzed texts. Such temporal information is combined with textual contents into an event scoring model in order to rank events based on their salience and query-relevance.

1 Introduction

We aim at building thematic timelines from multiple documents relevant to a specific, user-generated query. For instance, for the query "*Libya conflict*", our system will return important events related to the Libya conflict in 2011 involving Kadhafi forces, rebels, NATO intervention, etc. (Figure 1). Such a timeline can then be visualized as a textual, event-based summary, or through any existing graphical timeline visualization tool.

The main contribution of this paper is a two-step inter-cluster ranking algorithm aimed at selecting salient and non-redundant events from temporal clusters, which are sets of sentences describing events related to the query and that occurred at the same day. In the first step, a scoring model is proposed to rank sentences describing events, according to their relevance and salience to the topic. In the second step, the ranked events are iteratively reranked based on their content in order to reduce information redundancy. We finally obtain an extendable, chronological summary of important events concerning the query.

This paper is organized as follows: §2 introduces related work. §3 presents the resources used and gives an overview of the system. The salient date algorithm proposed by Kessler et al. (2012), that we used to build our temporal clusters, is briefly summarized in §4. §5 and §6 describe our ranking approach to event selection and a content-based reranking algorithm, respectively. The evaluations are presented in §7. §8 is dedicated to the conclusion and future work.

2 Related Work

Our work is closely related to event detection and tracking (EDT) and multidocument summarization (MDS). This section introduces some important work in these fields.

2.1 Event Detection and Tracking

EDT on news streams has been intensively studied. Early work concentrates on detecting events from article texts using vector-based techniques (Allan et al., 1998; Petrović et al., 2010) or graphical models

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1208–1217, Dublin, Ireland, August 23-29 2014.

Mar 19 2011. *The UN Security Council agrees a resolution authorising "all necessary measures" to protect civilians in Libya, where a revolt is under way against the regime of Moamer Kadhafi.*
Mar 19 2011. *French, US and British forces attack Kadhafi's forces from the air.*
Mar 19 2011. *They retreat from the rebel stronghold of Benghazi.*
Mar 26 2011. *Benefiting from the Western air strikes, rebels take the towns of Ajdabiya and Brega, moving on to the oil town of Ras Lanuf.*
Mar 29 2011. *International powers meet in London but fail to agree on a strategy against Kadhafi.*
Mar 30 2011. *Kadhafi's forces retake Ras Lanuf and Brega.*
Mar 30 2011. *Libyan foreign minister Mussa Kussa defects to Britain.*
Mar 31 2011. *NATO takes full command of the coalition campaign.*
Apr 01 2011. *In the first of several "friendly fire" incidents, NATO attacks kill nine rebels and civilians.*
Apr 06 2011. *Washington rejects a letter from Kadhafi calling for an end to air strikes, and repeats that Kadhafi must go.*
Apr 07 2011. *A world food program aid ship arrives at rebel-held Misrata, where shelling by Kadhafi's forces has killed or wounded hundreds.*
Apr 10 2011. *An African Union delegation headed by South African president Jacob Zuma meets Kadhafi and the rebels.*
Apr 10 2011. *The former accepts their peace plan, but the latter refuse, saying Kadhafi and his sons must step down.*
Apr 12 2011. *Britain and France call on their NATO allies to step up operations against Kadhafi's forces.*
...

Figure 1: A chronology about "*Libya conflict*" written by journalists.

(Sayyadi et al., 2009). These papers do not consider time, which is an essential dimension of event timelines.

Attempts to use temporal information for EDT are significant in the literature. To name but a few, Alonso et al. (2009) apply time-based clustering on search results. Yan et al. (2011) use document timestamps to calculate temporal proximity for timeline generation from web documents. Similarly, Zhao et al. (2007) use text similarity and time intensity for event clustering on social streams. Kessler et al. (2012) exploit temporal analysis to detect salient dates of an event from raw text. Following this direction, Battistelli et al. (2013) apply sequential pattern mining to select a one-sentence description for each salient date of an event.

2.2 Multidocument Summarization

Sentence extraction is essential in extractive text summarization. In the unsupervised approach, sentences are scored using term weight and term proximity induced from a document collection (Goldstein et al., 2000). In the supervised approach, training data generated from reference summaries are used to learn classification or ranking models. New sentences are selected based on their confidence value on learned models (Wan et al., 2007). As information comes from documents on the same topic, it should be noticed that it is also important to reduce redundancy in MDS (Carbonell and Goldstein, 1998).

Filippova (2010) builds a co-occurrence word graph from a collection of related sentences and generates a generic summary from the graph based on shortest path finding. Her algorithm is a hybrid method between extractive and abstractive approaches to MDS.

3 Resources and System Overview

3.1 Corpus and Chronologies

For this work, we use a corpus of newswire texts provided by the AFP French news agency. The English AFP corpus is composed of 1.3 million texts that span the 2004-2011 period (511 documents/day in average and 426 millions words). Each document is an XML file containing title, document creation time (DCT), set of keywords, and textual content split into paragraphs.

AFP "chronologies" (textual event timelines) are a specific type of articles written by AFP journalists in order to contextualize current events. These chronologies may concern any topic discussed in the media, and consist in a list of dates (typically between 10 and 20) associated with a text describing the related event(s). Figure 1 shows an example of such a chronology. Note that several important events can occur at the same date.

3.2 System Overview

Figure 2 shows the general architecture of the system. When the user submits a query, sentences are retrieved by the Lucene search engine and are clustered by the dates appearing in those sentences (step ①

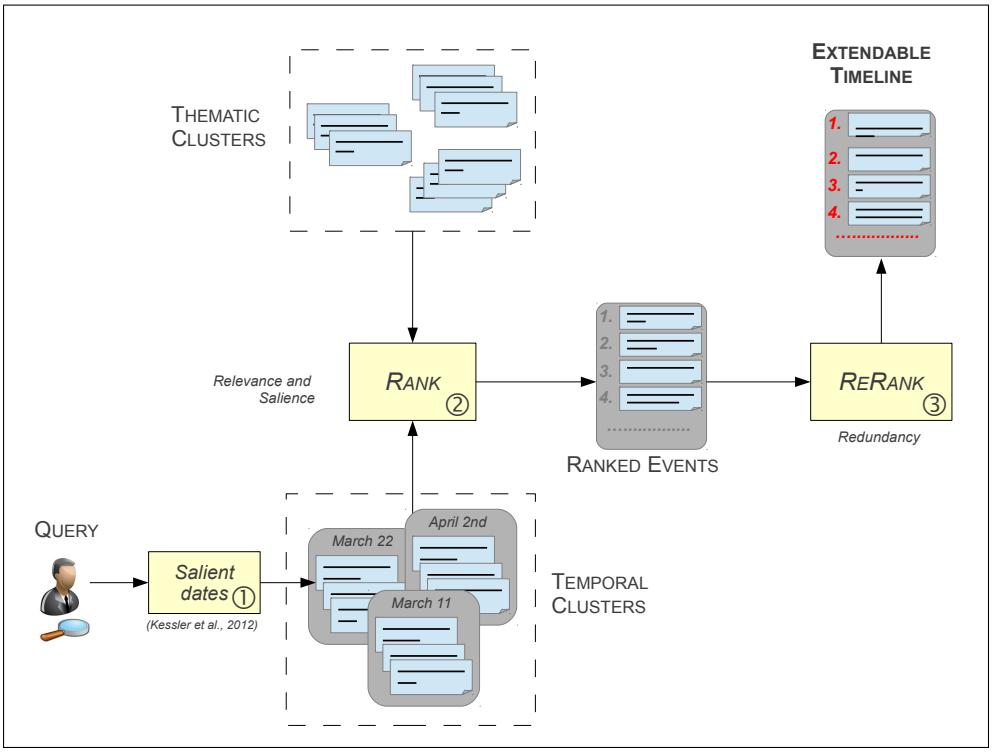


Figure 2: System overview.

in the Figure (Kessler et al., 2012)).

Then, all sentences are ranked by the relevance and salience of described events. This is done by modeling event relevance and salience as a scoring function (step ②). *Thematic* clusters are created by applying clustering on the set of events on the same date. Finally, sentences are reranked by an iterative algorithm aiming at reducing redundancy from the initial list (step ③) to achieve an extendable timeline.

4 Temporal Clusters

As stated in the introduction, our main contribution in this paper is to rank and select salient and non-redundant sentences from clusters, in order to build query-based timelines. We rely on the algorithm proposed by Kessler et al. (2012) for building temporal clusters. This section is a quick overview of their approach.

4.1 Preprocessing

A temporal analysis is performed on all documents from the AFP corpus (see §3.1) with the Heideltime (Strötgen and Gertz, 2013) parser. The main purpose is to collect as much temporal information as possible. Absolute dates and DCT-relative dates are extracted and normalized (full dates represented in a common format). DCT-relative dates are those which are relative to the date on which the document is published, such as "*Yesterday*" (day before DCT), "*next Friday*" (first Friday following the DCT) or "*on Friday*" (can be first Friday preceding or following the DCT, depending on the tense of the verb that governs the temporal expression).

In a corpus containing 426 millions words, 845,000 absolute dates and 4.6 millions relative dates were detected and normalized.

4.2 Temporal Cluster Building

At query time, temporal clusters (or "salient date sets") are then built with the help of a search engine (Lucene in that case[1]). Articles are indexed by Lucene at sentence-level (a document = a sentence).

[1] `http://lucene.apache.org`

...launched the first air strike on	March 19	*, has deployed around 20 Rafale and Mirage...*
...last week, said they were arrested on	Saturday	*along with Getty photographer Joe Raedle...*
...since coalition air strikes began	Saturday	*, a figure that could not be confirmed...*
...United State to launch air raids on	March 19	*, are in a hurry to get out of whether NATO...*

Figure 3: All the temporal expressions in the following sentences were normalized at date level as 'Mar 19 2011'.

Given the query, a number of sentences are retrieved by search engine. Dates are extracted from these sentences. These dates are then ranked by their "salience" in the set of documents. The idea behind the notion of salient date is that if a date is important in a sub-corpus (Lucene output), then we can say that important events occurred at this date, and then that these events must appear in a timeline.

In practice, salience is mostly defined by the number of occurrences of the date in the documents from the search engine, as well as some other features that are used to feed a machine learning classifier.

The output of this salient date algorithm is then a ranked list of dates, where each date comes together with a set of sentences that contain this date and that are relevant to the query. We call *temporal clusters* these sets of sentences linked to a specific date (see Figures 2 and 3).

5 Event Ranking

Our ranking mechanism relies on the mutual relation between relevance and salience. It aims at ranking events based on these two factors. The problem of information redundancy will be addressed by a reranking step in §6. Our principal motivation is that an event has more chance to be selected into a timeline if it is both relevant to the topic and important, or in other words, salient w.r.t other related events. The concepts of relevance and salience are realized in our ranking function by considering term proximity and date frequency, respectively.

Previous works in event detection normally formalize events as individual terms or syntactic patterns, which facilitates the use of text content. Instead, as our method utilizes both time and text content, we come to a formalization of an event as a pair of its mentioned date and its one-sentence description.

Given an input query, the aim of ranking is to select the most relevant and salient events. The relevance of an event is calculated by vector-based query similarity, and augmented by the average relevance of its containing thematic cluster. Salience is contributed by date frequency and averaged term weight. As a result, the overall score of an event e given a query q is the multiplication of the following four factors:

$$\begin{aligned} score(e|q) = & rel_e(e|q) \\ & * rel_{cl}(cl|q) \\ & * salience_e(e|d,q) \\ & * salience_d(d|q), \end{aligned} \tag{1}$$

where:

- $rel_e(e|q)$ is the relevance of e to q (see §5.1).
- $rel_{cl}(cl|q)$ is the relevance of a thematic cluster cl to q, which is the averaged relevance of its members (see §5.2).
- $salience_e(e|d,q)$ is the salience of e w.r.t the date d that the event happens. It is calculated as the average salience of the terms in its one-sentence description. Term salience, in turn, is calculated based on term frequency in the date cluster (see §5.3).
- $salience_d(d|q)$ is the salience of d w.r.t to q. Date salience is the averaged salience of all the events in that date (see §5.4).

5.1 Event Relevance: $rel_e(e|q)$

The motivation behind considering relevance is that if an event is relevant to the query then it is an important event. We use the conventional TFIDF vector space model with bag-of-word assumption to represent document and query vectors. For relevance, the similarity between document and query vectors is the built-in Lucene score formula[2],

$$rel_e(e|q) = cosine(\vec{e}, \vec{q}) * norm_L(\vec{e}, \vec{q}). \tag{2}$$

5.2 Thematic Cluster Relevance: $rel_{cl}(cl|q)$

Date salience does not always correctly reflect the importance of event. For instance, the date of Haiti earthquake considers the earthquake itself as the main event. However, related events such as the sorrow expression of UN Secretary General also happen immediately after the earthquake but still in the same date. Such satellite events will have the same date salience as the central event. In another case, a date when there is no central event but there are many "consequent" events will also have a high salience value. E.g., on the day after the earthquake, international aids are planned; number of victims is estimated; aftermath events are invoked, etc.

Those examples show that the "one event per date" assumption is weak in reality. To overcome this weakness, we apply an hierarchical clustering technique, in which two clusters are merged if their normalized Manhattan distance is lower than a threshold θ, to generate thematic sub-clusters inside a date cluster [3]. In in-house experiments, we observed that different values of θ did not significantly vary performance. We hence selected $\theta = 0.5$ for our system.The score of each thematic cluster is then calculated as averaged document relevance,

$$rel_{cl}(cl|q) = \frac{\sum_e rel_e(e|q)}{|cl|}. \tag{3}$$

5.3 Event Salience: $salience_e(e|d, q)$

An important event tends to contain salient terms. Those terms, in turn, tend to occur frequently on a date. We hence come to measure term salience as its frequency of occurrence on the date $f(t|d, q)$, and event salience as the averaged salience of its terms. For term normalization, stopwords are removed and tokens are normalized by the Porter stemming algorithm (Porter, 1997).

$$salience_e(e|d, q) = \frac{\sum_{t \in e} f(t|d, q)}{|e| \sum_{t' \in d} f(t'|d, q)} \tag{4}$$

5.4 Date Salience: $salience_d(d|q)$

The use of temporal clusters, i.e. date clusters, is motivated by the observation that an important event happens on a salient date. Date salience is the total relevance of all events happening on that date (the numerator):

$$salience_d(d|q) = \frac{\sum_e rel_e(e|q)}{\sum_d \sum_e rel_e(e|q)}. \tag{5}$$

The denominator is used to normalize date salience so that it is comparable to other factors in (1).

6 Event Reranking

The score described in previous section leads to a ranked list of salient and relevant events. However, it does not consider the fact that some information can be redundant between events. The reranking algorithm presented in this section strives to reduce such redundancy. In principal, information redundancy is

[2] `https://lucene.apache.org/core/3_6_2/api/core/org/apache/lucene/search/Similarity.html`

[3] In our implementation, for each one-sentence document, we used the whole texts of its containing article to create its document vector. Manhattan distance is the sum of the absolute difference of term weight between two clusters

Rank	Date	Event Description
		NO RERANK
1	Mar 31 2011	*The North Atlantic Treaty Organisation takes over formal command of the military operation.*
2	Mar 31 2011	*NATO took command of operations over Libya on March 31.*
3	Mar 31 2011	*NATO takes command of the coalition campaign.*
4	Mar 19 2011	*[...] French, US and British forces launch UN-mandated air strikes and push them back.*
5	Mar 30 2011	*Libyan foreign minister Mussa Kussa defects.*
...	...	...
		RERANK
1	**Mar 31 2011**	*The North Atlantic Treaty Organisation takes over formal command of the military operation.*
2	Mar 19 2011	*[...] French, US and British forces launch UN-mandated air strikes and push them back.*
3	Mar 30 2011	*Libyan foreign minister Mussa Kussa defects.*
4	Mar 23 2011	*US Defence Secretary Robert Gates on Wednesday held talks in Cairo on the conflict in Libya [...]*
5	Apr 04 2011	*[...] photographer Manu Brabo disappeared on April 4 while covering the conflict in Libya.*
...	...	...
{2}	Mar 31 2011	*NATO took command of operations over Libya on March 31.*
...	...	...
{3}	Mar 31 2011	*NATO takes command of the coalition campaign.*
...	...	...

Figure 4: The effect of reranking on the order of events (by score).

Algorithm 1 Reranking algorithm

1: $out \leftarrow \phi$
2: **while** $(!terminate)$ **do**
3: **for** $e \in S(q) \setminus out$ **do**
4: $score(e|q)$
5: **end for**
6: $e^* = \underset{e \in S(q) \setminus out}{\text{argmax}} \; score(e|q)$
7: $out \leftarrow out \cup e^*$
8: $d^* = date(e^*)$
9: **for** $t \in e^*$ **do**
10: $used(d^*) \leftarrow used(d^*) \cup t$
11: **end for**
12: **end while**

estimated by the distinction between *used* and *unused* terms. The algorithm iteratively recomputes event salience (hence the overall event score) based on used/unused terms as follows:

$$salience^*_e(e|d,q) = \frac{\sum_{t^* \in e} f(t^*|d,q)}{|e| \sum_{t' \in d} f(t'|d,q)}, \tag{6}$$

where t^* is an unused term on the date d. A used term is the one that already occurred in better-ranked sentences. This formula is different from (4) in the distinction between used and unused terms. Each time a new event is selected, its appropriate list of used terms is updated with the terms in the one-sentence description of the selected event. Each date has its own list of used/unused terms.

The algorithm for reranking is provided in Algorithm 1. At first, the score of all sentences related to the query $S(q)$ is calculated using the formula (1) with event salience defined in (6) (lines 3-5). Then, the highest scored sentence is selected into the output (lines 6-7) and is removed from the pool. In line 8, d^* is the date when the event e happens: $d^* = date(e^*)$. The list of used terms on its date is updated with the terms from that selected sentence (lines 9-11). A new iteration restarts by recalculating score of unselected sentences according to new lists of used terms. The algorithm terminates after K iterations, i.e. when K events have been selected into timeline.

Figure 4 illustrates the effect of reranking on the order of events in a timeline. The upper shows the top events ranked by score without reranking. The date 'Mar 31 2011' appears three times in 1^{st}, 2^{nd}, and 3^{rd} events. The lower shows the ranking of events after the highest scored event has been selected. As an effect of reranking, the two events previously ranked 2^{nd} and 3^{rd} now fall down the list.

Mar 19 2011. **(2)** *With the forces of Libyan leader Moamer Kadhafi threatening rebel-held Benghazi, French, US and British forces launch UN-mandated air attacks and push them back.*
Mar 19 2011. **(9)** *Norwegian Prime Minister Jens Stoltenberg said Saturday Norway would contribute six F-16 warplanes to the international military operation –led by the United States, France and Britain– to enforce a no-fly zone over Libya.*
Mar 19 2011. **(11)** *Residents of another western town, Yafran, say nine people died there in an offensive that began on Monday.*
Mar 21 2011. **(4)** *Kadhafi's forces retreat from the rebel stronghold of Benghazi.*
Mar 22 2011. **(1)** *In Western Libya fighting intensifies in Misrata, which has been in the hands of rebels for a month.*
Mar 24 2011. **(10)** *When I ask: What is the next stage? Do you have a road map? I see they do not, he said Thursday.*
Mar 25 2011. **(12)** *Ping returned early Friday from Europe after meeting with French Foreign Minister Alain Juppe and an envoy sent by the European Union's Chief Diplomat Catherine Ashton.*
Mar 28 2011. **(6)** *Qatar follows France in recognising the rebel shadow government.*
Mar 29 2011. **(3)** *Kadhafi loyalists push the rebels back.*
Mar 30 2011. **(8)** *Kadhafi's forces push back.*
Mar 31 2011. **(5)** *NATO takes command of the coalition campaign.*
Apr 04 2011. **(13)** *Italy joins France and Qatar in recognising the rebel Transitional National Council.*
Apr 13 2011. **(7)** *A Libya contact group of 20 countries and organisations, including the rebels, meets in Qatar.*
Apr 23 2011. **(14)** *The United States carried out its first predator drone strike in Libya on Saturday, the Pentagon said, declining to give details on the targets or location.*
...

Figure 5: Timeline for the query "*Libya conflict*" created by the Rank-Rerank method. Events are shown in chronological order, each accompanied with its rank starting from 1, displayed as a number between ().

7 Evaluations

Our system for building timelines is named as RaRE, as short for "*Rank and RErank*". We use a set of 91 chronologies manually written by expert journalists from the AFP news agency (Figure 1) as golden reference summaries for evaluation. As our generated timelines are extendable, we need to define its length for evaluation. Considering the characteristics of reference summaries, we decide that if a reference summary of a timeline contains k events, we appropriately use only the k highest ranked events in the timeline for evaluation (Figure 5). The evaluations of the date selection and summary generation are presented in §7.1 and §7.2, respectively.

7.1 Evaluate Date Selection

We evaluate the dates selected by timelines returned by our system. The purposes of this evaluation are two-fold: *i)* Since time (as date in our case) is an essential dimension of chronological timeline, it is necessary to evaluate the time selected by timelines; *ii)* The novelty of this work w.r.t Kessler et al. (2012) is the mixture of content and temporal information. We need to show empirical evidences that at least, this mixture does not break the performance of date selection.

The dates occurring in a timeline are compared with the dates occurring in its reference timeline using Mean Average Precision (MAP) metric. It should be noted that by using MAP@k as evaluation metric, a date with higher rank has more impact than another date with lower rank. We use two systems presented in Kessler et al. (2012), named as DFIDF and ML in Table 1, for comparison as follows:

- DFIDF is an unsupervised system solely relying on date frequency with a tfidf-like scoring function. This method uses the AFP corpus, the same as the one used in our work. As the AFP corpus is temporally analyzed, the method indexes all the occurrences of dates in the corpus. Dates are then scored and ranked with so-called DFIDF, a tfidf-like scoring mechanism.

- ML is a supervised system that learns a classifier and ranks unseen dates based on classification

System	MAP
DFIDF	71.46
ML	79.18
RaRE	77.83

Table 1: Comparison of salient date detection using MAP.

System	P	R
DFIDF*	27.24	25.50
ML*	29.93	27.54
RaRE-no-rerank	28.82	24.47
RaRE	31.23	26.63

Table 2: Comparison of MDS using ROUGE at 95% confidence interval.

confidence. The method leverages the dates in reference summaries to create training data with salient/non-salient examples. Temporal features such as date frequency, DCT, novelty, etc., are extracted to learn an adaptive boosting classifier.

As shown in Table 1, our method is close to ML. This result is encouraging as ML requires training data; and on the other hand, our system is not designed to directly solve the task of date selection. As expected, our system beats the unsupervised system DFIDF by a large margin. This superiority shows that the mixture of temporal information and content leads to an improvement on date selection over using only the former.

7.2 Evaluate Summary Generation

In order to evaluate timelines as text summaries, we ignore dates and consider all the entries in a timeline as one summary. We use ROUGE metric (Lin, 2004) to evaluate generated timelines against reference summaries.

The following baselines are implemented (Table 2): In DFIDF*, salient dates are taken from the outputs of the DFIDF system described in previous section. Each salient date is equivalent to a cluster containing all the events happening in that date. We then select the event the most relevant to the query, i.e. the event with the highest Lucene score, as representative of that salient date. Note that consequently, DFIDF* makes an assumption, which is not assumed in RaRE, that there is only one event happens in a particular date. The same assumption is presumed in Battistelli et al. (2013). However, because their system is particularly designed for French and is intended to parse small corpora, we could not conduct a direct comparison with their method. ML* is built similarly to DFIDF*, except that salient dates are instead taken from the ML system. The RaRE-no-rerank system is identical to RaRE in the ranking step, but the reranking step is omitted.

Our system is superior to DFIDF* as expected. Moreover, it outperforms ML*, even though ML* performs better on the task of date selection. Among these three systems that combine temporal information and textual contents for summary generation, our system is the most successful. Furthermore, RaRE outperforms RaRE-no-rerank, which shows that reducing redundancy by reranking improves the performance of summary generation.

8 Conclusion and Future Work

We presented a two-step inter-cluster ranking algorithm for event selection. The *rank* step sorts events based on their salience and query relevance. The event scoring function is based on both date frequency induced from temporal analyzed texts and term weighting induced from contents to reflect these two factors. The *rerank* step allows to reduce information redundancy by using inter-sentence dependency between the descriptions of events happening in the same time period (i.e. the same date in this work).

Ranking based on sentences may be sensitive to sparsity. In the future, we will expand local contexts, for instance, to neighboring sentences, to acquire richer textual representation of events. One remaining issue is that reference chronologies, written by the journalists, are very subjective, and that we have only one example of chronology per topic. In the future, we will conduct a manual evaluation in order to complete results from this automatic evaluation. With the help of a validation interface, journalists will be provided ranked list of events w.r.t. their queries. They will then be able to select and edit the events that they wish to validate for their future timelines. Such an interface will both help journalists to produce new timelines, and bring a new evaluation methodology for our system.

Acknowledgements

This work has been partially funded by French National Research Agency (ANR) under project Chronolines (ANR-10-CORD-010). We would like to thank the French News Agency (AFP) for providing us with the corpus. We would like to thank anonymous reviewers for comments and suggestions on the paper.

References

James Allan, Ron Papka, and Victor Lavrenko. 1998. On-line new event detection and tracking. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 37–45, New York, NY, USA. ACM.

Omar Alonso, Michael Gertz, and Ricardo Baeza-Yates. 2009. Clustering and exploring search results using timeline constructions. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 97–106, New York, NY, USA. ACM.

Delphine Battistelli, Thierry Charnois, Jean-Luc Minel, and Charles Teissedre. 2013. Detecting salient events in large corpora. In *Proceedings of the 14th international conference on Computational Linguistics and Intelligent Text Processing*, CICLing'13, Berlin, Heidelberg. Springer-Verlag.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 335–336, New York, NY, USA. ACM.

Katja Filippova. 2010. Multi-sentence compression: finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 322–330, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization - Volume 4*, NAACL-ANLP-AutoSum '00, pages 40–48, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rémy Kessler, Xavier Tannier, Caroline Hagège, Véronique Moriceau, and André Bittar. 2012. Finding salient dates for building thematic timelines. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 730–739, Jeju Island, Korea, July. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, July. Association for Computational Linguistics.

Saša Petrović, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 181–189, Stroudsburg, PA, USA. Association for Computational Linguistics.

M. F. Porter. 1997. An algorithm for suffix stripping. In *Readings in information retrieval*, pages 313–316. Morgan Kaufmann Publishers Inc.

Hassan Sayyadi, Matthew Hurst, and Alexey Maykov. 2009. Event detection and tracking in social streams. In *Proceedings of the Third International Conference on Weblogs and Social Media, ICWSM 2009, San Jose, California, USA, May 17-20, 2009*.

Jannik Strötgen and Michael Gertz. 2013. Multilingual and Cross-domain Temporal Tagging. *Language Resources and Evaluation*, 47(2):269–298.

Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. 2007. Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 2903–2908, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. 2011. Timeline generation through evolutionary trans-temporal summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 433–443, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qiankun Zhao, Prasenjit Mitra, and Bi Chen. 2007. Temporal and information flow based event detection from social text streams. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 2*, AAAI'07, pages 1501–1506. AAAI Press.

Integrating Language and Vision to Generate Natural Language Descriptions of Videos in the Wild

Jesse Thomason* University of Texas at Austin jesse@cs.utexas.edu

Subhashini Venugopalan* University of Texas at Austin vsub@cs.utexas.edu

Sergio Guadarrama University of California Berkeley sguada@eecs.berkeley.edu

Kate Saenko University of Massachusetts Lowell saenko@cs.uml.edu

Raymond Mooney University of Texas at Austin mooney@cs.utexas.edu

Abstract

This paper integrates techniques in natural language processing and computer vision to improve recognition and description of entities and activities in real-world videos. We propose a strategy for generating textual descriptions of videos by using a factor graph to combine visual detections with language statistics. We use state-of-the-art visual recognition systems to obtain confidences on entities, activities, and scenes present in the video. Our factor graph model combines these detection confidences with probabilistic knowledge mined from text corpora to estimate the most likely subject, verb, object, and place. Results on YouTube videos show that our approach improves both the joint detection of these latent, diverse sentence components and the detection of some individual components when compared to using the vision system alone, as well as over a previous n-gram language-modeling approach. The joint detection allows us to automatically generate more accurate, richer sentential descriptions of videos with a wide array of possible content.

1 Introduction

Integrating language and vision is a topic that is attracting increasing attention in computational linguistics (Berg and Hockenmaier, 2013). Although there is a fair bit of research on generating natural-language descriptions of images (Feng and Lapata, 2013; Yang et al., 2011; Li et al., 2011; Ordonez et al., 2011), there is significantly less work on describing videos (Barbu et al., 2012; Guadarrama et al., 2013; Das et al., 2013; Rohrbach et al., 2013; Senina et al., 2014). In particular, much of the research on videos utilizes artificially constructed videos with prescribed sets of objects and actions (Barbu et al., 2012; Yu and Siskind, 2013). Generating natural-language descriptions of videos *in the wild*, such as those posted on YouTube, is a very challenging task.

In this paper, we focus on selecting content for generating sentences to describe videos. Due to the large numbers of video actions and objects and scarcity of training data, we introduce a graphical model for integrating statistical linguistic knowledge mined from large text corpora with noisy computer vision detections. This integration allows us to infer which vision detections to trust given prior linguistic knowledge. Using a large, realistic collection of YouTube videos, we demonstrate that this model effectively exploits linguistic knowledge to improve visual interpretation, producing more accurate descriptions compared to relying solely on visual information. For example, consider the frames of the video in Figure 1. Instead of generating the inaccurate description "A person is playing on the keyboard in the kitchen" using purely visual information, our system generates the more correct "A person is playing the piano in the house" by using statistics mined from parsed corpora to improve the interpretation of the uncertain visual detections, such as the presence of both a computer keyboard and a piano in the video.

2 Background and Related Work

Several recent projects have integrated linguistic and visual information to aid description of images and videos. The most related work on image description is Baby Talk (Kulkarni et al., 2011), which uses

*Indicates equal contribution

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1218–1227, Dublin, Ireland, August 23-29 2014.

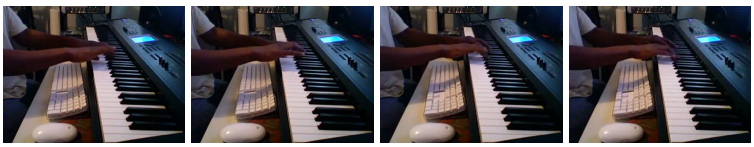


Figure 1: Frames which depict a person playing a piano in front of a keyboard from one of the videos in our dataset. Purely visual information is more confident in the computer keyboard's presence than the piano's, while our model can correctly determine that the person is more likely to be playing the piano than the computer keyboard.

a Conditional Random Field (CRF) to integrate visual detections with statistical linguistic knowledge mined from parsed image descriptions and Google queries, and the work of Yang et al. (2011) which uses corpus statistics to aid the description of objects and scenes. We go beyond the scope of these previous works by also selecting verbs through the integration of activity recognition from video and statistics from parsed corpora.

With regard to video description, the work of Barbu et al. (2012) uses a small, hand-coded grammar to describe a sparse set of prescribed activities. In contrast, we utilize corpus statistics to aid the description of a wide range of naturally-occurring videos. The most similar work is (Krishnamoorthy et al., 2013; Guadarrama et al., 2013) which uses an n-gram language model to help determine the best subject-verb-object for describing a video. Krishnamoorthy et al. (2013) used a limited set of videos containing a small set of 20 entities, and the work of Guadarrama et al. (2013) showed an advantage of using linguistic knowledge only for the case of "zero shot activity recognition," in which the appropriate verb for describing the activity was never seen during training. Compared to this prior work, we explore a much larger set of entities and activities (see Section 3.2) and add scene recognition (see Section 3.3) to further enrich the descriptions. Our experiments demonstrate that our graphical model produces a more accurate subject-verb-object-place description than these simpler n-gram language modeling approaches.

Our Contributions:

- We present a new method, a *Factor Graph Model* (FGM), to perform content selection by integrating visual and linguistic information to select the best subject-verb-object-place description of a video.
- Our model includes scene (location) information which has not been addressed by previous video description works (Barbu et al., 2012; Krishnamoorthy et al., 2013; Guadarrama et al., 2013).
- We demonstrate the scalability of our model by evaluating it on a large dataset of naturally occurring videos (1297 training, 670 testing), recognizing sentential subjects out of 45 candidate entities, objects out of 218 candidate objects, verbs out of 218 candidate activities, and places out of 12 candidate scenes.

3 Approach

Our overall approach uses a probabilistic graphical model to integrate the visual detection of entities, activities, and scenes with language statistics to determine the best subject, verb, object, and place to describe a given video. A descriptive English sentence is generated from the selected sentential components.

3.1 Video Dataset

We use the video dataset collected by Chen and Dolan (2011). The dataset contains 1,967 short YouTube video clips paired with multiple human-generated natural-language descriptions. The video clips are 10 to 25 seconds in duration and typically consist of a single activity. Portions of this dataset have been used in previous work on video description (Motwani and Mooney, 2012; Krishnamoorthy et al., 2013; Guadarrama et al., 2013). We use 1,297 randomly selected videos for training and evaluate predictions on the remaining 670 test videos.

3.2 Visual Recognition of Subject, Verb, and Object

We utilize the visual recognition techniques employed by Guadarrama et al. (2013) to process the videos and produce probabilistic detections of grammatical subjects, verbs, and objects. In our data-set there are 45 candidate entities for the grammatical subject (such as *animal, baby, cat, chef*, and *person*) and 241 for the grammatical object (such as *flute, motorbike, shrimp, person*, and *tv*). There are 218 candidate activities for the grammatical verb, including *climb, cut, play, ride*, and *walk*.

Entity Related Features From each video two frames per second are extracted and passed to pre-trained visual object classifiers and detectors. As in Guadarrama et al. (2013), we compute representations based on detected objects using ObjectBank (Li et al., 2010) and the 20 PASCAL (Everingham et al., 2010) object classes for each frame. We use the PASCAL scores and ObjectBank scores with max pooling over the set of frames as the entity descriptors for the video clip. Additionally, to be able to recognize more objects, we use the LLC-10k proposed by Deng et al. (2012) which was trained on ImageNet 2011 object dataset with 10k categories. LLC-10K uses a bank of linear SVM classifiers over pooled local vector-quantized features learned from the 7K bottom level synsets of the 10K ImageNet database. We aggregate the 10K classifier scores obtained for each frame by doing max pooling across frames.

Activity Related Features We use the activity recognizers described in Guadarrama et al. (2013) to produce probabilistic verb detections. They extract Dense Trajectories developed by Wang et al. (2011) and compute HoG (Histogram of Gradients), HoF (Histograms of Optical Flow) and MBH (Motion Boundary Histogram) features over space time volumes around the trajectories. We used the default parameters proposed in Wang et al. (2011) ($N = 32$, $n_\sigma = 2$, $n_r = 3$) and adopted a standard bag-of-features representation. We construct a codebook for each descriptor (Trajectory, HoG, HoF, MBH) separately. For each descriptor we randomly sampled 100K points and clustered them using K-means into a codebook of 4000 words. Descriptors are assigned to their closest vocabulary word using Euclidean distance. Each video is then represented as a histogram over these clusters.

Multi-channel SVM To allow object and activity features inform one another, we combine all the features extracted using a multi-channel approach inspired by Zhang et al. (2007) to build three non-linear SVM (Chang and Lin, 2011) classifiers for the subject, verb, and object, as described in Guadarrama et al. (2013). Note that we do not employ the hierarchical semantic model of Guadarrama et al. (2013) to augment our object or activity recognition. In addition, each SVM learns a Platt scaling (Platt, 1999) to predict the label and a visual confidence value, $C(t) \in [0, 1]$, for each entity or activity t. The output of the SVMs constitute the visual confidences on subject, verb, and object in all the models described henceforth.

3.3 Visual Scene Recognition

In addition to the techniques employed by Guadarrama et al. (2013) used to obtain probabilistic detections of grammatical subjects, verbs, and objects, we developed a novel scene detector based on state-of-the-art computer vision methods.

We examined the description of all the 1,967 videos in the YouTube dataset and extracted scene words from the dependency parses as described in Section 3.4. With the help of WordNet[1] we grouped the list of scene words and their synonyms into distinct scene classes. Based on the frequency of mentions and the coverage of scenes in the dataset, we shortlisted a set of 12 final scenes (*mountain, pool, beach, road, kitchen, field, snow, forest, house, stage, track*, and *sky*).

For the detection itself, we follow Xiao et al. (2010) and select several state-of-the-art features that are potentially useful for scene recognition. We extract GIST, HOG2x2, SSIM (self-similarity) and Dense SIFT descriptors. We also extract LBP (Local Binary Patterns), Sparse SIFT Histograms, Line features, Color Histograms, Texton Histograms, Tiny Images, Geometric Probability Map and Geometric specific histograms. The code for extracting the features and computing kernels for the features is taken from

[1] `http://wordnet.princeton.edu`

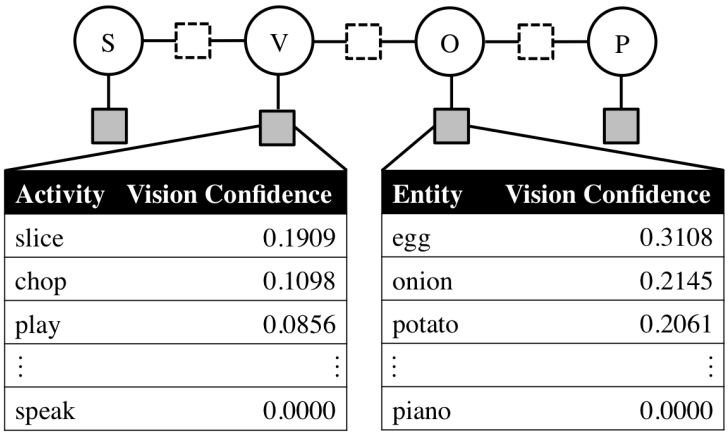
the original papers as described in Xiao et al. (2010). Using the features and kernels, we train one-vs-all SVMs (Chang and Lin, 2011) to classify images into scene categories. As in Xiao et al. (2010), this gave us 51 different SVM classifiers with different feature and kernel choices. We use the images from the UIUC 15 scene dataset (Lazebnik et al., 2006) and the SUN 397 scene dataset (Xiao et al., 2010) for training the scene classifiers for all scenes except *kitchen*. The training images for *kitchen* were obtained by selecting 100 frames from about 15 training videos, since the classifier trained on images from the existing scene datasets performed extremely poorly on the videos. We use all the classifiers to detect scenes for each frame. We then average the scene detection scores over all the classifiers across all the frames of the video. This gives us visual confidence values, $C(t)$, over all scene categories t for the video.

3.4 Language Statistics

A key aspect of our approach is the use of language statistics mined from English text corpora to bias visual interpretation. Like Krishnamoorthy et al. (2013), we use dependency-parsed text from four large "out of domain" corpora: English Gigaword, British National Corpus (BNC), ukWac and WaCkypedia_EN. We also use a small, specialized "in domain" corpus: dependency parsed sentences from the human-generated, English descriptions for the YouTube training videos mentioned in Section 3.1. We extract SVOP (subject, verb, object, place) tuples from the dependency parses. The subject-verb relationships are identified using *nsubj* dependencies, the verb-object relationships using *dobj* and *prep* dependencies. Object-place relationships are identified using the *prep* dependency, checking that the noun modified by the preposition is one of our recognizable places (or synonyms of the recognizable scenes as indicated by WordNet). We then extract co-occuring SV, VO, and OP bigram statistics from the resulting SVOP tuples to inform our factor-graph model, which uses both the out-of-domain (p_o) and in-domain (p_i) bigram probabilities.

3.5 Content Selection Using Factor Graphs

In order to combine visual and linguistic evidence, we use the probabilistic factor-graph model shown in Figure 2. This model integrates the uncertain visual detections described in Sections 3.2 and 3.3 with the language statistics described in Section 3.4 to predict the best words for describing the subject (S), verb (V), object (O), and place (P) for each test video. After instantiating the potential functions for this model, we perform a maximum a posteriori (MAP) estimation (via the max-product algorithm) to determine the most probable joint set of values for these latent variables.



Activity	Vision Confidence
slice	0.1909
chop	0.1098
play	0.0856
⋮	⋮
speak	0.0000

Entity	Vision Confidence
egg	0.3108
onion	0.2145
potato	0.2061
⋮	⋮
piano	0.0000

Figure 2: The factor graph model used for content selection (right), and sample frames from a video to be described (left). Visual confidence values are observed (gray potentials) and inform sentence components. Language potentials (dashed) connect latent words between sentence components. Samples of the vision confidence values used as observations for the verb and object are shown for the example test video.

Observation Potentials. The observations in our model take the form of confidence scores from the visual detectors described in Sections 3.2 and 3.3. That is, the potential for each sentence component $k \in \{S, V, O, P\}$, $\phi_k(t) = C_k(t)$ is the detection confidence that the classifier for component k (C_k) gives to the word t.

Language Potentials. Language statistics were gathered as described in Section 3.4 and used to determine the language potentials as follows:

$$\phi_{k,l}(t, s) := p(l = s|k = t) := \alpha p_o(l = s|k = t) + (1 - \alpha)p_i(l = s|k = t)$$

Where k and l are two contiguous components in the SVOP sequence and t and s are words that are possible values for these two components, respectively. We would expect

$$\phi_{V,O}(\text{ride}, \text{motorbike}) := p(\text{O=motorbike}|\text{V=ride})$$

to be relatively high, since *motorbike* is a likely object of the verb *ride*. The potential between two sequential components k and l in the SVOP sequence is computed by linearly interpolating the bigram probability observed in the out-of-domain corpus of general text (p_o) and the in-domain corpus of video descriptions (p_i). The interpolation parameter α adjusts the importance of these two corpora in determining the bigram probability. We optimized performance by fixing $\alpha = 0.25$ when cross-validating on the training data. This weighting effectively allows general text corpora to be used to smooth the probability estimates for video descriptions. We note that meaningful information would likely be captured by non-contiguous language potentials such as $\phi_{V,P}$, but that the resulting factor graphs would contain cycles, preventing us from performing exact inference tractably.

3.6 Sentence Generation

Finally, we use the SVOP tuple chosen by our model to generate an English sentence using the following template: *"Determiner (A,The) - Subject - Verb (Present, Present Continuous) - Preposition (optional) - Determiner (A,The) - Object (optional) - Preposition - Determiner (A,The) - Place (optional)"* The most probable prepositions are identified using preposition-object and preposition-place bigram statistics mined from the dependency parsed corpora described in Section 3.4. Given an SVOP tuple, our objective is to generate a rich sentence using the subject, verb, object, and place information. However, it is not prudent to add the object and place to the description of all videos since some verbs may be intransitive and the place information may be redundant. In order to achieve the best set of components to include, we use the above template to first generate a set of candidate sentences based on the SVO triple, SVP triple and the SVOP quadruple. Then, each sentence type (SVO, SVP, and SVOP) is ranked using the BerkeleyLM language model (Pauls and Klein, 2011) trained on the GoogleNgram corpus. Finally, we output the sentence with the highest average 5-gram probability in order to normalize for sentence length.

4 Experimental Results

We compared using the vision system alone to our model, which augments that system with linguistic knowledge. Specifically, we consider the *Highest Vision Confidence* (HVC) model, which takes for each sentence component the word with the highest confidence from the state-of-the-art vision detectors described in Sections 3.2 and 3.3. We compare the results of this model on the 670 test videos to those of our *Factor Graph Model* (FGM), as discussed in Section 3.5.

4.1 N-gram Baseline

Additionally, we compare both models against the existing, baseline *n-gram* model of Krishnamoorthy et al. (2013) by extending their best n-gram model to support places. To be specific, we build a quadragram model, similar to the trigram model of Krishnamoorthy et al. (2013). We first extract SVOP tuples from the dependency parses as described in Section 3.4. We then train a backoff language model with Kneyser-Ney smoothing (Chen and Goodman, 1996) for estimating the likelihood of the SVOP quadruple. On quadruples that are not seen during training, this quadragram language model backs off to SVO

Most	S%	V%	O%	[P]%	SVO%	SVO[P]%
n-gram	76.57	11.04	11.19	18.30	2.39	1.86
HVC	76.57	$^{+}$22.24	11.94	17.24	$^{+}$4.33	$^{+}$2.92
FGM	76.42	$^{+}$21.34	12.39	19.89	$^{+}$**5.67**	$^{+}$3.71
Any						
n-gram	86.87	19.25	21.94	21.75	5.67	2.65
HVC	86.57	$^{+}$38.66	22.09	21.22	$^{+}$10.15	$^{+}$4.24
FGM	86.27	$^{+}$37.16	$^{+}$**24.63**	**24.67**	$^{+}$10.45	$^{+}$6.10

Table 1: Average binary accuracy of predicting the **most** common word (top) and of predicting **any** given word (bottom). **Bold** entries are statistically significantly ($p < 0.05$) greater than the HVC model, while $^{+}$ entries are significantly greater than the n-gram model. No model scored significantly higher than FGM on any metric. [P] indicates that the score ranges only over the subset of videos for which any annotator provided a place.

triple and subject-verb, verb-object, object-place bigrams to estimate the probability of the quadruple. As in the case of the factor graph model, we consider the effect of learning from a domain specific text corpus. We build quadragram language models for both out-of-domain and in-domain text-corpora described in Section 3.4. The probability of a quadragram in the language model is computed by linearly interpolating the probabilities from the in-domain and out-of-domain corpus. We experiment with different number of top subjects, objects, verbs, and places to estimate the most likely SVOP quadruple from the quadragram language model. We report the results for the best performing n-gram model that considers the top 5 subjects, 5 objects, 10 verbs, and 3 places based on the vision confidences and an out-of-domain corpus weight of 1. This model also incorporates *verb expansion* as described in the original work (Krishnamoorthy et al., 2013).

4.2 Content Evaluation

Table 1 shows the accuracy of the models when their prediction for each sentence component is considered correct only if it is the word *most commonly* used by human annotators to describe the video, as well as the accuracy of the models when the prediction is considered correct if used by *any* of the annotators to describe the video. We evaluate the accuracy of each component (S,V,O,P) individually, and for complete SVO and SVOP tuples, where *all* components must be correct in order for a complete tuple to be judged correct. Because only about half (56.3%) of test videos were described with a place by some annotator, accuracies involving places ("[P]") are averaged only over the subset of videos for which any annotator provided a place. Significance was determined using a paired t-test which compared the distributions of the binary correctness of each model's prediction on each video for the specified component(s).

We also use the WUP metric from Wordnet::Similarity[2] to measure the quality of the predicted words to account for semantically similar words. For example, where the binary metric would mark "slice" as an incorrect substitute for "cut", the WUP metric will provide "partial credit" for such predictions. The results using WUP similarity metrics for the most common word and any valid word (maximum WUP similarity is chosen from among valid words) are presented in Table 2. Since WUP provides scores are in the range [0,1], we view the scores as "percent relevance," and we obtain tuple scores for each sentence by taking the product of the component WUP scores.

5 Discussion

It is clear from the results in Table 1 that both the HVC and the FGM outperform the n-gram language model approach used in the most-similar previous work (Krishnamoorthy et al., 2013; Guadarrama et al., 2013). Note that while Krishnamoorthy et al. (2013) showed an improvement with an n-gram model considering only the top few vision detections, the FGM considers vision confidences over the entire set

[2] `http://wn-similarity.sourceforge.net/`

Most	S%	V%	O%	[P]%	SVO%	SVO[P]%
n-gram	89.00	41.56	44.01	**57.62**	17.53	10.83
HVC	89.09	$^{+*}$48.85	43.99	56.00	$^{+}$20.82	$^{+}$12.95
FGM	89.01	$^{+}$47.05	$^{+}$**45.29**	$^{+}$**59.64**	$^{+}$21.54	$^{+}$**14.50**
Any						
n-gram	96.60	55.08	65.52	**61.98**	35.70	22.84
HVC	96.54	$^{+*}$65.61	65.32	60.67	$^{+}$42.53	$^{+}$27.75
FGM	96.32	$^{+}$63.49	$^{+}$**67.52**	$^{+}$**64.68**	$^{+}$42.43	$^{+}$29.34

Table 2: Average WUP score of the predicted word against the **most** common word (top) and the maximum score against **any** given word (bottom). **Bold** entries are statistically significantly ($p < 0.05$) greater than the HVC model; $^{+}$ entries are significantly greater than the n-gram model; * entries are significantly greater than the FGM. [P] indicates that the score ranges only over the subset of videos for which any annotator provided a place.

of grammatical objects. Additionally, our models are evaluated on a much more diverse set of videos while Krishnamoorthy et al. (2013) evaluate the n-gram model on 185 videos (a small subset of the 1,967 videos containing the 20 grammatical objects that their system recognized).

The performance differences between the vision system (HVC) and our integrated model (FGM) are modest but significant in important places. Specifically, the FGM makes improvements to SVO (Table 1, top) and SVOP (Table 2, top) tuple accuracies. FGM also significantly improves both the O and [P] (Table 1, bottom, and Table 2) component accuracies, suggesting that it can help clean up some noise from the vision systems even at the component level by considering related bigram probabilities. FGM causes no significant losses under the binary metric, but performs worse than the HVC model on predicting a verb component semantically similar to the correct verb under the WUP metric (Table 2). This loss on the verb component is worth the gains in tuple accuracy, since tuple prediction is the more difficult and most central part of the content selection task. Additionally, experiments by the authors of Guadarrama et al. (2013) on Amazon Mechanical Turk have shown that humans tend to heavily penalize tuples and descriptions even if they have most of the components correct.

Table 3 shows frames from some test videos and the sentence components chosen by the models to describe them. In the top four videos we see the FGM improving raw vision results. For example, it determines that a person is more likely slicing an onion than an egg. Some specific confidence values for the HVC can be seen for this video in Figure 2. In the bottom two videos of Table 3 we see the HVC performing better without linguistic information. For example, the FGM intuits that a person is more likely to be driving a car than lifting it, and steers the prediction away from the correct verb. This may be part of a larger phenomenon in which YouTube videos often depict unusual actions, and consequently general language knowledge can sometimes hurt performance by selecting more common activities.

6 Future Work

Compared to the human gold standard descriptions, there appears to be room for improvement in detecting activities, objects, and scenes with high precision. Visual recognition of entities and activities in diverse real-world videos is extremely challenging, partially due to lack of training data. As a result our current model is faced with large amounts of noise in the vision potentials, especially for objects. Going forward, we believe that improving visual recognition will allow the language statistics to be even more useful. We are currently exploring deep image feature representations (Donahue et al., 2013) to improve object and verb recognition, as well as model transfer from large labeled object ontologies (Deng et al., 2009).

From the generation perspective, there is scope to move beyond the template based sentence generation. This becomes particularly relevant if we detect multiple grammatical objects such as adjectives or adverbs. We need to decide whether additional grammatical objects would enrich the sentence de-

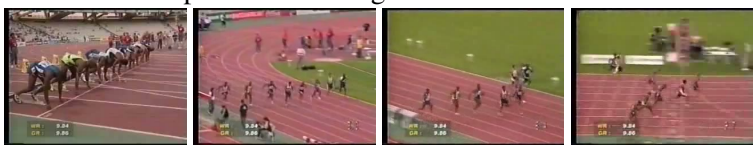
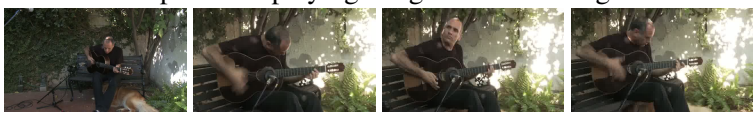
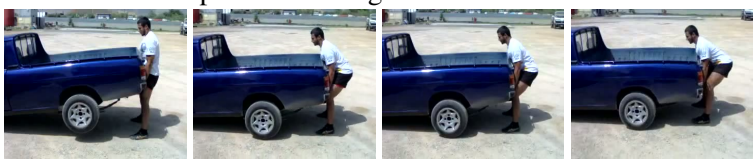
FGM improves over HVC	
"A person is slicing the onion in the kitchen" 	Gold: person, slice, onion, *(none)* HVC: person, slice, egg, kitchen FGM: person, slice, onion, kitchen
"A person is running a race on the road" 	Gold: person, run, race, *(none)* HVC: person, ride, race, ground FGM: person, run, race, road
"A person is playing the guitar on the stage" 	Gold: person, play, guitar, tree HVC: person, play, water, kitchen FGM: person, play, guitar, stage
"A person is playing a guitar in the house" 	Gold: person, play, guitar, *(none)* HVC: person, pour, chili, kitchen FGM: person, play, guitar, house
HVC better alone	
"A person is lifting a car on the road" 	Gold: person, lift, car, ground HVC: person, lift, car, road FGM: person, drive, car, road
"A person is pouring the egg in the kitchen" 	Gold: person, pour, mushroom, kitchen HVC: person, pour, egg, kitchen FGM: person, play, egg, kitchen

Table 3: Example videos and: (Gold) the most common SVOP provided by annotators; (HVC) the highest vision confidence selections; (FGM) the selections from our factor graph model. The top section shows videos where the FGM improved over HVC; the bottom shows videos where the HVC did better alone. For each video, the sentence generated from the components chosen from the more successful system is shown.

scription and identify when to add them appropriately. With increasing applications for such systems in automatic video surveillance and video retrieval, generating richer and more diverse sentences for longer videos is an area for future research. In comparison to previous approaches (Krishnamoorthy et al., 2013; Yang et al., 2011) the factor graph model can be easily extended to support this. Additional nodes can be attached suitably to the graph to enable the prediction of adjectives and adverbs to enrich the base SVOP tuple.

7 Conclusions

This work introduces a new framework to generate simple descriptions of short videos by integrating visual detection confidences with language statistics obtained from large textual corpora. Experimental results show that our approach achieves modest improvements over a pure vision system and significantly improves over previous methods in predicting the complete subject-verb-object and subject-verb-object-place tuples. Our work has a broad coverage of objects and verbs and extends previous works by predicting place information.

There are instances where our model fails to predict the correct verb when compared to the HVC model. This could partially be because the SVM classifiers that detect activity already leverage entity information during training, and adding external language does not appear to improve verb prediction significantly. Further detracting from performance, our model occasionally propagates, rather than correcting, errors from the HVC. For example, when the HVC predicts the correct verb and incorrect object, such as in "person ride car" when the video truly depicts a person riding a motorbike, our model selects the more likely verb pairing "person drive car", extending the error from the object to the verb as well.

Despite these drawbacks, our approach predicts complete subject-verb-object-place tuples more closely related to the most commonly used human descriptions than vision alone (Table 2), and in general improves both object and place recognition accuracies (Tables 1, 2).

Acknowledgements

This work was funded by NSF grant IIS1016312, DARPA Minds Eye grant W911NF-10-9-0059, and NSF ONR ATL grant N00014-11-1-0105. Some of our experiments were run on the Mastodon Cluster (NSF grant EIA-0303609).

References

Andrei Barbu, Alexander Bridge, Zachary Burchill, Dan Coroian, Sven Dickinson, Sanja Fidler, Aaron Michaux, Sam Mussman, Siddharth Narayanaswamy, Dhaval Salvi, Lara Schmidt, Jiangnan Shangguan, Jeffrey Mark Siskind, Jarrell Waggoner, Song Wang, Jinlian Wei, Yifan Yin, and Zhiqi Zhang. 2012. Video in sentences out. In *Association for Uncertainty in Artificial Intelligence (UAI)*.

Tamara Berg and Julia Hockenmaier. 2013. Workshop on vision and language. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. NAACL.

Chih-Chung Chang and Chih-Jen Lin. 2011. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

David L. Chen and William B. Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 190–200. Association for Computational Linguistics.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics (ACL)*, pages 310–318. Association for Computational Linguistics.

Pradipto Das, Chenliang Xu, Richard F. Doell, and Jason J. Corso. 2013. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jia Deng, Kai Li, Minh Do, Hao Su, and Li Fei-Fei. 2009. Construction and analysis of a large scale image ontology. Vision Sciences Society.

Jia Deng, Jonathan Krause, Alex Berg, and Li Fei-Fei. 2012. Hedging Your Bets: Optimizing Accuracy-Specificity Trade-offs in Large Scale Visual Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2013. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, June.

Yansong Feng and Mirella Lapata. 2013. Automatic caption generation for news images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(4):797–812.

Sergio Guadarrama, Niveda Krishnamoorthy, Girish Malkarnenkar, Subhashini Venugopalan, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2013. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *IEEE International Conference on Computer Vision (ICCV)*, December.

Niveda Krishnamoorthy, Girish Malkarnenkar, Raymond J. Mooney, Kate Saenko, and Sergio Guadarrama. 2013. Generating natural-language video descriptions using text-mined knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 541–547.

Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Alexander Berg, Yejin Choi, and Tamara Berg. 2011. Baby talk: Understanding and generating image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2169–2178. IEEE.

Li-Jia Li, Hao Su, Eric Xing, and Li Fei-Fei. 2010. Object bank: A high-level image representation for scene classication and semantic feature sparsification. In *Advances in Neural Information Processing Systems (NIPS)*.

Siming Li, Girish Kulkarni, Tamara L. Berg, Alexander C. Berg, and Yejin Choi. 2011. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL)*, pages 220–228, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tanvi S. Motwani and Raymond J. Mooney. 2012. Improving video activity recognition using object recognition and text mining. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pages 600–605.

Vicente Ordonez, Girish Kulkarni, and Tamara L. Berg. 2011. Im2text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, pages 1143–1151.

Adam Pauls and Dan Klein. 2011. Faster and smaller n-gram language models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 258–267. Association for Computational Linguistics.

John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances In Large Margin Classifiers*, pages 61–74. MIT Press.

Marcus Rohrbach, Qiu Wei, Ivan Titov, Stefan Thater, Manfred Pinkal, and Bernt Schiele. 2013. Translating video content to natural language descriptions. In *IEEE International Conference on Computer Vision (ICCV)*.

Anna Senina, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Sikandar Amin, Mykhaylo Andriluka, Manfred Pinkal, and Bernt Schiele. 2014. Coherent multi-sentence video description with variable level of detail. *arXiv preprint arXiv:1403.6173*.

Heng Wang, Alexander Klaser, Cordelia Schmid, and Cheng-Lin Liu. 2011. Action recognition by dense trajectories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3169–3176. IEEE.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. 2010. Sun database: Large-scale scene recognition from abbey to zoo. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3485–3492.

Yezhou Yang, Ching Lik Teo, Hal Daumé, III, and Yiannis Aloimonos. 2011. Corpus-guided sentence generation of natural images. In *Conference on Emperical Methods in Natural Language Processing (EMNLP)*, pages 444–454.

Haonan Yu and Jeffrey Mark Siskind. 2013. Grounded language learning from video described with sentences. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 53–63.

Jianguo Zhang, Marcin Marszałek, Svetlana Lazebnik, and Cordelia Schmid. 2007. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision (IJCV)*, 73(2):213–238.

Cross-Topic Authorship Attribution: Will Out-Of-Topic Data Help?

Upendra Sapkota and **Thamar Solorio**
The University of Alabama at Birmingham
1300 University Boulevard
Birmingham, AL 35294, USA
{upendra,solorio}@cis.uab.edu

Manuel Montes-y-Gómez
Instituto Nacional de Astrofísica
Optica y Electrónica
Puebla, Mexico
mmontesg@ccc.inaoep.mx

Steven Bethard
The University of Alabama at Birmingham
1300 University Boulevard
Birmingham, AL 35294, USA
bethard@cis.uab.edu

Paolo Rosso
NLE Lab - PRHLT Research Center
Universitat Politècnica de València
Valencia, Spain
prosso@dsic.upv.es

Abstract

Most previous research on authorship attribution (AA) assumes that the training and test data are drawn from same distribution. But in real scenarios, this assumption is too strong. The goal of this study is to improve the prediction results in cross-topic AA (CTAA), where the training data comes from one topic but the test data comes from another. Our proposed idea is to build a predictive model for one topic using documents from all other available topics. In addition to improving the performance of CTAA, we also make a thorough analysis of the sensitivity to changes in topic of four most commonly used feature types in AA. We empirically illustrate that our proposed framework is significantly better than the one trained on a single out-of-domain topic and is as effective, in some cases, as same-topic setting.

1 Introduction

Authorship Attribution is the problem of identifying who, from a number of given candidate authors, wrote the given piece of text. The authorship attribution task can be viewed as a multi-class single-label text classification task where each author indicates a class. However, the purpose of AA is to model each author's writing style. AA methods have a wide range of applications, including Forensic Linguistics (spam filtering (de Vel et al., 2001), verifying the authorship of threatening emails), cybercrimes (identifying authors of malicious code and defending against pedophiles), and plagiarism detection (Stamatatos, 2011).

The AA methods can be useful in applied areas such as law and journalism where the identification of the true author of a piece of text (such as a ransom note) may be able to save lives or help prosecute offenders. One of the outstanding problems in AA studies is the unrealistic assumption that the samples of both known and unknown authorship are drawn from the same distribution. This assumption considerably simplifies the AA task but also limits the practical usability of the methods. In practical scenarios usually the documents under investigation are from a different domain than that of the training documents. We feel the need to advance the way AA methods are designed so that the bridge between domains will be minimized to obtain the optimum performance. Therefore, we try to improve the performance of cross-topic AA (CTAA), one of the dimensions of cross-domain AA (CDAA) where training and test data come from different topics.

In this paper, we focus on one of the outstanding research questions on AA: *Can we reliably predict the author of a document written in one topic with a predictive model developed using documents from other topics?* We hypothesize that the addition of training data even if it comes from a topic different than that of the test data improves cross-topic AA performance. To test the hypothesis, we compare the performance of our proposed model trained on documents from all available out-of-topic data with two models, one trained on single out-of-topic data and another trained on the same topic (intra-topic)

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1228–1237, Dublin, Ireland, August 23-29 2014.

data. We also compare the performance of using four widely used features in AA to demonstrate their discriminative power in intra-topic and cross-topic AA. The contributions of this study are as follows:

- We propose a new method to identify the author of a document on a topic using a predictive model trained on examples from different topics. The successful results attained indicate that authors maintain a consistent style across topics.
- This is the first comprehensive study showing empirically which widely used features in AA are effective for cross-topic AA. We demonstrate that character n-grams are a strong discriminator among authors in CTAA and that lexical features are less effective in CTAA than they are for intra-topic AA.
- We empirically illustrate that having the same amount of training documents from multiple topics is significantly better than having documents from a single topic. It shows that topic variety in training documents improves the performance of CTAA.
- We also demonstrate that across all genres, adding an extra topic to the training data gives a character n-gram model a greater boost in performance than to a stop-word, a stylistic or a lexical model. This is true regardless of the topics on which the model is trained.
- Our proposed methodology is simple to implement suggesting that our findings on cross-topic AA will be generalizable to other classification problems too.

The paper is organized as follows. Section 2 describes two cross-topic datasets while Section 3 describes the methodology for our experiments. Section 4 describes different features while Section 5 presents the experimental setup. We present the evaluation and analysis in Sections 6 and 7. In Section 8, we describe previous studies on cross-topic AA. Finally, Section 9 presents our conclusions and some future directions.

2 Cross-Topic Datasets

Although several corpora are available for traditional AA, we need datasets containing documents from a number of authors from different domains (different topics, different genres). We need many topics to be able to test cross-topic performance, and many genres to ensure that our findings are robust across different styles of text. Obtaining such corpora is a challenging task since most authorship attribution studies focus on a single domain. We have found two datasets that meet our criteria, one having both cross-topic and cross-genre flavor, and the other having only cross-topic flavor. The first corpus contains communication samples from 21 authors in six genres (Email, Essay, Blog, Chat, Phone Interview, and Discussion) on six topics (Catholic Church, Gay Marriage, War in Iraq, Legalization of Marijuana, Privacy Rights, and Sex Discrimination), which we call dataset 1. This dataset was obtained from Goldstein-Stewart *et al.* (2009). Using this dataset, it is possible to see how the performance of cross-topic AA changes across different genres.

Another corpus is composed of texts published in The Guardian daily newspaper written by 13 authors in one genre on four topics (dataset 2) due Stamatatos *et al.* (2013). It contains opinion articles (comments) about World, U.K., Culture, and Politics. Table 1 shows some statistics about the datasets.

Corpus	#authors	#genres	#topics	avg #docs/author	avg #sentences/doc	avg #words/doc
Dataset 1	21	6	6	36	31.7	600
Dataset 2	13	1	4	64	53	1034

Table 1: Some statistics about dataset 1 and dataset 2.

In dataset 1, the average document length is almost half the average document length in dataset 2, while the number of authors is almost twice as that in dataset 2. Also, in dataset 1, there is only one document written by an author on each topic on each genre. However, there are, on average, 16 documents per author per topic on each genre in dataset 2. Overall, dataset 1 seems more challenging and resembles more a realistic scenario of forensic investigations where very few short documents per author might be available.

3 Methodology

To answer our research question and test our hypothesis, we designed three training scenarios. First of all, to demonstrate the complexity of cross-topic tasks, we compare the performance between two training conditions: Intra-Topic (IT), and Single Cross-Topic (SCT). Once we show that it is important to solve this CTAA problem, we design one more training condition based on our proposed idea, Multiple Cross-Topics (MCT) and compare its performance with the IT and the SCT scenarios.

Intra-Topic (IT) In this scenario, all the documents in both the training and test data belong to the same topic. Although this is a strong assumption that does not hold true in most of the realistic scenarios, we examine AA under such conditions in order to be able to compare it with our proposed methods.

Single Cross-Topic (SCT) In this setting, the test data consists of documents from a single topic while the AA model is trained using documents belonging to another topic different than the topic of the test data, but from the same genre. For example, in dataset 1, for 'Chat' genre, a model could be trained on a topic 'Gay Marriage' and tested on the topic 'Legalization of Marijuana'. We experiment on all combinations of test/train topics, i.e., for each test topic, we train separately on each of the remaining topics.

Multiple Cross-Topics (MCT) Unlike in SCT and IT scenarios, here for each test topic, we train on documents from all available topics other than the one used for testing. Our assumption is that authors somehow maintain their unique writeprints across different topics. Therefore, even though the additional data comes from a topic different than that of the test data, we expect to see improvements in the performance of cross-topic AA.

In the SCT scenario, since there is a mismatch between the training and test topic, we expect to obtain experimental results worst than that of the IT scenario. However, we expect that the performance of cross-topic AA using our proposed MCT scenario will be better than SCT in all the cases.

4 Features

The choice of features depends greatly on the type of classification problem. Previous research has explored various types of features that can discriminate among the candidate authors. Stylistic features, character-level and word-level n-grams are the most frequently and successfully used features (Houvardas and Stamatatos, 2006; Zheng et al., 2006; Frantzeskou et al., 2007; Abbasi and Chen, 2008; Luyckx and Daelemans, 2011; Koppel et al., 2011). We consider four of the most widely used features. Our goal behind exploring four different types of features is to understand which features are the best for cross-topic AA.

Lexical Features. Bag-of-words is one of the commonly used document representations that uses single-content words as document features. Authorship attribution approaches using a bag-of-words representation have been found to be effective (Diederich et al., 2003; Kaster et al., 2005; Zhao and Zobel, 2005; Coyotl-Morales et al., 2006). We call bag-of-words the lexical features since we exclude stop-words.

Stop-Words. Stop-words carry no or very little semantic meaning of the texts, however, their use indicates the presence of certain syntactic structures. Although, these words are excluded in the topic-based text classification tasks due to lack of any semantic information in them, we believe these features will be effective in cross-domain AA as hinted by previous work (Goldstein-Stewart et al., 2009). Typically, words such as articles, prepositions, and conjunctions are considered as stop-words. We use a list of stop words publicly available for download (`www.webconfs.com/stop-words.php`).

Stylistic Features. Previous research has shown stylistic features to be effective in AA (Stamatatos, 2006; Bhargava et al., 2013). We use 13 stylistic features: number of sentences, number of tokens per sentence, number of punctuations per sentence, number of emoticons per document, percentage of words without vowel, percentage of contractions, percentage of total alphabetic characters, percentage of two consecutive punctuations, percentage of three consecutive punctuations, percentage of upper case words,

total parenthesis count, percentage of sentence initial words with first letter capitalized, and percentage of words without vowel.

Character n-grams. An n-gram is a sequence of n-contiguous characters. These features capture both the thematic as well as stylistic information of the texts, and hence have been proven to be very effective in previous AA studies (Keselj et al., 2003; Peng et al., 2003; Escalante et al., 2011). Since these features carry stylistic choices of the authors, we believe they will be stable across domains.

5 Experimental Settings

Following the training scenarios discussed previously in Section 3, we performed a set of experiments. We used 643 predefined stop-words. We considered as lexical features all words that were not stop words, and were among the 3,500 most frequent words occurring at least twice in the training data. We used 3,500 most frequent character 3-grams occurring at least six times in the training data.

Since dataset 1 is already balanced across authors, we used all the documents from this dataset. However, dataset 2 was originally imbalanced, therefore we chose at most ten documents per author to avoid a highly skewed distribution. In order to create a corpus like in the realistic scenarios of forensic investigations such as tweets, SMS, and emails, we chunked each selected document by sentence boundaries into five new short documents. This shortening of the documents increases the complexity of the task but enhances the practical applicability of our methods. We use these chunked versions for evaluating our proposed method. Splitting the documents in this way has been used in the past to deal with the lack of more documents per author(Luyckx and Daelemans, 2011; Koppel and Winter, 2014).

We obtained the performance measures using support vector machines (SVMs) implemented in Weka (Witten and Frank, 2005) with default parameters. We considered using SVMs because preliminary results showed this algorithm outperformed other reasonable alternatives. We used prediction accuracy as the performance measure to evaluate different training scenarios. Rather than just comparing the accuracies, we make most of the decisions based on statistical significance computed using two-tailed t-tests with 95% confidence interval.

All the experiments for cross-topic settings are carried out by controlling the genre. In the IT scenario, we computed the accuracy on each test topic using stratified 10-fold cross-validation. In the SCT scenario, for each test topic, prediction accuracy was computed by training separately on each remaining topic and averaging performances. We computed the accuracy on each test topic in the MCT scenario by withholding one topic as test topic and training on all other topics. For each training scenario, we computed one single score for each genre by averaging the accuracies across all test topics belonging to that genre.

6 Experimental Results and Evaluation

In this section, we report results and analysis on different experiments we carried out. We will start by showing empirically the challenge of cross-topic AA. Then, we will show results of our proposed approach.

6.1 Is Cross-Topic AA More Difficult than Intra-Topic AA?

Genre	Lexical Features			Stop-words			Stylistic Features			Character n-grams		
	IT	SCT	IT-SCT	IT	SCT	IT-SCT	IT	SCT	IT-SCT	IT	SCT	IT-SCT
Chat	25.71	13.11	96.11*	19.21	16.54	16.14*	**41.90**	27.49	34.39*	39.21	27.56	42.27*
Essay	26.58	5.92	348.99*	16.80	11.77	42.74*	15.66	14.56	7.02	**30.90**	13.28	132.68*
Email	19.80	6.22	218.33*	16.43	12.67	29.68*	**25.29**	24.4	3.52	24.94	14.52	71.76*
Phone Interview	37.62	10.29	265.6*	33.49	18.00	86.06*	33.02	16.16	51.06*	**56.99**	25.46	123.84*
Blog	22.18	6.32	250.95*	15.37	11.25	36.62*	13.16	11.31	14.06*	**25.38**	12.03	110.97*
Discussion	23.37	11.64	100.77*	23.37	16.31	43.29*	30.99	15.8	49.02*	**40.69**	25.28	60.96*

Table 2: Comparison of AA performance on IT and SCT scenarios on dataset 1. For each feature type, the IT and SCT columns indicate the accuracy (%) while the IT-SCT column is the relative gain of IT over SCT. For each genre, bold figures represent the best accuracy. Statistical significance is indicated by * in positive direction and by $^\flat$ in negative direction.

First of all, we want to understand if the cross-topic problem is more difficult than the intra-topic problem of AA. We compared the performance of the IT and the SCT scenarios using four types of features on various genres of dataset 1 as shown in Table 2. We clearly observed that for each genre, and for each feature type, the performance of the IT scenario is better than the SCT scenario and the difference is statistically significant. The only exceptions are 'Email' and 'Essay' genres for stylistic features. This is a strong indication that irrespective of the type of domain as well as the features considered, cross-topic AA is much more difficult than intra-topic AA.

6.2 Does Our Proposed Method Improve CTAA Performance?

We target to answer: *Can we reliably predict the author of a document written in one topic with a predictive model developed using documents from multiple other topics?* We carry out various experiments and compare the performance of our proposed MCT scenario with that of IT and SCT scenarios separately. Although, comparing MCT with only SCT would be enough to answer our research question and test our hypothesis, we are also interested in gaining more insights about cross-topic AA and understanding how it compares to IT, the simplest case of AA.

Genre	IT	SCT	MCT	MCT-IT	MCT-SCT
Chat	25.71	13.11	33.02	28.43^{*}	151.87^{*}
Essay	26.58	5.92	12.64	$-52.45^{\flat}$	113.51^{*}
Email	19.80	6.22	11.87	$-40.05^{\flat}$	90.84^{*}
Phone Interview	37.62	10.29	20.95	$-44.31^{\flat}$	103.6^{*}
Blog	22.18	6.32	13.15	-40.71	108.07^{*}
Discussion	23.37	11.64	25.26	8.09	117.01^{*}

(a) Lexical Features

Genre	IT	SCT	MCT	MCT-IT	MCT-SCT
Chat	19.21	16.54	33.49	74.34^{*}	102.48^{*}
Essay	16.80	11.77	22.06	31.31^{*}	97.08^{*}
Email	16.43	12.67	24.97	51.98^{*}	116.06^{*}
Phone Interview	33.49	18,00	38.89	16.12	115.67^{*}
Blog	15.37	11.25	20.43	32.92	81.6^{*}
Discussion	23.37	16.31	32.59	39.45^{*}	99.82^{*}

(b) Stop-words

Genre	IT	SCT	MCT	MCT-IT	MCT-SCT
Chat	41.90	27.49	37.62	-10.21	36.85^{*}
Essay	15.66	14.56	23.36	49.17^{*}	60.44^{*}
Email	25.29	24.4	33.12	30.96^{*}	35.74^{*}
Phone Interview	33.02	16.16	23.49	-28.86	45.36^{*}
Blog	13.16	11.31	15.67	26.29^{*}	38.55^{*}
Discussion	30.99	15.8	24.33	-21.49	53.99^{*}

(c) Stylistic Features

Genre	IT	SCT	MCT	MCT-IT	MCT-SCT
Chat	39.21	27.56	57.46	46.54^{*}	108.49^{*}
Essay	30.9	13.28	36.66	18.64	176.05^{*}
Email	24.94	14.52	36.53	46.47^{*}	151.58^{*}
Phone Interview	56.99	25.46	56.35	-1.12	121.33^{*}
Blog	25.38	12.03	33.41	31.64	177.72^{*}
Discussion	40.69	25.28	49.91	22.66^{*}	97.43^{*}

(d) Character n-grams

Table 3: Performance of lexical, stop-words, stylistic, and character $n-$gram features on dataset 1. The SCT, IT and MCT columns indicate the accuracy (%) while the MCT-SCT and MCT-IT columns present the relative gain of MCT over the other scenario. Statistical significance is indicated by * in positive direction and by $^{\flat}$ in negative direction.

MCT-SCT columns on Table 3 illustrate the statistical significance of MCT over SCT in a positive direction for all the genres. Using any type of feature in any genre, it is possible to significantly improve the performance of CTAA by training a machine learning algorithm using documents from all available out-of-domain topics. This serves as evidence to confirm our hypothesis and answer our research question that documents written in one topic can be reliably predicted with a model developed using documents from multiple other topics. This indicates that authors maintain a consistent writing style across topics.

In the MCT-IT column in Table 3(a), we can seen that the IT is significantly better than the MCT in three genres, while the MCT is better than the IT in only one. This is because lexical features directly capture the choices of authors in a certain thematic area, and hence they yield a good performance in the intra-topic setting. However, we observed contrasting and interesting patterns using stop-words, stylistic features, and character n-grams (MCT-IT column of Tables 3(b), 3(c), and 3(d)). MCT was better than IT, and the difference was significantly better, in 10 genres, while IT performance was significantly better than MCT in none of the genres. This is a very interesting finding as we observed that the cross-topic AA problem can be solved as effectively as the intra-topic AA problem using these features and a variety of topics.

Also using dataset 2, we found that for each type of feature, MCT is better than SCT, and the difference is statistically significant as shown in Table 4. This is another supporting evidence to our hypothesis. The small gain of IT over MCT suggests that our proposed approach is competitive even with the IT scenario.

Feature Type	IT	SCT	MCT	MCT-IT	MCT-SCT
Lexical Features	63.98	21.46	38.62	-39.64$^\flat$	79.96*
Stop-words	45.01	31.66	41.21	-8.44	30.16*
Stylistic Features	32.85	27.46	32.17	-2.07	17.15*
Character n-grams	75.08	45.87	64.54	-14.04$^\flat$	40.7*

Table 4: Performance of four types of features on three different training scenarios on dataset 2. For each feature type, the SCT, IT and MCT columns indicate the accuracy (%) while the MCT-SCT and MCT-IT columns present the relative gain of MCT over the other scenario. Statistical significance is indicated by * in positive direction and by $^\flat$ in negative direction.

6.3 Sensitivity of Features to Changes in Topic

We also want to demonstrate the behavior of four different feature types to changes in topic. We want to test if lexical features favor intra-topic AA and character n-grams favor cross-topic AA. Unlike lexical features, character n-grams carry stylistic choices of authors, and hence are expected to be robust across topics. In Table 2, for each genre, the relative gain of IT over SCT using lexical features is highest compared to that of stop-words, stylistic features, and character n-grams, thereby indicating that lexical features are more effective for ITAA than for CTAA. It is also apparent in Table 2 that the gain of characters n-grams is always better than that of stop-words and stylistic features. While looking at the performance on the SCT scenario using four features, it is observed that character n-grams give the best performance, while stop-words and stylistic features give the second best performance, which leaves lexical features at the bottom. This is because the first three features are topic-independent and hence were able to better discriminate among authors in cross-topic scenarios than lexical features. However, overall, character n-grams have the highest discriminative power in both IT and SCT, which confirms findings of earlier research (Stamatatos, 2013).

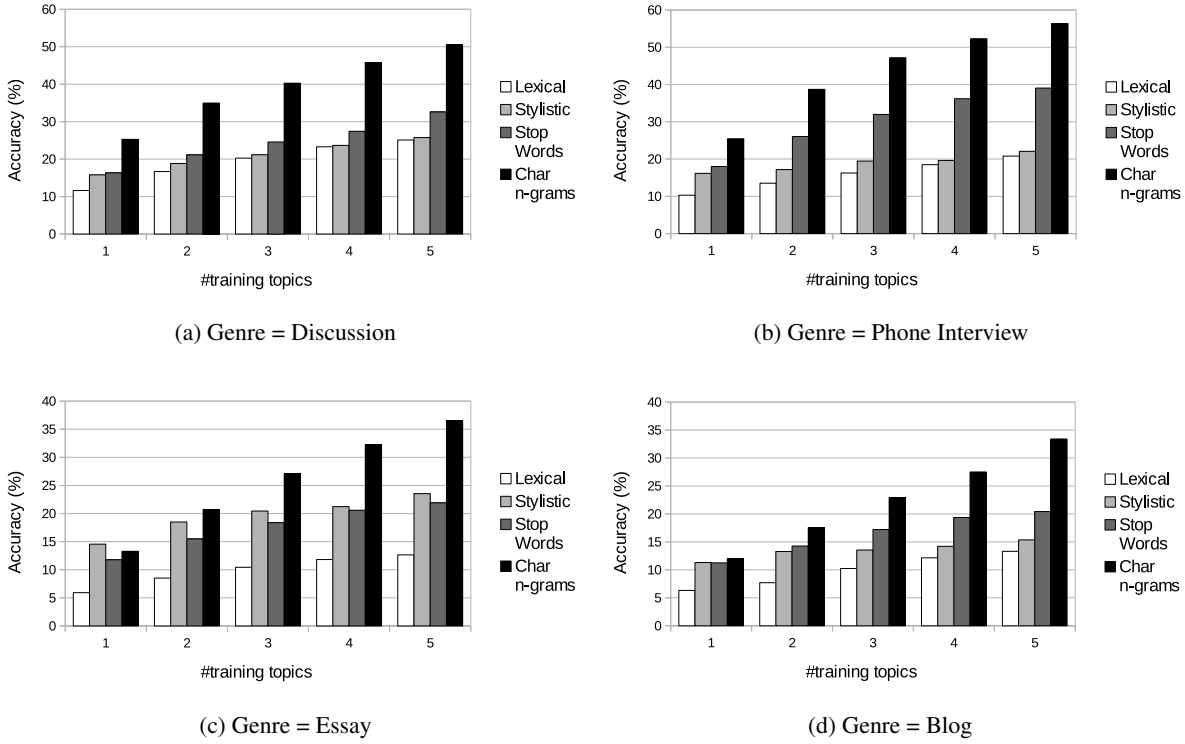
In Table 3, character n-grams, when compared to lexical features, stop-words, and stylistic features, yield the highest average relative gain on MCT over the SCT scenario (138.77%, vs 114.15% for lexical features, 97.41% for stop-words, 46.55% for stylistic features). Also, comparing the prediction accuracies of all four features separately in SCT, IT, and MCT scenarios, it is observed that character n-grams score best in most of the genres on each training scenario. This confirms that character n-grams have higher discriminative power in cross-topic AA than stop-words, stylistic features and lexical features.

For cross-topic AA, we observed that the accuracy across the board is not high. It is because the CTAA task is harder than other single domain classification tasks since the topics of the test data are fully disjoint with the topics of the training data. On top of that, the shorter document length makes it more challenging. The current system might not be production quality, but our findings will enable better models in the future that hopefully will be accurate enough to solve CTAA problems more effectively.

6.4 Cross-Topic AA with Varying Number of Training Topics

For traditional AA, it has been shown that around 10,000 word-tokens per author suffice as a 'reliable minimum for an authorial set' (Burrows, 2007). In our study, we have as few as 600 word-tokens per author, much less than the minimum size requirement stated by previous research. In this section, we look at how performance improves with increase in amount of training data by adding additional topics.

To explore this, we experimented by training on documents from all possible combinations of topics. In dataset 1, there are a total of six topics. Therefore, for each test topic, we experiment separately using one, two, three, four, and five topics for training. When measuring performance on k training topics, we gather all possible combinations of training on k of the five topics and then average the performance across all these combinations. For example, if we use two topics for training, then for each test topic, there are $\binom{5}{2} = 10$ possible training combinations that we then average to get a final score. We illustrate the results in Figure 1 for four genres using four types of features. Irrespective of the genres, topics, and types of features used, CTAA performance improves gradually with addition of more data. In most genres, this improvement seems to be almost linear with the number of topics trained on, suggesting that gathering more out-of-topic data should continue to improve the performance. We also observed that the character



(a) Genre = Discussion

(b) Genre = Phone Interview

(c) Genre = Essay

(d) Genre = Blog

Figure 1: Effect of training on varying number of topics in CTAA using lexical, stop-words, stylistic, and character n-gram features on dataset 1.

n-grams are the most effective author discriminator in cross-topic AA.

We performed a deeper analysis of the effect of individual topics, which is shown in Table 5. We took an initial topic as training data and then paired it with each of the other topics as additional training data and measured the average performance gain from the addition of the second topic. It is shown that across all genres, adding a second topic to the training data gives a character n-gram model greatest boost in performance than to a stop word or a stylistic or a lexical model. This is true regardless of the topics on which the model is trained. We do not observe negative transfer as in transfer learning (Pan and Yang, 2010) because in cross topic AA authors maintain styles across topics.

Initial Topic	**Genre = Chat**				**Genre = Email**			
	Lexical	**Stop-words**	**Stylistic**	**Character n-grams**	**Lexical**	**Stop-words**	**Stylistic**	**Character n-grams**
Sex Discrimination	5.85	5.57	1.67	10.33	2.24	7.29	8.86	9.72
Legalization of Marijuana	7.86	7.76	1.57	12.19	2.91	3.32	5.21	7.39
Catholic Church	6.24	8.76	6.24	14.33	2.41	4.48	3.59	5.22
Privacy Rights	5.9	4.66	1.9	14.05	2.97	6.45	4.6	10.06
War in Iraq	8.1	7.95	3.48	15.57	3.96	7.58	2.99	7.79
Gay Marriage	7.19	5.85	7.19	10.29	2.57	4.31	1.98	6.82

Table 5: Average performance gain from adding an additional topic as training data across different initial topics on dataset 1. Each value is the average accuracy gain after adding the second topic.

7 Is it Just 'More Data' that is Helping or is 'Diversity' Relevant?

The quantity of training data was not controlled in the experiments presented in Section 6, therefore, we performed some additional experiments where we did control for this. In Table 6, we present the comparison of SCT and MCT scenarios using the same amount of training data to understand whether the performance improvement in the MCT scenario is due to diversity or due to the fact of adding more data. We use dataset 1 to make this comparison. For the SCT scenario, for each test topic, we averaged

performance over three random samplings, where in each sampling we randomly selected four documents per author in each training topic. For the MCT scenario, for each test topic, we averaged performance

	Lexical Features			Stop-words			Stylistic Features			Character n-grams		
Genre	**SCT**	**MCT**	**MCT-SCT**	**SCT**	**MCT**	**MCT-SCT**	**SCT**	**MCT**	**MCT-SCT**	**SCT**	**MCT**	**MCT-SCT**
Chat	12.24	13.94	13.89^{*}	14.37	16.35	13.78^{*}	26.52	28.52	7.54^{*}	24.39	25.17	3.2^{*}
Essay	9.11	11.3	24.04^{*}	12.43	14.12	13.6^{*}	21.35	22.93	7.4^{*}	18.37	19.58	6.59^{*}
Discussion	9.65	10.52	9.02^{*}	12.93	13.7	5.96^{*}	19.57	20.85	6.54^{*}	19.84	21.48	8.27^{*}
Email	8.84	9.98	12.9^{*}	12.48	13.91	11.46^{*}	20.89	21.92	4.93^{*}	17.91	20.76	15.91^{*}
Phone Interview	8.94	10.84	21.25^{*}	14.65	17.67	20.61^{*}	19.73	20.94	6.13^{*}	18.84	26.35	39.86^{*}
Blog	8.45	9.66	14.32^{*}	12.78	14.05	9.94^{*}	18.53	19.62	5.88^{*}	17.58	19.95	13.48^{*}

Table 6: Comparison of MCT and SCT scenarios on controlled training data using four types of features on dataset 1. For each feature type, the SCT and MCT columns indicate the accuracy (%) while the MCT-SCT columns present the relative gain of MCT over the SCT. Statistical significance is indicated by * in positive direction and by $^{\flat}$ in negative direction.

over three random samplings, where in each sampling we randomly selected four training topics. For each selection of four training topics, we averaged performance over three random samplings where in each sampling we randomly selected one document per author in each training topic. Thus, we ended up with the same number of documents for training both models. Even with the same amount of training data, training on documents from different topics is better than training on documents from a single topic, with statistically significant performance gains ranging from 3.2% to 39.86% as shown in Table 6. This demonstrates that data from a diverse set of topics will still give a boost in performance and is always significantly better than using data from the same topic.

8 Related Work

The majority of the work in authorship attribution deals with single-domain datasets. However, there have been a handful of studies that add some cross-topic flavor in the AA task (Mikros and Argiri, 2007; Goldstein-Stewart et al., 2009; Schein et al., 2010; Stamatatos, 2013). Mikros *et al.* (2007) concluded that many stylometric variables are actually discriminating topic rather than author and their use in AA should be done carefully. However, the study was performed on a single corpus containing only two authors in two topics that raises questions on reliability of their conclusions. Stamatatos (2013) illustrated the effectiveness of character n-grams in cross-topic AA. It was also shown in that study that avoiding rare features is effective in both intra-topic and cross-topic AA. However, all these conclusions came from training an SVM classifier in only one fixed topic. In contrast, in our paper, we draw our conclusions from all possible training/testing combinations rather than fixing in advance the training topic.

Goldstein-Stewart *et al.* (2009) also carried out some cross-topic experiments by concatenating the texts of an author from different genres. This experimental setting results in a corpus where each test document contains a mix of genres, which is not representative of real world AA problems. Still, to provide some comparisons to the work of Goldstein-Stewart *et al.* (2009), we concatenated all the texts in dataset 1 produced by an individual on a single topic, across all genres to produce one document per author on each topic. We compare our results with those reported in the paper under same training/testing conditions. We withheld one topic and trained on documents from the other five topics.

Test Topic	**Lexical**	**Stop-words**	**Stylistic**	**Character n-grams**	**Stop-words + Character n-grams**	**Previous Work**
Sex Discrimination	66.67	76.19	33.33	95.24	**95.24**	95
Catholic Church	76.19	95.24	38.10	95.24	**100**	95
Gay Marriage	80.95	80.95	42.86	90.48	90.48	**95**
Legalization of Marijuana	52.38	66.67	33.33	95.24	**100**	100
Privacy Rights	42.86	52.38	28.57	95.24	90.48	**100**
War in Iraq	57.14	71.43	38.10	100	**100**	81
Average	62.7	73.81	35.72	95.24	**96.03**	94.33

Table 7: Comparing performance of our work with previous work in the same training/testing setting. The results in the last column were obtained from Goldstein-Stewart *et al.*(2009). For each test topic, the bold figure represents the best performance.

The last column of Table 7 presents the results obtained by using the combination of stop-words and 88 Linguistic Inquiry and Word Count (LIWC) features as reported in Goldstein-Stewart *et al.* (2009). We observed that the combination of character n-grams and stop-words, on average, performs better than those reported in the paper. On this fixed training/testing scenario, we see better accuracies, as high as 100%, across the board. This is because, in this experiment, each training sample on average was ≈ 25 times longer than the training sample in our chunked versions. This illustrates that authorship attribution of short documents, as in our chunked versions, is a challenging task, but we believe it resembles a more realistic scenario of forensic investigations.

9 Conclusions and Future Work

In this research, we presented the first comprehensive study with rigorous analysis on cross-topic AA. Although previous work had hinted some of our findings, it was based on very limited experiments (using only one fixed topic for training). We investigated CTAA using all possible combinations of topics to draw more robust and stable conclusions. We first illustrated the difficulty of cross-topic AA by comparing its performance with intra-topic AA using different types of features. We demonstrated that a framework trained on documents belonging to thematic areas different than that of the documents under investigation statistically improves the performance of cross-topic AA. This improves the ability of the model to find the authors of documents belonging to a new topic not present during the training of the model. By controlling the training data, we demonstrated that training on diverse topics is better than training on a single topic confirming that MCT not only benefits from more data but also from a thematic variety. We also showed a statistical analysis that lexical features are closer to the thematic area and hence were an effective author discriminator in intra-topic attribution. Similarly, character n-grams prove to be a very powerful feature especially in a condition where training and test documents come from different thematic areas. Although intra-topic AA is easier than cross-topic AA, our proposed model for CTAA achieves performance close or in some cases, better than that of an intra-topic AA model. Another interesting conclusion of our study is that addition of more training data from any topic, no matter how distant or close it is with the topic of documents under investigation, improves the performance of CTAA for all types of features. We believe that our contribution to cross-topic AA will be generalizable to other classification problems too.

In the future, we plan to explore the cross-genre problem of AA that is critical for tasks like linking user accounts across emails, blogs, and other social media. Our proposed CTAA approach can be directly applied to the cross-genre problem but we may discover different feature behavior in this scenario. We also plan to explore domain adaptation and transfer learning techniques to solve CDAA problems.

Acknowledgements

This research was partially supported by ONR grant N00014-12-1-0217, NSF award 1254108, and NSF award 1350360. It was also supported in part by the CONACYT grant 134186 and the WIQ-EI IRSES project (grant no. 269180) within the FP 7 Marie Curie.

References

A. Abbasi and H. Chen. 2008. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Trans. Inf. Syst.*, 26(2):7:1–7:29, April.

M. Bhargava, P. Mehndiratta, and K. Asawa. 2013. Stylometric analysis for authorship attribution on twitter. In *Big Data Analytics*, volume 8302 of *Lecture Notes in Computer Science*, pages 37–47. Springer International Publishing.

J. Burrows. 2007. All the way through: Testing for authorship in different frequency strata. *Literary & Linguistic Computing*, 22:27 – 47.

R. María Coyotl-Morales, L. Villaseñor Pineda, M. Montes-y Gómez, and P. Rosso. 2006. Authorship attribution using word sequences. In *Proceedings of the 11th Iberoamerican Conference on Progress in Pattern Recognition, Image Analysis and Applications*, CIARP'06, pages 844–853, Berlin, Heidelberg. Springer-Verlag.

O. de Vel, A. Anderson, M. Corney, and G. Mohay. 2001. Multi-topic e-mail authorship attribution forensics. In *Proceedings of the Workshop on Data Mining for Security Applications, 8th ACM Conference on Computer Security*.

J. Diederich, J. Kindermann, E. Leopold, and G. Paass. 2003. Authorship attribution with support vector machines. *Applied Intelligence*, 19:109–123, May.

H. J. Escalante, T. Solorio, and M. Montes-y Gómez. 2011. Local histograms of character n-grams for authorship attribution. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 288–298, Portland, Oregon, USA, June. Association for Computational Linguistics.

G. Frantzeskou, E. Stamatatos, S. Gritzalis, and C. E. Chaski. 2007. Identifying authorship by byte-level n-grams: The source code author profile (SCAP) method. *Journal of Digital Evidence*, 6(1).

J. Goldstein-Stewart, R. Winder, and R. Evans Sabin. 2009. Person identification from text and speech genre samples. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 336–344, Stroudsburg, PA, USA. Association for Computational Linguistics.

J. Houvardas and E. Stamatatos. 2006. N-gram feature selection for authorship identification. In J. Euzenat and J. Domingue, editors, *AIMSA 2006*, volume 4183 of *LNAI*, pages 77–86.

A. Kaster, S. Siersdorfer, and G. Weikum. 2005. Combining text and linguistic document representations for authorship attribution. In *SIGIR Workshop: Stylistic Analysis of Text for Information Access*, pages 27–35.

V. Keselj, F. Peng, N. Cercone, and C. Thomas. 2003. N-gram based author profiles for authorship attribution. In *Proceedings of the Pacific Association for Computational Linguistics*, pages 255–264.

M. Koppel and Y. Winter. 2014. Determining if two documents are written by the same author. *Journal of the Association for Information Science and Technology*, 65(1):178–187.

M. Koppel, J. Schler, and S. Argamon. 2011. Authorship attribution in the wild. *Language Resources and Evaluation*, 45:83–94.

K. Luyckx and W. Daelemans. 2011. The effect of author set size and data size in authorship attribution. *Literary and Linguistic Computing*, 26(1):35–55.

G. K. Mikros and E. K. Argiri. 2007. Investigating topic influence in authorship attribution. In *Proceedings of the International Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection*, pages 29–35.

S. Jialin Pan and Q. Yang. 2010. A survey on transfer learning. *IEEE Trans. on Knowl. and Data Eng.*, 22(10):1345–1359, October.

F. Peng, D. Schuurmans, V. Keselj, and S. Wang. 2003. Language independent authorship attribution using character level language models. In *10th Conference of the European Chapter of the Association for Computational Linguistics, EACL*, pages 267–274.

A. I. Schein, J. F. Caver, R. J. Honaker, and C. H. Martell. 2010. Author attribution evaluation with novel topic cross-validation. In *The 2010 International Conference on Knowledge Discovery and Information Retrieval*, Valencia, Spain, October.

E. Stamatatos. 2006. Authorship attribution based on feature set subspacing ensembles. *International Journal on Artificial Intelligence tools*, 15(5):823–838.

E. Stamatatos. 2011. Plagiarism detection using stopword n-grams. *Journal of the American Society for Information Science and Technology*, 62(12):2512–2527.

E. Stamatatos. 2013. On the robustness of authorship attribution based on character n-gram features. *Journal of Law & Policy*, 21(2):421 – 439.

I. H. Witten and E. Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kauffmann, 2nd edition.

Y. Zhao and J. Zobel. 2005. Effective and scalable authorship attribution using function words. In *Proceedings of 2nd Asian Information Retrieval Symposium*, volume 3689 of *LNCS*, pages 174–189, Jeju Island, Korea.

R. Zheng, J. Li, H. Chen, and Z. Huang. 2006. A framework for authorship identification of online messages: Writing-style features and classification techniques. *J. Am. Soc. Inf. Sci. Technol.*, 57(3):378–393, February.

Online Gaming for Crowd-sourcing Phrase-equivalents

A. Kumaran
Microsoft Research
Bangalore, India
a.kumaran@microsoft.com

Melissa Densmore
University of Cape Town
Cape Town, South Africa
mdensmore@acm.org

Shaishav Kumar
Microsoft Research
Bangalore, India
v-shaisk@microsoft.com

Abstract

We propose the use of a game with a purpose (GWAP) to facilitate crowd-sourcing of phrase-equivalents, as an alternative to expert or paid crowd-sourcing. Doodling is an online multi-player game, in which one player (*drawer*), draws pictures on a shared board to get the other players (*guessers*) to guess the meaning behind an assigned phrase. In this paper we describe the system and results from several experiments intended to improve the quality of information generated by the play. In addition, we describe the mechanism by which we take candidate phrases generated during the games and filter out true phrase equivalents. We expect that, at scale, this game will be more cost-efficient than paid mechanisms for a similar task, and demonstrate this by comparing the productivity of an hour of game play to an equivalent crowd-sourced Amazon Mechanical Turk task to produce phrase-equivalents over one week.

1 Introduction

While it is fairly well known when individual words have the same meaning, it is far more difficult to determine when phrases or even sentences carry the same basic idea. While it might be possible to address this task with machine learning techniques, building a corpus of sentences from which to seed a database requires human intelligence. We suggest a *game with a purpose* (GWAP) that will serve to generate phrases with similar meanings, while simultaneously providing meta-information about the quality of the match. In this drawing game, called *Doodling*, individuals compete in groups to guess the meaning behind a given drawing that is being drawn by one designated *drawer* trying to convey a given phrase or a short sentence. The designated drawer decides when a guessed phrase matches the source phrase. For example "*How far is the airport*?" might match semantically "W*hat is the distance to the airport*?" In addition, the drawer can indicate for each partial guess how close it is on a scale of 1-3 to help the guessers converge on phrases that will match the given phrase or sentence. We then pass all of the guesses and annotations through an SVM classifier to automatically identify potential phrase-equivalents. In this study we examine several techniques for using this system to generate high quality data while also making the game more enjoyable. We measure the efficacy of each technique by comparing our results to a gold standard: using human evaluators to rate the phrase matches generated through the game manually. We also compare Doodling to a paid crowdsourcing paradigm – Amazon's Mechanical Turk – to source phrase equivalents for the same set of phrases, and we show that our approach might be cost effective for large scale sourcing of paraphrases of equivalent quality.

2 Background

In this section we define the problem we are trying to address, and discuss the various ways it has been approached in the past.

2.1 Phrase-equivalents & Evaluation of Quality

In this paper, we define phrase equivalents (PEs) as text elements – phrases or short sentences – that have same or similar semantic content, but with surface structure different from each other. PEs are similar to *paraphrases*, but broader in scope, inclusive of partial matches in meaning as well as complete

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1238–1247, Dublin, Ireland, August 23-29 2014.

paraphrases. PEs are useful for many NLP systems from simple language modelling and smoothing, to complex Machine Translation technology for generation of a surface form in the target language. Most existing corpora are hand-created, and hence they tend to be small in size, and available only in limited languages and domains. Other data driven approaches – such as, creation of paraphrases using monolingual machine translation (Quirk et al., 2004), mining inference rules from text corpora (Lin & Pantel, 2001), or paraphrase extraction from parallel corpora (Dolan et al., 2004) (Barzilay & McKeown, 2001) – were shown to be effective, but such approaches require significant seed corpora which are available only in limited domains and languages. In addition, the (Lin & Pantel, 2001) approach can generate equivalents using user defined patterns, and may not be appropriate for generating loosely related conceptual paraphrases like the human generated ones that Doodling may generate.

The criteria used for evaluating phrase equivalents differ vastly in research literature, ranging from conceptual equivalence (Barzilay & McKeown, 2001), to interchangeability (Ibrahim et al, 2003), to preservation of grammatical correctness and semantic equivalence (Callison-Burch, 2005), and the standard metric of BLEU score (Callison-Burch, 2005; Papineni et al., 2002). In general, there is no accepted standard model for measuring quality, hence we adopted manual annotation by experts.

2.2 Crowdsourcing & Games with a Purpose (GWAP) for Computational Linguistics

Many flavors of crowdsourcing paradigms exist for the creation of language data. From the *for-pay* model where the contribution is for monetary rewards (Callison-Burch, 2009; Irvine & Klementiev, 2010; Chen & Dolan, 2011), to the *for-recognition* model, where the contribution is made for individuals' visibility in a community (e.g., SourceForge), and the *common-good* model, value is produced for the benefit of some community (Kumaran et al., 2009). In this paper, we explore the *for-fun* model (Cooper et al., 2010; Law et al., 2007; Von Ahn & Dabbish, 2004; Von Ahn et al., 2006), in which data is a by-product of some gameplay, often referred to as "Games with a Purpose" (Von Ahn & Dabbish, 2008), which have been shown to be very successful in many domains.

Specifically with respect to generation of paraphrases or phrase equivalents, (Chen & Dolan, 2011) present their paraphrase collection using video annotations, focusing primarily on viability of establishing Mechanical Turk for providing paraphrases in a productive way. (Barzilay & McKeown, 2003) posited that multiple translations of a foreign text may be a naturally occurring source for paraphrases as each is authored by a different translator; our approach is analogous to this approach, though our source phrases/sentences are not from a foreign language. (Chklovski, 2005) presents an online paraphrase collection tool and studies the incentive model for responsible contributions by volunteers. Paraphrases generated by Doodling would be similar to paraphrases labelled under class "Phrasal" and to a lesser extent class "Elaboration" in (Chen & Dolan, 2011). In our earlier work (Kumaran et al., 2012) we focused on a proof-of-concept methodology using a Pictionary-based approach for generation of paraphrases. In this paper, we expand our concept for generating phrase equivalents in scale inexpensively, using several game and UI/UX features, and also compare it with a realistic for-pay baseline using Mechanical Turk. The power of our methodology is its self-verification mechanisms (by drawer annotating the response for convergence, and the final acceptance) that validates the generated paraphrases.

3 Doodling as a Game

In this section, we present the design elements and the game flow of the Doodling game.

3.1 Game Design

In the Doodling game, the games are played in rooms with one player (designated as the *Drawer*) sketches an assigned concept - as phrase or sentence - while other players in the game room (*Guessers*) attempt to guess the assigned concept from the drawing that is being replicated to all screens. The Guessers typically start guessing the words first (based on the concept that the Drawer starts sketching on the screen); while the game will automatically indicate exact partial matches (for example, "Taxi" as a guess for the given phrase, "Taxi Driver"), the drawer also has the ability to provide feedback using annotations. The Drawer may annotate partial guesses as incorrect (red), on the right track (orange), or partially correct (green), to guide the convergence. All the guessers' guesses and the drawer's annotation are broadcast to all the players in the room. Such broadcasting provides a mechanism in which players

can build on the top of other's guesses, gradually building up the phrase or the sentence. At some point, if one of the guessers guess the right phrase exactly, the game is closed automatically. In addition, if the drawer judges the guess as having the same meaning as the assigned concept (for example, "Cabbie" for "Taxi Driver"), he/she can end the round by marking the guess as correct, rewarding the guesser with game points. If the timer runs out before a correct guess happens, then the game times out. Figure 1 shows the UI during the progress of a game (the given text element being "*taxi driver*").

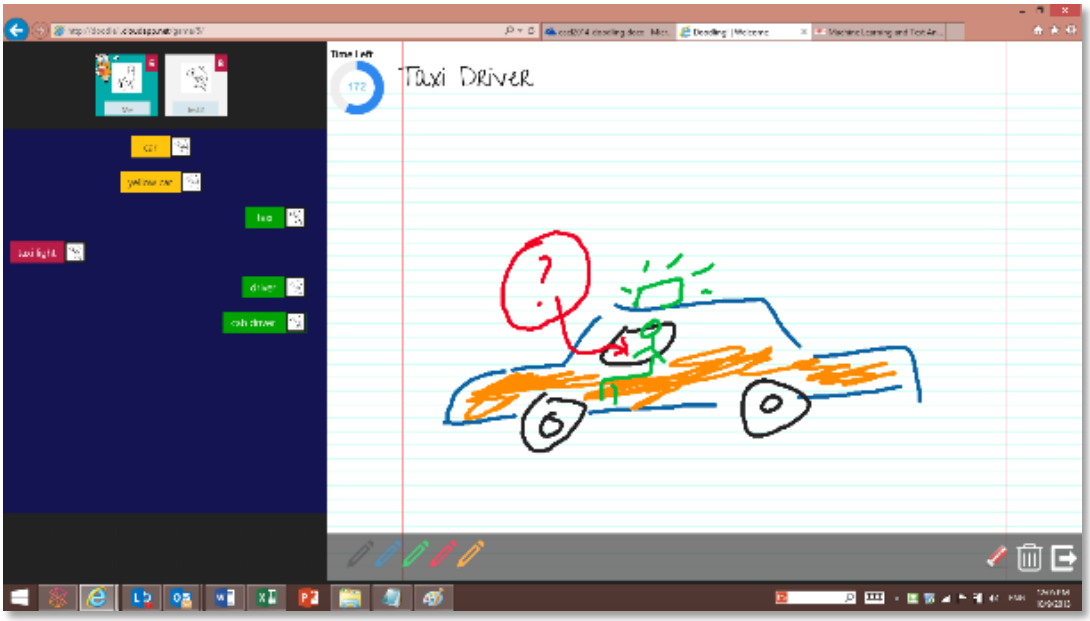


Figure 1: Doodling Game

Our primary intuition is that the sketches provide a language-independent means of communication of concepts that is effectively employed for the generation of phrase equivalents. Thus, we leverage a fun drawing-guessing game to fulfill the linguistic purpose of generating phrase-equivalents. An important aspect of making Doodling effective was to make it engaging to play. We underwent multiple user studies followed by changes to the game's UI/UX. Earlier trials had revealed the need for additional feedback from the drawer, leading to the introduction of 3-stage annotations of guessed phrases. From a usability standpoint, the UI and gestures were optimized for use with touchscreen capable devices, including of the use of swipe gestures for annotating incoming guesses.

The Doodling game subscribes to the Inversion Problem (Von Ahn & Dabbish, 2008), where one of the players produces an output in the form of a sketch for a given input phrase. The other players attempt to guess the given input. The game may produce multiple surface forms of a single semantic intent that have a relationship similar to that of the input-output pair in the "noisy-channel" model.

3.2 Game Elements

While the game dynamics promote the resolution of the underlying computational problem (i.e., the generation of phrase equivalents), we made certain modifications to the basic *sketch-and-convey* metaphor – in the formation and constitution of the game rooms, in the assignment of roles to players in a round-robin fashion, and the drawer's feedback using annotation, in exposing every player's guess to the entire game room, and the winning strategy that encourages building on each other's guesses – in order to help the rounds finish successfully, converge faster, and be more competitive. Above all, the game dynamics and the UI were designed to make Doodling enjoyable as a game.

Roles: Users may join existing game rooms, or can create a new private game room after logging in to the Doodling portal. In a game room, one of the users is assigned – randomly – the role of Drawer (D), and the others the role of Guessers (G). At the end of a given game round, the role of drawer cycles among the game room participants. All G's both compete (the first guesser to guess right – either fully or partially – is rewarded), as well as collaborate (each builds on other's guesses to build longer phrases for bigger rewards) in guessing the text element being conveyed by the D.

Game Round: Like the sketches, the individual guesses of a given G are broadcast to the entire room, along with any annotation from D on each of the guesses (red/orange/green). While the right guesses (either lexicographic match, or as judged by D) gives the game point to the specific G, the broadcast of guesses and feedbacks from D to the entire game room provides a transparent mechanism to help each player build on the guesses of the others. The game round closes with exact reproduction of the source phrase by one of the G's, or by D accepting a full semantic equivalent by double tapping a tile. As an incentive for the role of the drawer, the D is also rewarded with some game points.

Data: In our current experiments, we used standard phrases from a generic WikiTravel (http://wikitravel.org/en/wikitravel:phrasebook template) tourism phrase book as input elements. The authors subjectively classified each text element as Easy or Hard, depending on the potential difficulty to express it as a sketch; though such annotation implies additional preparatory work, it may be well worth the investment as such tagged corpora forms the seed for many variations. We plan to add text elements in many domains (Celebrities, Movies and Idioms), to provide diversity to the players.

Text Element	Diff.	Granularity
Cheese Omelette	Easy	Phrase
Museum of Modern Art	Hard	Phrase
I would like a bowl of soup.	Easy	Sentence
I am not feeling well!	Hard	Sentence

Table 1: Sample of text elements used in the initial seed corpus

In order to understand the dynamics of the game, and to improve the quality and quantity of the phrase equivalents generated in Doodling, we incorporated many features.

Number of Players: The application supports 2-4 players per game room, to measure the effect of room size on convergence rate and the player enjoyment. We hypothesize that those game rooms with more players will lead to better completion primarily due to higher productivity in phrase generation.

Hints & Reminders: we provided hints to all guessers at the beginning of the game to prime then on what to expect about the guess phrase. Hints are simple text elements, such as "Short Phrase" or "Hard Sentence", etc. In addition, we also provided some reminders periodically for improving the game dynamics, especially for the new players, including a reminder to the drawer that they can accept non-exact phrases with the same meaning by double-tapping on the guess tile. Reminders appear on the screen, and fade away unobtrusively. Some game rooms were provided the hints, while others are not, in order to measure how helpful the hints are for game completion.

Soft Matches: Exact lexicographic guesses (full or partial) are automatically rewarded by the game engine. However, as the primary mechanism for gathering paraphrases, soft matches were allowed and rewarded at the discretion of the drawer (either by the double-tap action that accepts a guess as a correct phrase equivalent, or by the swipe-right action that which indicates a potential partial match). Yet, to discourage collusion or cheating, a reporting mechanism is provided: The final accepted guess along with the input text element are shown to all participants, to report any unsatisfactory acceptance.

Metrics: For measuring the effectiveness of the Doodling game, we define many metrics ranging from completion statistics (completion rate and completion time), to quality by comparison with gold data (true positives as compared with user-annotated data, precision and accuracy of automatically classified data), to qualitative user feedback (fun factor).

4 Doodling: Experimental Evaluation

Doodling is an HTML5 app that is accessible from most devices - touchscreen laptop or tablets - and deployed in the cloud (http://doodle1.cloudapp.net/). After deployment, we recruited volunteers (primarily graduate students) to log in and play the game for one hour. As the volunteers entered the game server, they were assigned to different game rooms; each room was instrumented for a specific configuration (game room size - between 2 and 4 players, and availability of hints and reminders). Each room was given the same set of 38 phrases in the same sequence, to keep the variability to a minimum. After an hour, the games were closed and the players asked to fill in an online questionnaire.

In these trials, the 14 volunteers played a total of 112 games, in different game rooms. Most players had previously been exposed to the *sketch-and-convey* metaphor through Pictionary-type games.

4.1 Quality of the Generated Data

Basis for evaluation: We first extracted all of the text elements annotated as a potential match (green, orange, or winning) by the Drawer. Each of the three authors then independently classified according to the relevance of the match. The following five classes were used for annotating every annotated text element: EF (Exact Full Match), EP (Exact Partial Match), TF (True Full Match), TP (True Partial Match) or NM (Not a Match). Partial matches entailed guesses which captured some sub-element of the seed text, but not the entire meaning. We then measured inter-annotator agreement of author's annotations using a Fleiss Kappa measure (Fleiss, 1971), which stood at 0.7424, indicating substantial agreement among our annotations. Hence, we used our annotation (using majority voting for resolving any conflicts) as the gold data set for validating automatically the user generated paraphrases, in subsequent sections.

Quality of the generated data: Of the 112 games played, 98 of them completed successfully. Games were considered incomplete if the timer expired before successful completion. Of the 98 completed games, 15 of the final guesses were false positives (i.e. NM, wrong answers accepted erroneously), 42 games closed with guessers reproducing the exact text element given to the drawer (i.e. EF), and 19 games closed with Drawer correctly accepting a guess that is semantically equivalent to the given text element (i.e. TF, a true phrase-equivalent), and the remaining producing various degrees partial semantic matches (i.e. TP, true partial phrase-equivalents). The average time of completion for successfully completed games was 160 seconds.

In addition, most of the games, irrespective of whether closed correctly or not, produced partial equivalents to the given text element as intermediate guesses, thus providing valuable data for research. These include all the potential matches which were not accepted as the final answer for a game, but were marked as green or orange via the drawers' swipe-based annotation. Table 3 shows the breakdown of the gold classification of all of the potential matches.

4.2 From Game to Corpus

Once assured the quality of the generated data, we devised a methodology for automatically detecting phrase equivalents (full or partial) from the user generated data, so that the game would be able to scale without the need for human annotators to verify individual guesses. We designed a classifier for automatically validating phrase equivalents (partial or full), based only on the game meta data, and very shallow text level features, and not based on any linguistic (such as, dictionaries, thesauri, etc.) or other specialized corpora (such as, parallel or paraphrase corpora). Our basic premise is that if such a classifier can identify good paraphrases with simple features, then we will be able to identify the phrase equivalents automatically, in new domains or languages.

Our classifier uses only simple game and text-level features: hardness of the input text element (easy/medium/hard), status of completion flag and cheating flag at completion, order and time of the guess, drawer's annotation (green/orange/red), cross-game evidence, substring similarities to the input text element and orthographic overlap with the input text. First, we extracted any exact matches (EF or EP) by removing any text elements that were a substring of the original guess, leaving us with a training corpus of 122 potential phrase-equivalents. We trained the classifier using a 5-fold cross-validation this corpus. Some paraphrases thus extracted are shown in Table 2.

Source phrase	**Paraphrases extracted**
Police Officer	Policeman, Police Inspector, Police Superintend
I lost my luggage.	I need to find my bag at the lost-and-found counter, Lost-and-found luggage counter.
School Teacher	Class Teacher, Teacher teaching in school.
Railway Station	Railroad Station, Railway Platform

Table 2: Automatically Extracted Paraphrases

	Doodling	Doodling + SVM			MTurk
	Raw Corpus	Training Corpus	SVM = NM	SVM = TF|TP	Corpus
Size	234	122	73	49	92
Exact Full (EF)	42	EF and EP Data automatically removed from Corpus using String and Substring Match			0
Exact Partial (EP)	71				21

True Full (TF)	30	30	2	28	53
True Partial (TP)	11	11	5	6	13
Not a Match (NM)	81	81	66	15	5
Precision (TF+TP/Size)	17%	34%	10%	69%	72%

Table 3: Comparison of corpora produced by Doodling and MTurk to gold data. SVM numbers are an average of the results generated during the 5-fold cross-validation.

The classifier reduces the burden on expert hand-annotators, by automatically filtering out text elements that are likely to not be a match. As can be seen in Table 3, only 17% of the raw corpus constitutes useful data. Removing exact and substring matches (EF and EP) increases the precision to 34%. The usable corpus produced by the classifier (SVM=TF|TP) has a precision of 69%, with only 10% of the remaining corpus (SVM=NM) constituting false negative, or "lost" data. The overall accuracy of the classifier (% of true positives + true negatives) is 82%.

This methodology provides a viable means of generating paraphrase corpora, with a small amount of hand-crafted corpus in a new domain. The classifier can be fine-tuned either for accuracy of prediction (precision) or productivity (recall); in our experiments we fine-tuned it for precision. Also, we believe that given that these features used are devoid of linguistic or domain information, our results may provide a lower bound on the quality of automatic identification of phrase equivalents; this may be improved substantially by use of appropriate linguistic resources or specialized corpora.

In addition to phrase-equivalent data, many of the guesses relate semantically to the input text element, in varying degrees. Using similar features as used in the classifier, the annotation data can be used for identifying sets of related words for given input text elements, creating valuable resources for search query expansion.

5 Mechanical Turk Experiments

To understand the quantitative difference between Doodling and a paid crowdsourcing model for generating paraphrases we designed a "Data Collection" Mechanical Turk task using the same phrases that were used in our user experiments. Based on previous work relating to designing of Turk experiments and accepted best practices, we kept the task description simple: Each task asked a respondent to generate five unique and semantically equivalent phrases for a given source phrase. The respondents were chosen based on their familiarity with English as their first language, and each phrase was to be annotated by 20 respondents over one week duration; this duration was chosen to keep the respondent population size roughly equal to that of our user experiment. Reward for completing the generation of five phrase-equivalents for a single given phrase was fixed at $0.10USD, in line with the rewards given out for tasks with similar levels of difficulty as cited in published literature (Callison-Burch et al. 2009; Dolan et al, 2011). Though the time frame was a larger than the duration of our experiments (one hour) significantly, the overall time taken for task is comparable to the time spent in gameplay.

At the end of the one-week duration of the experiment, 14 out of 38 phrases got at least one set of valid paraphrases, leading to a completion percentage of 37%. Most of the submitted phrases were annotated only by one respondent; the average number of respondents per phrase was 1.23. The annotation data was judged by the authors in the same scale as outlined in Section 5.1, and the Fleiss Kappa measure for the annotation was 0.74, signifying significant agreement between their judgments. Overall, 72% of the MTurker generated paraphrases were accepted as full or partial alternatives (See Table 3). While the quality of data is very good, any misunderstanding of the task generated results that are significantly off the mark: For example, "*How do I get to the nearest international airport?*" was generated for "*International Airport*" as the source phrase. Since the participation and completion was low, we extended the duration of the task by another week, but the second week yielded only 2 additional completed tasks indicating that the duration of the experiment was not the sole factor in the relative low rate of task completion; perhaps it is the nature of the task that did not attract significant participation.

6 Discussions

6.1 Viability of Doodling as a Game

The 85% successful completion (98 out of 112) of the games is encouraging, and indicates the viability of the game to complete successfully. At the end of the experimental session (wherein 30 rounds of the game had been completed by each player on an average), the players were asked to fill in an online survey to measure various qualitative metrics on effectiveness of Doodling as a game. A wide variety of questions were asked, ranging from specific input (*How did [a specific feature] affect your ability to guess the right phrase?*) to generic qualitative measures (*Would you play this game again?*). Among the questions were three specific questions on how much the players enjoyed the game as a drawer, as a guesser and overall, in a scale of 1 (*Hated it*.) to 5 (*Loved it*!). From the 10 respondents, the enjoyment factor averaged at 4.7 overall. Such high score validates the game design and UX as a viable mechanism for an enjoyable game. Further, 9 out of 10 respondents said that they would *definitely* play the game again, with comments such as "It was very interesting and fun" and "This game is kind of addictive", indicating attraction of the game for subsequent engagement

6.2 Use of Hints & Reminders

We find no evidence for the hints or reminders to be valuable either in improving the quality of the result, or helping the time for convergence/completion. We note that several gamers resorted to other means of indicating the structure of the guess phrases, such as drawing out a number of dashes to indicate the size of the guess phrase, with some of them requesting us to do the same.

6.3 Scaling Up: Comparison with Mechanical Turk for crowd-sourcing phrase equivalents

GWAPs have been criticized for their complexity, long time-to-market, and hidden running costs (Wang et al., 2012). Paid crowd-sourcing methods, by comparison, are simpler to set up, and have lower initial costs. While a concrete, direct comparison is not possible, Table 3 lays out some of the differences between the two methods, especially with reference to our metrics.

	Mechanical Turk (MT)	Doodling
Experimental Operating Costs	US$82	US$90
Ongoing Costs	US$0.10/source phrase	US$90/month
Setup Costs	Minimal	3 man-months
Players/Workers	9	14
Time	2 weeks	1 hour
Completion (Games with ≥ 1 TF generated)	14/38 (37%)	38/38 (100%)
Quantity (# of Unique TFs)	53	28
Precision (% of usable data)	72%	69% (with Classifier)

Table 4: Comparison of MTurk and Doodling experiments for generation of phrase-equivalents

In the case of the Doodling game, the development of the game took 3 man-months, while Mechanical Turk's (MT) setup time was minimal. Both the Doodling and MT experiments had similar operational costs, at US$90 and US$82 respectively. This cost of $90 for Doodling consists of hosting and bandwidth charges incurred for two virtual servers running on a commercial cloud platform. However, once we scale Doodling up to permit more users and higher productivity, we expect the costs to remain fixed, whereas MT costs will scale proportionally to the productivity at US$0.10 per source. In addition, even with approximately equal investment, one hour of Doodling game play is more productive than the two weeks of MT task. As discussed in Section 5, we encountered a significant limitation of paid crowd-sourcing: workers may not choose to do tasks they consider uninteresting. While it is possible to increase the pay rate to increase the completion rate, this entails additional costs, with deteriorating completion rates. While we expect the productivity of Doodling to scale with the number of users, MT's productivity is low even for our limited experiment, and may not scale at all.

To put this in perspective, the time taken to generate useful data using Mechanical Turk varies highly depending on the task: (Chen and Dolan, 2011) reported a duration of 2 months, whereas (Callison-Burch et al., 2009) reported 2 days for their experiments. In our Doodling experiment, the task completion rate for the game (one hour, 14 players) is faster than the equivalent Mechanical Turk task (two weeks, 9 workers). We argue that for scalable data collection, a fixed recurring cost for a reliable completion rate may be preferable over a variable recurring cost. Furthermore, the Doodling game setup is easily scalable to large user base with little marginal cost, and hence we hypothesize that the economy of scale will make Doodling cheaper than MT for diverse domains. Finally, while MT workers tend to be transient, gamers tend to be loyal, particularly if the game is perceived to be interesting. Such a user base may be likely to participate and be productive in other (perhaps related) GWAPs for the generation of useful language data.

6.4 Cheating

Doodling depends on fair gameplay in order to generate reliable phrase-equivalent. Although we did not have many cases of cheating during the trials, cases of cheating will be unavoidable as the game scales to more users. The drawer scribbling answers to the canvas is a most obvious form of cheating, which may require sophisticated image recognition algorithms to weed out automatically. However, we opted for a low-cost approach of allowing any guesser to mark a certain game round as cheating, if they find the drawer scribbling on the canvas. Any guesser can also mark a game round as cheating, if he/she finds the drawer concluding a game round with guesses that are not equivalent phrases. All guessers in a room other than the guesser who provided the accepted guess, are given three seconds to report cheating in case the guess was not found as a suitable equivalent phrase. While this methodology may not work in a two player room, we expect that in larger rooms the competitive nature of the players will keep a game honest. Frequent offenders may be penalized. Proposed penalties would be banning from game rooms, disabling certain roles or introducing harder authentication protocol to prune out offending players.

Along the same lines, we intend to introduce an "inappropriate or offending" flag, to be flagged for a drawing or a guess, by any of the players in the room. Such flags, once set, may need to be investigated offline, and the players penalized in order to discourage misuse or abuse of the game environment.

7 Conclusions and Future Work

In this paper, we explored gaming as a methodology for generating paraphrase data that is useful for NLP or IR research and development of practical systems. Specifically, we outlined a *game-with-a-purpose* – Doodling – that is based on *sketch-and-convey* metaphor, where a sketch by a Drawer was used as a mechanism for abstracting a concept (the source phrase) which was then surfaced by different guessers in the game room, potentially producing paraphrases. We showed that our online multiplayer game was effective in generating paraphrase data, by mining user guesses in the familiar *sketch-and-convey* paradigm, and rewarding phrase-equivalents in addition to exact phrase guesses. Our experiments for just one hour with volunteers have shown that this game can generate high quality data in scale. Most importantly, our volunteers rated the game "very enjoyable", even after an hour of continual play. In addition, we presented a classification mechanism to automatically identify good partial or full phrase-equivalents from the user guesses, using only the meta-level features of the game and shallow text features, opening an avenue for data generation in diverse domains, with a small seed corpora. We believe the quality of such identification may be improved significantly with addition of linguistic resources, such as, dictionaries or thesauri. Finally, our experiments with Amazon's Mechanical Turk indicated that our game is comparable to and potentially more scalable than paid crowd-sourcing. We believe such a game may be a viable mechanism for generating paraphrase data in diverse domains and languages, cheaply.

7.1 Future Work

Currently, we are in the process of developing and releasing Doodling as a multiplayer game app, providing a potential opportunity to study its uptake in the Internet, and the quality of data generated. In our experiments we measured, through a post-game survey, the potential for Doodling being a fun game, and we obtained a score of 4.7 out of 5 for "fun-factor", in addition to many verbal comments on

how enjoyable the game was. Such user feedback amply indicate Doodling's potential for scaling well as a game in diverse domains, such as sports, entertainment and idioms. Also, while the current implementation of Doodling game works well for phrases, we have ample evidence that it works for short sentences (such as, "My luggage is lost", "Where is the nearest post office?" etc.). We hope to extend it to complex sentences as future work.

One of our goals long term is to explore the game's potential for generating parallel data – perhaps through a game being played between two players conversant in two different languages. While this multi- and cross-lingual game poses significant challenges, it provides for an interesting exploration into generation of parallel data through games. Significantly, it may also provide opportunities for language learning and/or cross-cultural awareness, as many of the idioms and culture-specific phrases are not readily conveyed by the surface forms in one language or another. If successful, this may pave way for cost-effective generation of parallel data between many languages of the world.

Reference

Barzilay, R. 2003. Information Fusion for Mutli-document summarization: Paraphrasing and Generation. *Ph.D. thesis @ Columbia University*.

Barzilay, R., and McKeown, K. 2001. Extracting paraphrases from a parallel corpus. *39th Annual Meeting of the Association for Computational Linguistics*.

Callison-Burch, C. 2009. Fast, cheap, and creative: evaluating translation quality using Amazon's Mechanical Turk. *EMNLP'09*.

Callison-Burch, C., Cohn, T., and Lapata, M. 2008. ParaMetric: An Automatic Evaluation Metric for Paraphrasing, International Conference on Computational Linguistics, 2008.

Chen, D.L. and Dolan, W. 2011. Collecting highly parallel data for paraphrase evaluation. *49th Annual Meeting of the Association for Computational Linguistics*.

Lin, D., and Pantel, P. DIRT - Discovery of inference rules from text. *Proceedings of the seventh ACM SIGKDD* International conference on Knowledge discovery and data mining. ACM, 2001.

Chklovski, T. Collecting paraphrase corpora from volunteer contributors. *Proceedings of the 3rd K-CAP*. International conference on Knowledge Capture, ACM, 2005.

Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fey, A., Baker, D., Popovic, Z. and Foldit Players. 2010. Predicting protein structures with a multiplayer online game. *Nature (466), Aug 2010*.

Dolan, W., Quirk, C., and Brockett, C. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. *20th International Conference on Computational Linguistics*.

Fleiss, J. L. 1971. Measuring nominal scale agreement among many raters. Psychological Bulletin, 76, 378–382.

Ibrahim, A., Katz, B., and Lin, J. 2003. Extracting structural paraphrases from aligned monolingual corpora. *Second International Workshop on Paraphrasing (collocated with ACL 2003)*.

Irvine, A. and Klementiev, A. 2010. Using Mechanical Turk to annotate lexicons for less commonly used languages. *NAACL-HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk.*

Kumaran, A., Jauhar, S. K., and Basu, S. 2012. Doodling: A Gaming Paradigm for Generating Language Data. *Human Computation Workshop 2012*.

Kumaran, A., Saravanan, K., Datha, N., Ashok, B. and Dendi, V. 2009. WikiBABEL: a wiki-style platform for creation of parallel data. *ACL-IJCNLP 2009*.

Law, E.L.M., Von Ahn, L., Dannenberg, R. B. and Crawford, M. 2007. Tagatune: A game for music and sound annotation. *ISMIR'07*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. J. 2002. Bleu: A method for automatic evaluation of machine translation. *40th Annual Meeting of the Association for Computational Linguistics*.

Quirk, C., Brockett, C., and Dolan, W. 2004. Monolingual machine translation for paraphrase generation. *Empirical Methods in Natural Language Processing (EMNLP-2004)*.

Von Ahn, L. and Dabbish, L. 2004. Labeling images with a computer game. *CHI'04*.

Von Ahn, L., Kedia, M. and Blum, M. 2006. Verbosity: a game for collecting common-sense facts. *CHI'06*.

Von Ahn, L. and Dabbish, L. 2008. Designing games with a purpose. *Communications of the ACM, Vol 51*.

Wang, A., Hoang, C. D. V. and Kan, M. 2012. Perspectives on crowdsourcing annotations for natural language processing. Language Resources & Evaluation Conference, 2012.

Unsupervised Verb Inference from Nouns Crossing Root Boundary

Soon Gill Hong
Department of KSE
KAIST
Daejeon, Republic of Korea
hsoongil@gmail.com

Sin-Hee Cho
Department of KSE
KAIST
Daejeon, Republic of Korea
chosinhee@kaist.ac.kr

Mun Yong Yi
Department of KSE
KAIST
Daejeon, Republic of Korea
munyi@kaist.ac.kr

Abstract

Inference about whether a word in one text has similar meaning to another word in the other text is an essential task in order to understand whether two texts have similar meaning. However, this inference becomes difficult especially when two words do not share a lexical root, do not have the same argument structure, or do not have the same part-of-speech. This paper presents an unsupervised approach for inferring verbs from nouns along with a new online resource PreDic (PREdicate DICtionary) that contains verbs inferred from nouns sharing similar concepts but not the root. The verbs in PreDic are categorized into three groups, enabling applications to target precision-oriented, recall-oriented, or harmony-oriented results as needed. The experiment results show that the proposed unsupervised approach performs similar to or better than WordNet and NOMLEX. Furthermore, a new domain-verb association measure is presented to show the association relationships between inferred verbs and domains to which the verbs are possibly applied.

1 Introduction

The variability of expression is an underlying phenomenon in natural language, and the recognition of the variability serves as the foundation of understanding natural language. Recognizing textual entailment is a research area that seeks to understand this variability, and thus to identify, generate, or extract textual entailment relations from texts. Textual entailment describes a relation of texts where the meaning of one text can be inferred plausibly from another text (Dagan et al., 2010). As a related term, paraphrases refer to expressions that deliver almost the same information using different words (Androutsopoulos and Malakasiotis, 2009). As a lighter form of textual entailment, inference rules refer to expressions that carry not only the same meanings but also similar meanings and could be useful to question answering (Lin and Pantel, 2001).

Much research in recent years has focused on recognizing textual entailment pairs in natural language texts. For example, consider the following sentences:

(1) Emily Bronte *wrote* Wuthering Heights.

(2) Emily Bronte *authored* Wuthering Heights.

Given that these two sentences deliver the same meaning, the verbs *wrote* and *authored* are in a textual entailment relation.

Textual entailment plays a very important role in many areas. For example, in question answering, paraphrases from bilingual parallel corpus were used to expand the original questions (Lin and Pantel, 2001; Duboue and Chu-Carroll, 2006; Riezler et al., 2007); in information extraction, paraphrases were extracted and then used to find entities to fill the slots of binary relations (Shinyama and Sekine, 2003); in machine translation, paraphrases were captured and used as part of reference translations (Madnani et al., 2007; Marton et al., 2009).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1248–1259, Dublin, Ireland, August 23-29 2014.

Among textual entailment relations, recognizing the textual entailment of the *predicate* part of a sentence is a hard task, especially when two sentences use different words of different parts-of-speech and different argument structures. This difficulty becomes worse if the predicates of two sentences share neither any lexical root nor have proper chains from thesauri. For example, consider the following sentences:

(3) The *ingredients* of pasta are flour, eggs, and a little bit of water.

(4) Pasta is *made from* flour, eggs, and a little bit of water.

(5) What is the main *ingredient* of pasta?

As example (3) and (4) deliver the same meaning, the words *ingredients* and *made from* have a textual entailment relation. Moreover, example (4) can be an answer to example (5). However, those text pieces share neither any lexical root (*make* vs. *ingredient*) nor any syntactic structure (*X predicate Y* vs. *predicate preposition (of) X linking-verb (is) Y*) nor part-of-speech (*verb* vs. *noun*), so recognizing them as a textual entailment relation is harder than between examples (1) and (2). These inferences from nouns to verbs crossing root boundary remain unclear, and no resources have been published so far, to the best of our knowledge.

This paper presents a new unsupervised approach of inferring verbs from nouns, which share concepts but do not share roots, from glosses of multiple dictionaries. Unsupervised verb inference from nouns crossing root boundary, which covers the variable expressions between nouns and verbs, can be used to help recognize textual entailment relations. PreDic implemented the new approach and can be accessed online at *http://lod.kaist.ac.kr/predic*. PreDic only works for English nouns.

2 Related Work

Collecting similar words from a text is largely based on the *Distributional Hypothesis* (Harris, 1981). The basic idea is that words that occur in the same contexts tend to have similar meanings. Many studies in the literature acquired inference rules or paraphrases based on this hypothesis (Lin, 1998; Lin and Pantel, 2001; Bhagat and Ravichandran, 2008). If we apply that idea to the glosses of dictionaries, then we obtain many similar or relevant words in the glosses for entry words in the dictionaries.

Using dictionary glosses to understand natural language has been a popular approach. Lesk (1986) tried to identify the correct sense of each of two adjacent words, each of which having more than one gloss in the dictionary, by counting overlaps among the combinations of each gloss of each word. Glosses also have been used for extending the functionalities of another resource. Extended WordNet was built by analyzing glosses and extracting extra relations for WordNet synsets (Harabagiu et al., 1999).

Nominalization is a way of inferring nouns mainly from verbs or adjectives, especially when they share the same root. Macleod et al. (1998) built a dictionary of nominalization, NOMLEX (NOMinalization LEXicon). NOMLEX contains the nominalizations of verbs with additional information to relate the complements of nouns to the arguments of the corresponding verbs. This dictionary can be used to capture the following textual entailment relation (Bedaride and Gardent, 2009).

(6) Rome's *destruction of* Carthage.

(7) Rome *destroy(ed)* Carthage.

Argument-Mapped WordNet (Szpektor and Dagan, 2009) provides explicit mappings of arguments between verbs to alleviate the difficulty of tracking argument changes. They manually built or automatically captured rules to augment WordNet's inference capability, which permits inference over predicates only on substitution relations, such as synonyms and hypernyms, e.g. *buy* $\Rightarrow$ *acquire*. The Argument-Mapped WordNet defined only unary rules for verb-nominalization relations and verb-verb relations (e.g., X_{obj}*'s employment* $\Leftrightarrow$ *employ* X_{obj} as a nominalization-verb relation or X_{subj} $break_{\{intrans\}} \Rightarrow damage_{\{trans\}}$ X_{obj} as a verb-verb relation).

However, neither the resources of nominalization nor the mappings of argument changes can recognize examples (3) and (4) as textual entailment relations. Nonetheless, we may find a clue by chaining in WordNet. WordNet (Miller, 1995) and Extended WordNet (Harabagiu et al., 1999) contain links among synsets, so paraphrases that cross lexical root boundaries can be captured by chaining (i.e., *noun A → verb form B of noun A → verb synonym C of verb B*). We adopted this approach to compare PreDic to WordNet in the experiment.

3 Methodology

Among many entailment relations, our methodology has focused on the relations between nouns and verbs that represent similar concepts without sharing their roots. Specifically, verbs for nouns that do not share the same root are collected and then used to recognize textual entailment relations, such as the examples (3) and (4) above. The following sections describe how we collect noun-verb entailment relations crossing roots and how we categorize them for applications.

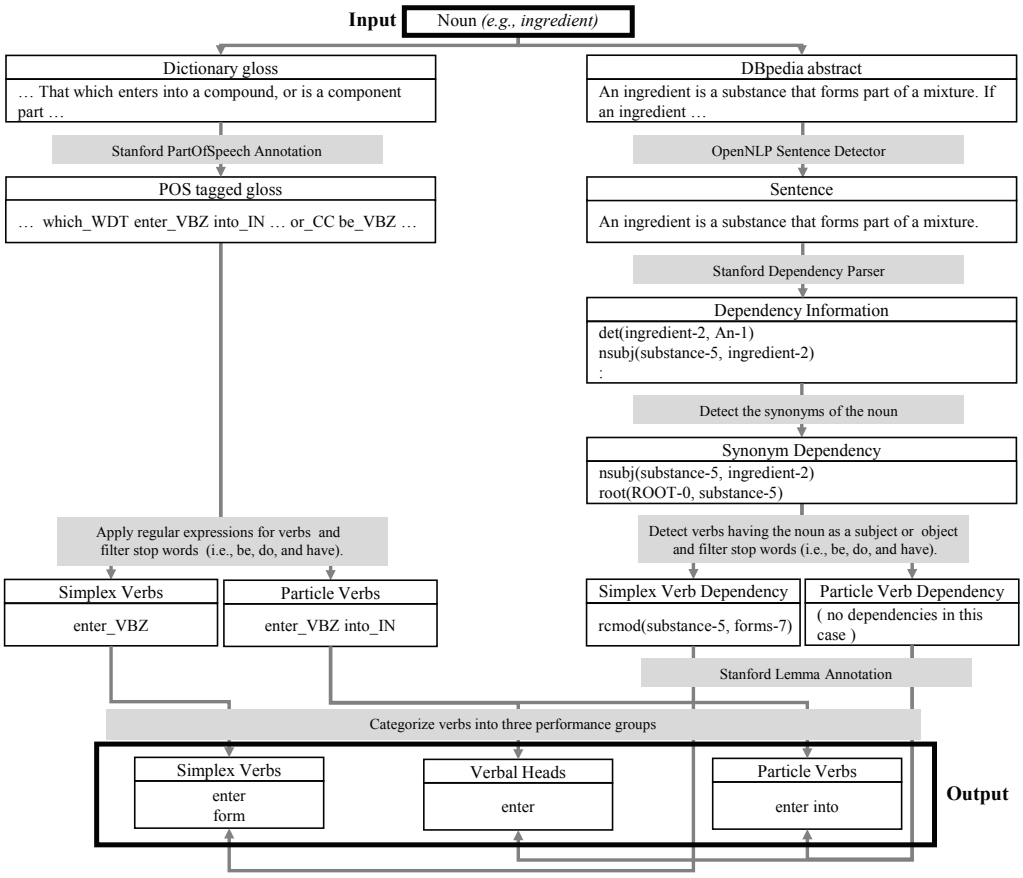


Figure 1: Algorithm consists of two major steps for generating verbs from nouns: acquisition and categorization. Verbs are detected by matching regular expression patterns against POS tagged glosses and by analyzing dependency information, and then categorized into three verb groups (Simplex, Particle, and Verbal Head).

3.1 Acquisition

Verbs that can express a similar concept of a noun can be extracted by analyzing the dictionary glosses of the noun. For example, *The Collaborative International Dictionary of English* describes *ingredient* as "...That which *enters into* a compound, or is a component part of any combination, recipe, or mixture; an element; a constituent..." This example shows that verbs used in the dictionary glosses for a noun can be regarded as having entailment relationships with the noun. Encyclopedias, such as Wikipedia[1], can

[1]http://www.wikipedia.org

also be used as a source for collecting such verbs. Here is an excerpt from Wikipedia article on the word *ingredient*: "An ingredient ... *forms...used ... purported ... required ... listed ... consists of* ..." Table 1 shows sample collected verbs that have entailment relationship with the noun *ingredient*.

Noun	Dictionary Gloss	Collected Verb
ingredient	An ingredient is a substance that *forms* part of a mixture (in a general sense)... If an ingredient itself *consists of* more than one... (Wikipedia)	*form, consist of, enter into*
	... which *enters into* a compound, or is a component part of any combination, recipe, or mixture; an element; a constituent... (The Collaborative International Dictionary of English)	

Table 1: An example of how verbs are collected from glosses. Simplex verbs and particle verbs are collected from the glosses.

Our approach uses five freely available online resources to infer verbs: The Collaborative International Dictionary of English Version 0.48 (which is also referred to as GCIDE), WordNet 3.0, DBpedia version 3.8[2] (which is a structured version of Wikipedia), dictionary.cambridge.org (especially, Cambridge Learners Dictionary and Cambridge Advanced Learners Dictionary), and www.merriam-webster.com (especially, Merriam-Webster's Collegiate Dictionary and Merriam-Webster's Learners Dictionary).

Fig. 1 shows the algorithm for generating and categorizing verbs. Dictionary glosses and texts from DBpedia are processed in different ways in the algorithm. As most of the dictionary glosses are phrases rather than sentences, they have simple syntax and few numbers of verbs. Therefore, after tagging parts-of-speech to every word in the glosses, regular expressions are used to capture verbs. However, texts from DBpedia are composed of several sentences and contain comparatively large numbers of verbs. Thus, dependency parsing is used to capture the verbs that have "close" relations to the noun.

The detailed procedures for generating verbs from dictionary glosses are described here. At the beginning, Stanford CoreNLP[3] adds a part-of-speech tag to each word in the gloss. Then, a regular expression captures simplex verbs of which part-of-speech tag is either one of the "VB", "VBD", "VBG", "VBN", "VBP", or "VBZ". Another regular expression captures particle verbs of which verb's part-of-speech tag is one of the listed above and particle's part-of-speech tag is either "RP" or "IN". Then, verbs that are too commonly used such as *have*, *be*, and *do* are filtered out. At the end, the captured verbs are categorized into three verb groups. A detailed explanation of the categorization is described at section 3.2.

The detailed procedures for generating verbs from DBpedia texts are described here. At the beginning, OpenNLP Sentence Detector[4] splits the text into sentences. Next, Stanford Dependency Parser[5] generates dependency information about the words in each sentence. Then, a list of noun synonyms are gathered from *nsubj* and *root* tags (for more information about the tags or relation names, see de Marneffe et al. (2008)). Next, simplex verbs are captured. That is, if the noun or any of the noun synonyms appears at the head position with any of *rcmod, ccomp, parataxis, vmod, partmod,* and *infmod* tag, then the word in the dependent position is captured. Similarly, if any of the noun synonyms appears at the dependent position with any of *nsubj, nsubjpass, xsubj*, and *dobj*, then the word at the head position is captured. In case of *pobj* that has "preposition" as a head and "object" as a dependent, the verb located at a different dependency relation is extracted by recursively tracing dependency relations. Afterwards, particle verbs are captured by finding particles for each of the simplex verb. That is, if a dependency relation has a *prep* tag and has any one of the simplex verbs at the head position, then the word at the dependent position is regarded as a particle candidate. When the part-of-speech tag for the particle candidate is either "RP" or "IN", then the particle candidate is regarded as a particle of interest. Consequently, the combination of the simplex verb and the particle is generated as a particle verb. Finally, the captured

[2]http://wiki.dbpedia.org/Downloads38?v=6c5

[3]http://nlp.stanford.edu/software/corenlp.shtml

[4]https://opennlp.apache.org

[5]http://nlp.stanford.edu/software/stanford-dependencies.shtml

verbs are categorized into three verb groups.

3.2 Categorization

We pay special attention to particle verbs. Particle verbs are a combination of a verb usually with an adverb or a preposition (Blaheta and Johnson, 2001). The adverbs or prepositions, when combined with simplex verbs, generate another concept that simplex verbs alone do not carry. For example, by adding the second word to *shoot*, various concepts can be produced: *shoot up, shoot off*, etc (Meyer, 1975). Hence, the definition of particle verb we use here is, to a certain degree, similar to the definition of multi-word verbs that, at the least way, carry extra meaning, or some of the words have a restricted or modified meaning when they go together.

Thus, we assume that particle verbs in text play more important roles than simplex verbs by delivering the author's intention more specifically. Based on this assumption, we built a performance group model that categorizes each verb into up to three groups. Fig. 2 shows how collected verbs are assigned to three different performance groups: (1) group of simplex verbs and verbal heads from particle verbs, (2) group of particle verbs, and (3) group of verbal heads from particle verbs. A simplex verb can be assigned to only simplex group while a particle verb, as a whole or only as a verbal head, can be assigned up to three groups. Fig. 2 formalizes the concept of categorization.

Performance Group Model: $\{v1, v2p\} \rightarrow \{\{v1, v2\}, \{v2p\}, \{v2\}\}$

- input : *v1* (simplex verb), *v2p* (particle verb)
- output : $\{v1, v2\}$ (group of simplex verbs and verbal heads of particle verbs), $\{v2p\}$ (group of particle verbs), $\{v2\}$ (group of verbal heads of particle verbs)

Figure 2: Performance group model that assigns collected verbs into three verb groups (v1: simplex verb, v2: verbal head of particle verb, v2p: particle verb). A simplex verb can be assigned to only simplex group while a particle verb, as a whole or only as a verb part, can be assigned up to three groups.

For example, when *form, consist of*, and *enter into* are collected for *ingredient*, they are categorized as follows: *form, consist*, and *enter* are assigned to the simplex group; *consist of* and *enter into* are assigned to the particle group; *consist* and *enter* are assigned to the verbal head group. Table 2 shows an example of categorization in detail.

Collected Verb	**Verb Group**		
	Simplex	**Particle**	**Verbal Head**
form, consist of, enter into	form, consist, enter	consist of, enter into	consist, enter

Table 2: Examples of how verbs are categorized into up to three verb groups (Simplex: group of simplex verbs, Particle: group of particle verbs, Verbal Head: group of verbal heads of particle verbs).

3.2.1 Simplex Verb Group

Only simplex verbs among collected verbs are assigned to the simplex group. For example, if the following verbs are collected from the glosses on the word ingredient (*form, consist of, enter into*), then *form*, *"consist"* of *consists of*, and *"enter"* of *enter into* are assigned to the simplex group. As the number of verbs in the simplex group is the largest among the three verb groups, chances are that the number of recognized texts in entailment relations using verbs in this group would be the largest among the three groups. Therefore, verbs in this group should be used to recognize as much relevant information as possible in spite of low precision. In other words, this group is suitable for recall-oriented tasks.

3.2.2 Particle Verb Group

Only particle verbs composed of two words are assigned to the particle group. For example, *consists of* and *enter into* from the collected verbs in Table 2 are assigned to the particle group. As the verbs in

this group are all particle verbs, chances are that the number of recognized texts in entailment relations using verbs in this group would be the smallest among the three groups. Therefore, verbs in this group should be used to recognize as much accurate information as possible at the expense of low recall. In other words, this group is suitable for precision-oriented tasks.

3.2.3 Verbal Head Group

The verbal head of a particle verb is the word that determines the syntactic type or the nature of that particle verb. Only verbal heads of particle verbs are assigned to the verbal head group. For example, *"consist"* of *consist of* and *"enter"* of *enter into* from the collected verbs in Table 2 are assigned to the verbal head group. This group comes between the simplex group and the particle group in terms of both precision and recall. For example, if one searched for *consist* in a text, then texts with *consist of* and *consist in* would be retrieved. It is not clear whether *consist in* fits the search needs, but it is reasonable to think that the word *consist* is common in both of the two types of search results, and therefore, all of the results would share some meaning to a certain extent. Consequently, verbs in this group should be used as a compromise between precision and recall. In other words, this group is suitable for harmony-oriented tasks.

4 Experiment

The experiment aimed at proving three things: the application performance of PreDic compared to NOMLEX that is regarded as a baseline system, the application performance of PreDic compared to WordNet, and the efficiency of the performance group model in real use. We will discuss the application performance at sections 5.1 and the efficiency of performance group model at section 5.2, respectively. In this section, we describe how the experiment was designed and performed.

4.1 Task: Textual Entailment for Relation Extraction

Relation extraction is one of the application areas that uses textual entailment as a core function. PreDic was used to extract binary relations that have textual entailment. Binary relation, *relation(X,Y)*, is one of the typical relation types, and extracting binary relation can be classified into three tasks: given two instances of *X* and *Y* (e.g., *pizza* and *dough*), find relations (e.g., *ingredient*); given one instance of *X* (e.g., *pizza*) and a relation (e.g., *ingredient*), find the other instances of *Y* (e.g., *dough*); and given a relation (e.g., *ingredient*), find instances of *X* and *Y* (e.g., *pizza* and *dough*) (Sarawagi, 2008).

As the second type (i.e., given one instance of *X* and a relation, find the other instances of *Y*) can have predefined noun relations, an experiment with this type can show how noun relations and verb relations are used interchangeably. Therefore, the experiment was performed with a predefined list of subject instances and noun relations.

4.2 Test Data

A PASCAL RTE (Recognizing Textual Entailment) dataset would be the best choice for experiment. However, as a PASCAL RTE dataset for information extraction is composed of pairs of texts, rather than a text and a structured template like the second type mentioned above (Dagan et al., 2009) , it was difficult to validate the proposed approach's capability of inferring verbs from nouns.

Therefore, we decided to use pairs of templates and texts from Wikipedia because they are easily found in Wikipedia. Article names were used as subject instances, and the property names of the infobox were used as noun relations (Wikipedia's infobox is a fixed-format table provided by the system, and people populate the table to present a summary of an article text). However, we used DBpedia instead of Wikipedia in the experiment. This is because DBpedia is easier to access from application viewpoint. That is, it already captured infobox property names from Wikipedia's article. Furthermore, DBpedia provides first few sentences as abstracts from Wikipedia's article rather than full text that is sometime too long and complex to process.

Templates were built for *Cuisine* and *Country* domains. For the *Cuisine* domain, 1,029 cuisine instance names were prepared based on the top 10 countries that have the largest number of cuisine related pages in Wikipedia's cuisine category. For the *Country* domain, 206 country instance names were prepared.

Domain	Relation	Definition
Cuisine	ingredient	Substance of the cuisine
	origin	The country or period of origin
	serving	Temperature or dishes served with
Country	border	Geographical units such as countries, rivers, or mountain, etc.
	language	Official or unofficial spoken languages
	population	The number of people living in the borders of the country

Table 3: Noun relations and definitions about relations used for the experiment.

Each domain had three noun relations (*ingredient, origin*, and *serving* for *Cuisine*, and *border, language*, and *population* for *Country*). These relations were chosen according to the frequencies of infobox property names. Thus, a total of 3,087 (1,029 instances multiplied by three noun relations) templates were prepared for the *Cuisine* domain, and a total of 618 (206 instances multiplied by three noun relations) templates were prepared for the *Country* domain. Table 3 shows the noun relations and their descriptions used in this experiment.

Total 7,996 sentences were prepared as texts from DBpedia for the *Cuisine* domain, and 3,062 sentences were prepared as texts from DBpedia for the *Country* domain. Three human raters read these sentences and marked whether each sentence expressed a similar concept to the prepared templates. For example, if a rater read "Typically pasta is made from an unleavened dough of a durum wheat flour ...", then the rater was supposed to mark the sentence as "relevant" to the template of *ingredient (X, Y)*. The agreement could be subjective, so we adopted a majority vote from three raters for each sentence. Hence, the sentences upon which the two raters agreed were annotated as relevant and put into the answer set. Each rater worked independently and was not aware of how our proposed algorithm worked.

4.3 Execution

For a given template (e.g., *ingredient (Pasta, Y)*, a number of patterns were generated by substituting the noun relation with verbs from PreDic (e.g., *made from (Pasta, Y)*, *contain (Pasta, Y)*, etc.). When the subject and predicate of each sentence matched the subject instance and verb of each pattern, the sentence was marked as "retrieved". If the retrieved sentence exists in the answer set, then it is marked as "retrieved and relevant".

The performances of PreDic was compared to the performances of NOMLEX. The verbs from NOMLEX were manually collected for the experiment. We also compared the performances of PreDic to the performances of WordNet. However, getting similar verbs of PreDic from WordNet was hard because WordNet does not directly provide verbs for a noun unless the noun itself also has a verb form. Hence, we adopted to collect verbs chaining words by navigating relations in WordNet (Szpektor and Dagan, 2009). We performed chaining up to a certain level until we could collect a similar number of verbs to PreDic. For example, when a similar number of verbs were extracted in the first search for the noun, then all verbs were collected and stopped (level 1). If a similar number of verbs were not extracted, then the extracted noun synonyms were searched again for verbs, and so on. MIT Java WordNet Interface (Finlayson, 2013) was used to collect verbs for the six noun relations from locally installed WordNet 3.1 (see Appendix for the complete list of the acquired verbs from PreDic and WordNet for the experiment. Simplex verbs are omitted if they can be generated by particle verbs).

5 Result and Discussion

5.1 Comparison to NOMLEX and WordNet

As NOMLEX provides verbs as long as nouns have their verbal forms, the performances of the two nouns (i.e., *ingredient* and *language*) could not be measured. Moreover, NOMLEX provides only simplex verbs, so only the performances using simplex verbs could be measured. Table 4 shows that PreDic is better at recall for all relations. In terms of F1, PreDic is better for four relations (i.e., *ingredient,*

origin, language, and *population*) while NOMLEX is better for two relations (i.e., *serving* and *border*). However, PreDic is better at precision for only two relations (i.e., *ingredient* and *language*) while NOMLEX is better at precision for four relations (i.e., *origin, serving, border*, and *population*). Although NOMLEX performs more precisely, its limited coverage degraded the overall performance of the resource.

Relation (R.S.)	**Verb Group**	**PreDic**					**NOMLEX**				
		Ret	**R.R.**	**Pre**	**Rec**	**F1**	**Ret**	**R.R.**	**Pre**	**Rec**	**F1**
ingredient (1413)	simplex	1104	571	**0.52**	**0.40**	**0.45**	-	-	-	-	-
origin (636)	simplex	1298	242	0.19	**0.38**	**0.25**	97	81	**0.84**	0.13	0.22
serving (505)	simplex	1124	344	0.31	**0.68**	0.42	407	285	**0.70**	0.56	**0.63**
border (233)	simplex	259	112	0.43	**0.48**	0.45	105	105	**1.00**	0.45	**0.62**
language (80)	simplex	245	9	**0.04**	**0.11**	**0.06**	-	-	-	-	-
population (133)	simplex	216	7	0.03	**0.05**	**0.04**	3	1	**0.33**	0.01	0.01

Table 4: Application Performance of PreDic and NOMLEX. The coverage of NOMLEX is limited to the nouns that have verbal forms. R.S.: number of relevant sentences, Ret: number of retrieved sentences, R.R.: number of retrieved & relevant sentences, Pre: precision (%), Rec: recall (%), F1: F1 score (%). The best scores for each relation are printed in **bold**.

Table 5 shows the performance comparison between PreDic and WordNet. According to Table 5, PreDic is better at recall for all relations except *border*. PreDic is better at precision for three relations (*ingredient, origin*, and *population*) and WordNet is better for three relations (*serving, border*, and *language*). In terms of F1, PreDic is better for four relations (*ingredient, origin, border*, and *population*), and WordNet is better for two relations (*serving* and *language*).

Relation (R.S.)	**Verb Group**	**PreDic**					**WordNet**				
		Ret	**R.R.**	**Pre**	**Rec**	**F1**	**Ret**	**R.R.**	**Pre**	**Rec**	**F1**
ingredient (1413)	simplex	1104	571	0.52	**0.40**	0.45	798	424	0.53	0.30	0.38
	particle	293	245	**0.84**	0.17	0.29	0	0	-	0	-
	verbal head	971	527	0.54	0.37	**0.44**	49	6	0.12	0.00	0.01
origin (636)	simplex	1298	242	0.19	**0.38**	0.25	111	16	0.14	0.03	0.04
	particle	86	58	**0.67**	0.09	0.16	3	0	0.00	0.00	0.00
	verbal head	207	117	0.57	0.18	**0.28**	65	7	0.11	0.01	0.02
serving (505)	simplex	1124	344	0.31	**0.68**	0.42	393	272	**0.69**	0.54	**0.61**
	particle	83	6	0.07	0.01	0.02	1	0	0.00	0.00	0.00
	verbal head	963	292	0.30	0.58	0.40	384	272	0.71	0.54	0.61
border (233)	simplex	259	112	0.43	0.48	0.45	184	122	**0.66**	**0.52**	0.59
	particle	15	8	0.53	0.03	0.06	0	0	-	0	-
	verbal head	153	105	0.69	0.45	**0.54**	55	5	0.09	0.02	0.03
language (80)	simplex	245	9	0.04	**0.11**	0.06	16	6	**0.38**	0.08	**0.13**
	particle	54	4	0.074	0.05	0.06	0	0	-	0	-
	verbal head	122	8	0.066	0.10	0.08	7	0	0.00	0.00	0.00
population (133)	simplex	216	7	0.03	**0.053**	0.04	129	6	0.05	0.045	0.05
	particle	7	1	**0.14**	0.01	0.01	0	0	-	0	-
	verbal head	90	3	0.03	0.023	**0.27**	42	2	0.05	0.015	0.02

Table 5: Application Performance of PreDic and WordNet. PreDic shows better or similar performances than WordNet (refer to Table 4 for the acronyms in the table header). The best scores for each relation are printed in **bold**.

If we compile all counts and scores of each relation into verb groups by micro average and macro average, we can get more straightforward comparisons. Micro-averaging assigns equal weight to each instance (e.g., each retrieval) regardless of classes, whereas macro-averaging assigns equal weight to each class (e.g., each predicate). Table 6 shows that PreDic is the best at recall for both micro average (i.e., 0.43) and macro average (i.e., 0.42), while NOMLEX is best at precision for both micro average (i.e., 0.77) and macro average (i.e., 0.48). However, the large difference in precision for NOMLEX between micro and macro average (i.e., 29 percent) shows that NOMLEX performs well on some nouns but not on other nouns. In contrast, PreDic provides not only the better recall and broader coverage than NOMLEX and WordNet, but also competitive macro average precision (i.e., 0.39 vs. 0.41) compared to NOMLEX or even better micro average precision (i.e., 0.60 vs. 0.52) compared to WordNet.

	Verb Group	**Precision**			**Recall**		
		PreDic	**WordNet**	**NOMLEX**	**PreDic**	**WordNet**	**NOMLEX**
Micro Average	simplex	0.30	0.52	**0.77****	**0.43**	0.28	0.16**
	particle	0.60	0.00*	N/A	0.11	0.00	N/A
	verbal head	0.42	0.49	N/A	0.35	0.10	N/A
Macro Average	simplex	0.25	0.41	**0.48****	**0.42**	0.25	0.19**
	particle	0.39	0.00*	N/A	0.07	0.00	N/A
	verbal head	0.37	0.18	N/A	0.28	0.10	N/A

Table 6: Micro and Macro Average Performances of PreDic, WordNet, and NOMLEX. Scores for WordNet (*) were calculated by using only two relations (*origin* and *serving*) and scores for NOMLEX (**) were calculated by using only four relations. The best scores at precision and recall for each average are printed in **bold**.

The results imply that an unsupervised approach can outperform over hand-crafted resources. This also implies that unsupervised approaches can contribute to building diverse lexical resources and cover more variability of expressions in natural language as well.

5.2 Efficiency of Performance Group Model for PreDic

If we narrow the scope of performance to PreDic, we can see from the performances of PreDic in Table 5 that the verbs from the simplex group are best at recall for all relations as expected, the verbs from the particle group are best at precision for four relations (i.e., *ingredient, origin, language*, and *population*), and the verbs from the verbal head group are best at F1 for four relations (i.e., *origin, border, language*, and *population*). These results are consistent with the results in Table 6. The performances of PreDic in Table 6 show that the particle group is best at precision for micro and macro averages (i.e., 0.60 and 0.39, respectively) and the simplex group is best at recall (i.e., 0.43 and 0.42, respectively).

These results assure that our assumption for performance group model is convincing. This implies that performance group model can be adopted for implementing tasks as a precision-oriented, recall-oriented, or even harmony-oriented as needed. That is, as the variability of natural language is hard to predict, this performance group model plays a very important role in guiding applications on whether to focus on getting high-quality information at the expense of large quantities of information, large quantities at the expense of high quality, or a compromise between these two extremes given limited available time and cost. For example, when acquiring more verbs from another source, particle verb group can be used since it provides few but accurate seed verbs, whereas simplex verb group can be preferable when extracting information from texts because it offers more verbs.

6 Domain Verb Association

The relationship between nouns and inferred verbs can be measured by counting co-occurrences using web search engines (Soderland et al., 2004), and in this paper we used Google to collect the frequencies of the co-occurrences. According to Table 7, *prepare, contain, make from, form*, and *consist of* show

Verb	Hits w/ *ingredient*
prepare	56,100,000
contain	46,400,000
make from	37,800,000
form	33,700,000
consist of	15,500,000
attract	3,440,000
occupy with	3,420,000
display in	3,260,000
list by	2,050,000
use with	839,000
enter into	661,000

Table 7: Noun-Verb co-occurrence counts. These numbers provide conceptual relationships between the noun *ingredient* and the inferred verbs.

Verb	Hits w/ Cuisine	DVA
make from	7,452,978	**1.00**
consist of	1,454,047	**0.20**
prepare	718,686	**0.10**
form	406,778	**0.05**
use with	191,220	**0.03**
contain	101,411	**0.01**
enter into	22,749	0.00
display in	8,714	0.00
attract	4,298	0.00
list by	4,605	0.00
occupy with	9	0.00

Table 8: Domain Verb Associations (DVA) for *Cuisine* domain and the inferred verbs. The verbs with DVA above 0.00 (printed in **bold**) seem to be more associated with the *Cuisine* domain.

Verb	Hits w/ Drug	DVA
contain	1,651,933	**1.00**
make from	594,561	**0.36**
use with	490,588	**0.30**
consist of	54,163	**0.03**
form	42,963	**0.03**
prepare	30,366	**0.02**
attract	121	0.00
display in	51	0.00
enter into	12	0.00
list by	9	0.00
occupy with	0	0.00

Table 9: Domain Verb Associations (DVA) for *Drug* domain and the inferred verbs. The verbs with DVA above 0.00 (printed in **bold**) seem to be more associated with *Drug* domain.

much more co-occurrences with *ingredient* than with the other verbs (with a threshold of 10 million, for example). Although the co-occurrences do not consider the distance between two words, they must reveal the degree of relationships between the concept of nouns and their actual verbal forms in texts.

However, what matters more is how much inferred verbs are used with the words of interest rather than a noun itself. Furthermore, if we can rank preferred verbs by domains, inferred verbs can be more useful to applications that focus on specific domains. Hence we defined Domain Verb Association (DVA) to measure how frequently inferred verbs are used with domain instances that can be used as subjects or objects for the verbs. Let D denote a set of domain instances, V a set of verbs inferred from a predicate P, v_f a verb form of a (base form) verb v. Domain Verb Association measures a normalized association score for an ordered combination of a domain and a verb by summing the co-occurrences of each domain instance in the domain and each verb form of the base form of the verb:

$$DVA\,(D, v|P) = \frac{\sum_{d_i \in D} \sum_{v_f \in v} hits\,(d_i \,||\, v_f)}{\max_{v \in V} DVA\,(D, v|P)} \quad (1)$$

where *hits* is the number of search engine hits for query and $d_i \,||\, v_f$ is a concatenation of two words enclosed by " and ".

For the experiment, we defined present simple, past simple, and simple present passive voice as a set of verb forms, without taking the argument structures of the verbs into account for simplicity. We chose *hamburger, pasta*, and *sandwich* as a set of sample representative instances of the domain *Cuisine*. We also selected *Advil, Aspirin*, and *Benadryl* as a set of sample representative instances of the domain *Drug*. Consequently, queries $d_i \,||\, v_f$ were built like *"pasta makes from", "pasta made from"* or *"pasta is made from"* for each combination of a domain instance and an inferred verb. *DVA* scores for the association of the inferred verbs from *ingredient* and the two domains (*Cuisine* and *Drug*) using Eq. (1) are shown in Table 8 and Table 9, respectively. The results show that each domain prefers some verbs to other verbs in that *make from* is the most frequent in *Cuisine* domain but *contain* is the most frequent in *Drug* domain. *Make from* is used about 70 times more often than *contain* in *Cuisine* domain, while *contain* is used about two and a half times more often than *make from* in *Drug* domain. Certainly we believe that this measure will help to improve the application performance of using PreDic.

7 Conclusion

We have presented an unsupervised approach for inferring verbs from nouns crossing root boundary and introduced a new lexical resource, *PreDic*, which is an implementation of the approach and contains verbs inferred from nouns that share neither a root nor argument structure nor a part-of-speech. We have also demonstrated a performance group model that arranges verbs into three groups is practical enough to guide applications to pursue recall-oriented, precision-oriented, or harmony-oriented results. Furthermore, the Domain Verb Association measure was introduced to show the relationships between inferred verbs and domains to which the inferred verbs are possibly applied.

Many researchers have suggested effective approaches for verb entailment acquisition and built valuable lexical resources with which the variability of natural language expression can be understood more systematically. However, unsupervised verb inference from nouns that can deliver similar meaning without shared roots has not been explicitly addressed so far. This research presents compelling evidence that the proposed approach can be a stepping stone for such applications as information extraction or natural language question answering in understanding the variability of natural language expression and recognizing such relations in text. Our future research needs to incorporate more syntactic and external knowledge, and to learn more verbs using some of the inferred verbs as seeds. Moreover, the inference over composite nouns and other parts of speech will also be investigated. Notwithstanding these future research issues, the present research findings provide clear evidence that utilizing verb inference from nouns is a fruitful textual inference approach.

Acknowledgements

This research was conducted by the International Collaborative Research and Development Program (Creating Knowledge out of Interlinked Data) and funded by the Korean Ministry of Knowledge Economy.

References

Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, and Ruth Reeves. 1998. Nomlex: A lexicon of nominalizations. In *Proceedings of EURALEX* (Vol. 98, pp. 187–193).

Dekang Lin. 1998. Extracting collocations from text corpora. In *First workshop on computational terminology*, (pp. 57-63).

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(4):343–360.

De Marneffe, Marie-Catherine, and Christopher D. Manning. 2008. Stanford typed dependencies manual. *URL http://nlp. stanford. edu/software/dependencies manual. pdf.*

Don Blaheta and Mark Johnson. 2001. Unsupervised learning of multi-word verbs. In *Proc. of the ACL/EACL 2001 Workshop on the Computational Extraction, Analysis and Exploitation of Collocations* (pp. 54-60).

George A. Meyer. 1975. *The two-word verb: A dictionary of the verb-preposition phrases in American English* (No. 19). Walter de Gruyter.

George A. Miller. 1995. Wordnet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Idan Szpektor and Ido Dagan. 2009. Augmenting wordnet-based inference with argument mapping. In *Proceedings of the 2009 Workshop on Applied Textual Inference*.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 16(01):105–105.

Ion Androutsopoulos and Prodromos Malakasiotis. 2009. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38:135–187.

Mark Alan Finlayson 2013. Code for Java Libraries for Accessing the Princeton Wordnet: Comparison and Evaluation. In *Proceedings of the 7th Global Wordnet Conference*.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation* (pp. 24–26). ACM.

Nitin Madnani, Necip Fazil Ayan, Philip Resnik, and Bonnie J. Dorr. 2007. Using paraphrases for parameter tuning in statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation* (pp. 120–127). Association for Computational Linguistics.

Pablo Ariel Duboue and Jennifer Chu-Carroll. 2006. Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers* (pp. 33–36). Association for Computational Linguistics.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, (pp. 113–120). Association for Computational Linguistics.

Paul Bedaride and Claire Gardent. Noun/verb inference. 2009. *Human Language Technologies au a Challenge for the Computer Science and Linguistic*, 311–315.

Ravichandran Bhagat and Deepak Ravichandran. 2008. Large scale acquisition of paraphrases for learning surface patterns. In *ACL* (Vol. 8, pp. 674-682).

Sanda Harabagiu, George Miller, and Dan Moldovan. 1999. Wordnet 2–a morphologically and semantically enhanced resource. In *Proceedings of SIGLEX*, (vol. 99, pp. 1–8).

Stefan Riezler, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu Mittal, and Yi Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS* (Vol. 45, No. 1, p. 464).

Stephen Soderland, Oren Etzioni, Tal Shaked, and Daniel S. Weld. 2004. The use of Web-based statistics to validate information extraction. In *AAAI-04 Workshop on Adaptive Text Extraction and Mining* (pp. 21-26).

Sunita Sarawagi 2007. Information extraction. In *Foundations and trends in databases* (Vol. 1, No. 3, pp. 261-377).

Yusuke Shinyama and Satoshi Sekine. 2003. Paraphrase acquisition for information extraction. In *Proceedings of the second international workshop on Paraphrasing-Volume 16* (pp. 65–71). Association for Computational Linguistics.

Yuval Marton, Chris Callison-Burch, and Philip Resnik. 2009. Improved statistical machine translation using monolingually-derived paraphrases. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1* (pp. 381–390). Association for Computational Linguistics.

Zelig S. Harris. 1981. *Distributional structure* (pp. 3-22). Springer Netherlands.

Appendix. List of verbs used for the experiment from PreDic and WordNet

PreDic

ingredient	serving	language	language
attract	make	communicate by	use by
consist of	serve of	compare	use of
contain	take as	consist of	utter
display in	border	convey	want
enter into	approach	create for	write
form	arrange along	depend on	population
list by	border on	descend from	begin
make from	come near	describe	belong
need	confine within	distinguish by	cause
occupy with	contest	evolve	clothe
prepare	cross	example in	come
use with	define	execute	control
origin	divide	express in	convict
bear	form	garble	define
begin	foster of	group of	deplore
cause by	furnish with	include of	draw
come from	grow up	introduce	educate
create with	indicate	involve in	entail
derive	limit	man	experience
describe	live in	name	feed
exist	make	originate	give
fix during	open	produce	go
give	plant	record	hire
know	print	refer	increase
make	separate	related with	inhabit
name for	settle	rely on	interbreed
originate in	touch at	represent	keep down
proceed	use before	see as	live in
rise	walk	set	make up
spring into	language	speak by	occupy
start	achieve	speak in	populate
use	arrange	speak throughout	refer in
serving	articulate by	specify	remain
accept	associate	start with	represent
accord	base for	study of	seem
deliver	base on	take	take for
eat	belong	teach	use before
employ at	call	think	
help of	combine	understand	

WordNet

ingredient	ingredient	ingredient	serving	border	border	population
allot	neuter	vivify	serve well	evade	stick to	face up
alter	part	origin	service	exhibit	surround	follow
amend	pay back	begin	suffice	fence in	take a hop	front
animate	pay off	blood	swear out	frame in	telephone	live
assign	portion	buy in	wait on	frame up	throttle	look
break up	posit	carry	border	fudge	tie down	make up
broker	prepare	commence	abut	hedge	tie up	man
bushel	quicken	delineate	adhere	hem in	touch	personify
castrate	ready	describe	adjoin	hold fast	trammel	population
change	reanimate	draw	attach	inch	truss	present
compensate	recompense	get down	band	jump	wall	represent
cook	recreate	lead off	bandage	knell	language	
define	rectify	line	beleaguer	leap	address	
depart	remediate	origin	besiege	limit	articulate	
deposit	remedy	root	bind	march	formulate	
desex	renovate	rootle	bond	meet	give voice	
desexualise	repair	rout	border	obligate	language	
desexualize	resort	run along	bounce	oblige	lyric	
determine	restore	set about	bound	palisade	mouth	
disunite	revive	set out	bunt	parade	phrase	
divide	revivify	settle down	butt against	parry	sound	
doctor	secure	source	butt on	peal	speak	
factor in	separate	sprout	call up	phone	talk	
factor out	set forth	start out	cast	process	utter	
falsify	set off	steady down	circumvent	put off	verbalise	
fasten	set out	stock up	compose	rebound	verbalize	
fix	set up	stockpile	confine	recoil	vocalise	
fixate	situate	take root	constipate	redact	vocalize	
furbish up	spay	trace	contact	resile	voice	
gear up	specify	serving	couch	resound	word	
get	split up	answer	demonstrate	restrict	population	
heal	start out	assist	dodge	reverberate	comprise	
indemnify	sterilise	attend to	draw up	ricochet	confront	
ingredient	sterilize	dish out	duck	ring	constitute	
interpolate	take off	dish up	echo	set up	cost	
limit	tighten	function	elude	sidestep	earth	
make	touch on	help	ensnare	skirt	embody	
mend	unsex	process	entrap	smother	equal	
modify	vary	serve up	environ	spring	exist	

Enriching Wikipedia's Intra-language Links by their Cross-language Transfer

Takashi Tsunakawa **Makoto Araya** **Hiroyuki Kaji**
Graduate School of Informatics, Shizuoka University
3-5-1 Johoku, Naka-ku, Hamamatsu, Shizuoka 432-8011, Japan

{tuna, araya, kaji}@inf.shizuoka.ac.jp

Abstract

Although hyperlinks enhance the utility of Wikipedia, embedding them in articles imposes a burden on contributors. To alleviate this burden as well as enrich hyperlinks in Wikipedia articles, we propose a method for transferring intra-language links between different-language articles linked via an inter-language link. The method avoids anchor selection and disambiguation problems by which usual wikification methods are affected, by exploiting the analogy between different language editions of Wikipedia. The effectiveness of the method was demonstrated through an experiment of transferring intra-language links from English to Japanese. It increased the number of intra-language links in Japanese articles by 40.9%, and the accuracy of anchors selected was estimated to be 96.3%.

1 Introduction

Wikipedia is a Web-based encyclopedia constructed collaboratively by many contributors and continues to enlarge and improve daily. Because of its overwhelming scale, improved quality, and multilingual nature, it has acquired a huge number of readers worldwide. One of the distinguishing features of Wikipedia is that it is a hypertext, which greatly enhances its usefulness and usability. That is, an article is linked to its related articles in the same language via intra-language links as well as to its counterpart articles in different languages via inter-language links (ILLs), and readers can navigate within millions of articles.

Editing Wikipedia articles naturally includes linking them to their related articles, which imposes an additional burden on contributors. As a result, Wikipedia articles may remain incomplete; they sometimes lack important links as well as contain incorrect links. Thus, it is desirable to automate link-related editorial tasks such as embedding links in new articles and verifying links in existing articles. Linking a plain text, usually non-Wikipedia articles, to Wikipedia articles is called wikification, and much effort has been devoted to developing a variety of wikification methods over the past decade (Mihalcea and Csomai, 2007; Milne and Witten, 2008a; Fogarolli, 2009; Ratinov et al., 2011). However, wikification methods are still immature and affected by two hard problems; anchor selection, which involves keyword extraction or term recognition, and destination-article determination, which is a kind of word sense disambiguation (WSD).

We focused on the comparability of intra-language links between different language editions of Wikipedia, and developed a method for transferring intra-language links in one language edition to another language edition. Although the method is not applicable to texts other than Wikipedia articles, it avoids the problems of anchor selection and destination-article disambiguation by using analogy with different language editions. It does not require any language resources other than Wikipedia itself. When the target language is a morphologically rich one, a morphological analyzer is also required. Although the method is applicable to any language pairs, we evaluated its effectiveness through an

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1260–1268, Dublin, Ireland, August 23-29 2014.

experiment of transferring intra-language links from English to Japanese.

2 Basic Idea

In Wikipedia, an article in one language is often linked to another article in another language via an ILL. These two articles, which describe the same entity, concept or topic, are comparable. Note that this comparability holds not only for texts in articles but also for intra-language links, each of which links an anchor or an important term within an article to another same-language article describing the entity, concept, or topic denoted by the anchor term. Figure 1 gives an example pair of ILL-linked articles; an English article "Tata Motors" and a Japanese article "タタ・モーターズ." The former has an intra-language link from an anchor "Jaguar" to the English article "Jaguar Cars," while the latter has an intra-language link from an anchor "ジャガー" to the Japanese article "ジャガー (自動車)." These two intra-language links are comparable: namely, the anchors are translations of one another and the destination articles are linked via an ILL.

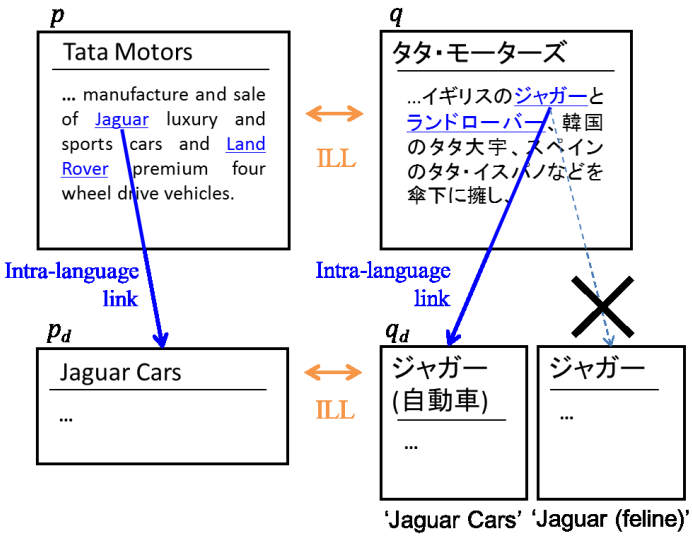


Figure 1. Transferring intra-language link.

The above fact inspired us to develop a method for transferring intra-language links between ILL-linked articles to enrich the intra-language links in each article. Suppose an extreme case in which an article q in one language, which is linked to its counterpart p in another language via an ILL, has no intra-language links. An intra-language link can be transferred from p to q as follows. First, following an intra-language link (p to p_d) and then the ILL (p_d to q_d), the final destination article q_d is identified as that to be linked from q. Second, the text of q is searched for possible anchors for the destination article q_d, which are learned from the entire Wikipedia beforehand. If two or more possible anchors are found, the most appropriate one will be selected according to a certain criterion. For example, suppose all intra-language links are missing from the Japanese article "タタ・モーターズ" in Figure 1. The intra-language link from the English article "Tata Motors" to "Jaguar Cars" and the ILL from "Jaguar Cars" to "ジャガー (自動車)" suggest that the Japanese article "タタ・モーターズ" should have an intra-language link to "ジャガー (自動車)." The possible anchors for "ジャガー (自動車)," which have been learned from all the Wikipedia articles linked to it, include "ジャガー (自動車)," "ジャガー," and others. Since the text of "タタ・モーターズ" contains "ジャガー," it is selected as the anchor for the destination article "ジャガー (自動車)."

It should be noted that our proposed method avoids the two hard problems in wikification, anchor selection and disambiguation, by exploiting the intra-language links provided by Wikipedia in another language. Resulting anchors are certainly important terms within q, since their counterparts have been selected as anchors by the author of counterpart p in another language. Even if an anchor were an ambiguous term, i.e., had two or more possible destination articles, it would be certainly linked to the appropriate one due to the "one sense per discourse" hypothesis (Gale et al., 1992). The hypothesis is extended to a pair of ILL-linked articles, p and q, as follows. A pair of corresponding anchors should be regarded as a single term and express the same sense in a discourse shared by p and q. In other words, they should be linked to articles that are linked via an ILL. Since the proposed method relies on this extended hypothesis, it will select correct destination articles for anchors in q as long as anchors in p have been linked to their correct destination articles.

It should also be noted that the proposed method first determines the destination articles then the anchors for them, while usual wikification methods first select anchors then determine their destination articles. The main reason for this is convenience of implementation; cross-language mapping of

destination articles is one-to-one (or one-to-zero), while that of anchors can be one-to-many. Determining destination articles prior to anchors, however, results in an additional advantage that allows a destination article to be proposed without an anchor for it. Since the pair p and q is not parallel but just comparable, the counterpart of an anchor in p is not always found in q. This is often the case when q is incomplete, under construction, or written in a different style from that of p. In such a case, our method proposes a destination article q_d without an anchor, and q will be linked to q_d once q is enlarged to contain a term appropriate as the anchor for q_d.

3 Proposed Method

The proposed method is divided into two steps; the preprocessing step for collecting possible anchors for all Wikipedia articles in a target language as well as estimating probabilities required in the succeeding step and the main step for transferring intra-language links in a source-language article p to the target-language article q linked to p via an ILL. In this section, a triplet (p, a, p_d) denotes an intra-language link from anchor a in article p to destination article p_d and, likewise, a triplet (q, b, q_d) does. Note that although an article can have two or more intra-language links from the same anchor at different positions in the text to the same destination article, they are treated as a single link.

3.1 Preprocessing Step

Collecting Possible Anchors for Wikipedia Articles

The title of a Wikipedia article can be used as an anchor for the article. However, a title is often accompanied by a parenthesized note indicating the domain of the article to discriminate from other articles with the same title. The title "ジャガー (自動車)" of an article that describes a car named Jaguar is an example; the parenthesized note "(自動車)" discriminates the article from another article "ジャガー", which describes an animal belonging to the cat family. Such a title accompanied by a parenthesized note rarely occurs in usual texts, and the title with the parenthesized note deleted is often marked as an anchor. Accordingly, we also regard a title with a parenthesized note deleted (e.g., "ジャガー") as a possible anchor. Other terms, typically synonyms of the article title, are often used as anchors. Therefore, we collect terms that are actually used as anchors for each article from the entire Wikipedia. Finally, we threshold possible anchors by their keyphraseness to eliminate general words. The keyphraseness $\kappa(b)$ of a term b is defined as the probability that b is used as an anchor in Wikipedia articles (Mihalcea and Csomai, 2007), i.e.,

$$\kappa(b) = \frac{|\{q|\exists q_d.\,(q, b, q_d) \in L_t\}|}{\mathrm{df}(b)},$$

where L_t is a set consisting of all intra-language links in the target-language Wikipedia and $\mathrm{df}(b)$ is the number of Wikipedia articles in which b occurs.

In summary, a set of possible anchors $\mathrm{A}(q_d)$ are constructed for a target-language destination article q_d as follows:

$$\mathrm{A}(q_d) = (\{title(q_d), title'(q_d)\} \cup \{b|\exists q.\,(q, b, q_d) \in L_t\}) \cap \{b|\kappa(b) \geq \theta\},$$

where $title(q_d)$ and $title'(q_d)$ are q_d's title with and without the parenthesized note, respectively, and θ is the threshold for the keyphraseness.

Estimating Probabilities

The following probabilities, which will be used to select one from among possible anchors for a destination article, are estimated from the entire Wikipedia.

- The probability that the target-language anchor is b on the condition that its source-language counterpart is a, i.e.,

$$\mathrm{P}(b|a) = \frac{\mathrm{count}(a, b)}{\sum_{b'} \mathrm{count}(a, b')},$$

$$\mathrm{count}(a,b) = \left|\left\{\left((p,a),(q,b)\right)\right| \begin{array}{c}\exists p_d.\,\exists q_d.\,(p,a,p_d) \in L_s \wedge (q,b,q_d) \in L_t \\ \wedge\, (p,q) \in ILL \wedge (p_d,q_d) \in ILL\end{array}\right\}\right|,$$

where L_s is a set consisting of all intra-language links in the source-language Wikipedia, and ILL is a set of all pairs of ILL-linked articles.

- The probability that the anchor is b on condition that the destination article is q_d, i.e.,

$$\mathrm{P}(b|q_d) = \frac{|\{q|(q,b,q_d) \in L_t\}|}{|\{q|\exists b'.(q,b',q_d) \in L_t\}|}$$

- The probability that the destination article is q_d on condition that the anchor is b, i.e.,

$$\mathrm{P}(q_d|b) = \frac{|\{q|\ (q,b,q_d) \in L_t\}|}{|\{q|\exists q'_d.(q,b,q'_d) \in L_t\}|}$$

3.2 Main Step

Let p and q be source-language and target-language articles that are linked via an ILL, respectively. Intra-language links in p are transferred to q as follows:

(i) For each source-language intra-language link (p,a,p_d), do (ii) to (v).

(ii) If p_d has an ILL to an article in the target language, let q_d be the destination article of the ILL from p_d. Otherwise, output "NOT TRANSFERRED" and move to the next intra-language link.

(iii) If $\mathrm{A}(q_d)$ is empty, output the transferred intra-language link (q, NULL, q_d), which means that q should be linked to q_d but does not contain a term appropriate as the anchor, and move to the next intra-language link.

(iv) For each possible anchor $b \in \mathrm{A}(q_d)$, search the text of q for b. If found, let $\mathrm{pos}(b,q)$ denote the position of its first occurrence in the text; otherwise, let $\mathrm{pos}(b,q) = -1$.

(v) If at least one possible anchor is found, choose the most appropriate one $\hat{b}$ according to an anchor priority score $\mathrm{Score}(b)$, i.e.,

$$\hat{b} = \underset{b \text{ s.t. } b \in \mathrm{A}(q_d) \wedge \mathrm{pos}(b,q) \geq 0}{\mathrm{argmax}} \mathrm{Score}(b).$$

and output the transferred intra-language link $(q, \hat{b}, q_d)$. Otherwise, output the transferred intra-language link (q, NULL, q_d).

We have the following five alternative anchor priority scores in step (v) above.

- Anchor translation probability: $\mathrm{Score}_1(b) = \mathrm{P}(b|a)$.
  This score favors the anchor that occurs most frequently as counterpart to the source-language anchor.

- Anchor probability: $\mathrm{Score}_2(b) = \mathrm{P}(b|q_d)$.
  This score favors the anchor by which the destination article is pointed most frequently.

- Destination article likelihood: $\mathrm{Score}_3(b) = \mathrm{P}(q_d|b)$.
  This score favors the anchor that is most likely to point the destination article.

- Spelling: $\mathrm{Score}_4(b) = 1 - \mathrm{dist}\big(b, title'(q_d)\big)/\max\{\mathrm{len}(b), \mathrm{len}(title'(q_d))\}$,
  where $\mathrm{dist}(s,s')$ is the Levenshtein distance between character strings s and s' (Levenshtein, 1966), and $\mathrm{len}(s)$ is the length of character string s.

  This score favors the anchor with the highest similarity to the article's title without a parenthesized note, which is the most representative term denoting the entity, concept, or topic described in the article.

- Position: $\mathrm{Score}_5(b) = 1/\mathrm{pos}(b,q)$.
  Note that in a Wikipedia article, among two or more occurrences of an important term, the first one tends to be marked as an anchor.

4 Experiment

4.1 Experimental Settings

We conducted an experiment on transferring intra-language links from the English edition to the Japanese edition of Wikipedia.

Input Data

The English edition of Wikipedia (2013-04-03 dump), consisting of 4,241,324 articles, and the Japanese edition of Wikipedia (2013-03-28 dump), consisting of 951,411 articles[1], were used for the experiment. Intra-language links were extracted from each dump file, and ILLs were obtained from Wikidata (2013-03-28 dump). Redirect pages were resolved preliminarily, i.e., if the destination of an intra-language link or ILL was a redirect page, the destination was replaced with an article pointed by the redirect page.

Anchor	English translation	Keyphraseness
ベイジアン・ネットワーク	Bayesian network	1
地獄の辞典	Dictionnaire Infernal	0.810
悪魔学	demonology	0.678
パリ	Paris	0.574
オカルト	occult	0.304
悪魔	devil	0.135
ニコラス	Nicholas	0.039
対立	conflict	0.001
半分	half	7.8×10^{-5}
より	Yori (kana)	4.4×10^{-6}

Table 1. Example of keyphraseness values.

From among a total of 366,358 pairs of English and Japanese articles linked by ILLs, 3,595 pairs were randomly selected as a test set. The remaining pairs were used as training data for constructing English and Japanese intra-language link sets, L_s and L_t. The English articles in the test set contained 179,963 intra-language links in total; these were input to the algorithm of the proposed method.

Keyphraseness Threshold

Limiting possible anchors to meaningful ones and gaining many links are in a trade-off relation adjustable by the keyphraseness threshold θ. In the experiment, θ was set to 0, 0.01, 0.05, and 0.1.

Keyphraseness values of several anchors are listed in Table 1. Technical words (e.g., "ベイジアン・ネットワーク" – Bayesian network) and uncommon proper names (e.g., "地獄の辞典" – Dictionnaire Infernal) tend to have high keyphraseness values. Common words (e.g., "悪魔" – devil and "対立" – conflict) and proper names (e.g., "パリ" – Paris and "ニコラス" – Nicholas), especially identical to a general noun, have middle or low values according to their commonness. Although some functional words (e.g., "より" – from) may be included in possible anchors for the Wikipedia articles of their homographic content words (e.g., "より" – Yori (kana)), they naturally have extremely low values. By setting θ to a value slightly greater than zero, functional words could be removed from possible anchors.

Comparison of Anchor Priority Scores

To determine the most effective anchor priority score, the accuracy of anchors selected according to each score was evaluated, assuming the existing intra-language links in the original Japanese articles as gold standard. That is, anchor accuracy Acc is defined as the percentage of originally pointed destination articles for which correct anchors were selected, i.e.,

$$\mathrm{Acc} = \frac{|Result \cap GoldSTD|}{|\{(q, b, q_d) \in Result | \exists b'.(q, b', q_d) \in GoldSTD\}|},$$

where $Result$ is a set consisting of all transferred intra-language links and $GoldSTD$ is the gold standard intra-language link set. Table 2 lists the anchor accuracies for each anchor priority score and each θ. Anchor translation probability exhibited the best results and, therefore, we adopted anchor translation probability as the anchor priority score.

[1] Redirect pages and articles with no intra-language links were not included in these counts.

Anchor priority score	Anchor accuracy (%)			
	$\theta = 0$	$\theta = 0.01$	$\theta = 0.05$	$\theta = 0.1$
Anchor translation probability	**96.3**	**93.9**	**93.0**	**92.0**
Anchor probability	95.6	93.3	92.4	91.4
Destination article likelihood	90.7	90.8	91.5	91.3
Spelling	95.1	93.1	92.5	91.8
Position	88.2	87.3	87.9	87.6

Table 2. Anchor accuracy.

4.2 Experimental Results

We inputted 179,963 English intra-language links to the algorithm of the proposed method and classified them into the following five classes. Examples of each class, except class B, are given in Figure 2, which is an excerpt from the results for the pair of English article "Jacques Collin de Plancy" and Japanese article "コラン・ド・プランシー."

A. Transferred to a Japanese intra-language link in the gold standard (bold underline in Figure 2)

B. Transferred to a Japanese intra-language link whose anchor is not the same as the gold standard link to the same destination article

C. Transferred to a Japanese intra-language link not in the gold standard (double underline in Figure 2)

D. Transferred to a Japanese intra-language link without anchor (wavy underline in Figure 2)

E. Not transferred to a Japanese intra-language link (dashed underline in Figure 2)

Table 3 lists the numbers of English intra-language links per class. The proposed method added many new intra-language links to Wikipedia articles. Since the total number of existing Japanese intra-language links in the test-set articles was $T = 161{,}940$, the increase rate of Japanese intra-language links was $100(C + D)/T = 100(13{,}916 + 52{,}275)/161{,}940 = 40.9\%$ ($\theta = 0$). When new links without anchors were excluded, the increase rate was $100C/T = 100 \cdot 13{,}916/161{,}940 = 8.6\%$ ($\theta = 0$).

The anchor accuracy of existing links was $100A/(A + B) = 100 \cdot 31{,}770/(31{,}770 + 1{,}219) = 96.3\%$ ($\theta = 0$). Anchor accuracy of new intra-language links could not be calculated because of the unavailability of gold standard data. However, the proposed method specifies the anchor b for destination article q_d only when possible anchors for it is found in the target-language article q. The specified anchor b is likely to be the counterpart of source-language anchor a pointing to p_d that is the source-language counterpart of q_d, regardless of whether b already points to q_d or not. Thus, the anchor accuracy of new links should be similar to that of existing links.

Among the $S = 179{,}963$ input English intra-language links, $100D/S = 100 \cdot 52{,}275/179{,}963 = 29.0\%$ ($\theta = 0$) were transferred to Japanese intra-language links with the anchor unspecified. This was because different language articles contain different contents even though they are linked via an ILL. The anchor-unspecified links are put in the "関連項目" sections ("See also" sections) of target-language articles, and Wikipedia authors are expected to enlarge or revise the articles so that these anchor-unspecified links can be converted to anchor-specified links. Additionally, among the $S = 179{,}963$ input English intra-language links, $100E/S = 100 \cdot 80{,}783/179{,}963 = 44.9\%$ were not transferred to Japanese intra-language links. We assumed this was mainly due to missing Japanese articles. Note that the total number of Japanese articles is less than one-fourth that of English articles. The percentage of not-transferred links will decrease with the growing number of Japanese articles.

Jacques Collin de Plancy

Jacques Albin Simon Collin de Plancy (Plancy-l'Abbaye, 28 January 1793 –Paris, 1881) was a French occultist, demonologist and writer; he published several works on occultism and demonology.[1][2]

He was born **Jacques Albin Simon Collin** on 28 (in some sources 30) January 1793 in Plancy (presently Plancy-l'Abbaye) son of Edme-Aubin Collin and Marie-Anne Danton, sister of Georges-Jacques Danton who was executed the year after Jacques was born.[3] He later added the aristocratic "de Plancy" himself - an addition which would later cause accusations against his son in his career as a diplomat. He was a free-thinker influenced by Voltaire. He worked as a printer and publisher in Plancy-l'Abbaye and Paris. Between 1830 and 1837, he resided in Brussels, and then in the Netherlands, before he returned to France after having converted to the Catholic religion.

…

In 1818 his best known work, *Dictionnaire Infernal*, was published.

…

コラン・ド・プランシー

コラン・ド・プランシー（J. Collin de Plancy，1794 年〔一説には 1793 年〕 － 1881 年〔没年は 1887 年とも[1]〕）は、19 世紀に活躍したフランスの文筆家。

…

成人しパリ で教職などに就いていたが、
'Paris'
文筆家を志し、1818 年、彼自身の最大の代表作となる『地獄の辞典』初版を刊
'Dictionnaire Infernal'
行、以後積極的に著述に勤しむ。 『地獄の辞典』はその後もライフワーク的に改定が行われ、最終的にはオカルト関連
'occult'
の項目が 3, 799 に及ぶ大著となった。

…

学術的資料としては役に立たないばかりか、後世の悪魔学研究に混乱をきたさせ
'demonology'
るような部分も多い。

…

関連項目　'See also'

ブリュッセル　'Brussels'
ヴォルテール　'Voltaire'

Figure 2. Example results of transferring intra-language links.

	Transfer result		Number (percentage)			
	Destination	**Anchor**	$\theta = 0$	$\theta = 0.01$	$\theta = 0.05$	$\theta = 0.1$
A	Existing	Correct	31,770 (17.7%)	30,951 (17.2%)	30,661 (17.0%)	30,298 (16.8%)
B		Incorrect	1,219 (0.7%)	2,025 (1.1%)	2,298 (1.3%)	2,625 (1.5%)
C	New	Found	13,916 (7.7%)	12,812 (7.1%)	11,421 (6.3%)	10,335 (5.7%)
D		Not found	52,275 (29.0%)	53,392 (29.7%)	54,800 (30.5%)	55,922 (31.1%)
E	Not transferred		80,783 (44.9%)			

Table 3. English intra-language links classified according to results.

4.3 Additional Comments on Experimental Results

Among alternative anchor priority scores, anchor translation probability seems most effective because this is a posterior probability of the target-language counterpart to the source-language anchor. Anchor probability is also useful because this is a posterior probability of the anchor for the destination. Higher accuracy with spelling score indicates that Wikipedia editors tend to use the title of the destination as an anchor. This may be caused by manually specifying the anchor and destination independently. Contrary to expectations that the first occurrence likely becomes an anchor, position score exhibited the worst results. More detailed analysis of the context in which a term tends to be selected as an anchor is necessary.

Table 2 shows that the anchor probability, unexpectedly, decreases with a rise of the keyphraseness threshold. It was caused by articles that have only one possible anchor with keyphraseness value be-

low the threshold (e.g., "駅" – station). When the threshold was set high, the possible anchor set for such an article became empty and, as a results, the algorithm failed to reproduce the existing links to it.

In this experiment, we transferred English links onto Japanese articles. Since the English edition of Wikipedia is richer than Japanese, it has been assumed that an English-to-Japanese direction is more effective than the inverse. However, among the 179,963 links in English and 161,940 links in Japanese extracted from the test set of English-Japanese article pairs, only 32,989 links are paired with their counterparts and others do not have counterparts. This fact indicates that a Japanese-to-English transfer of links is also useful for enriching English articles. It also leads a low anchor recall, which is the percentage of correct links among existing links: $100A/T = 100 \cdot 31{,}770/161{,}940 = 19.6\%$ ($\theta = 0$). Combining usual wikification techniques should help improve the anchor recall.

5 Discussion

We now discuss two future directions, an extension to multiple language combination and a variation for inappropriate intra-language link detection.

The proposed method can be straightforwardly extended to three or more language combinations: Even if two source articles in two different languages are handled separately, the target article would be more enriched with the union of two transferred link sets. While this contributes to increasing the coverage of links, the reliability of links can also be improved by taking the intersection of the two transferred link sets. A more sophisticated combination of multiple source languages is a further problem.

In the experiment, existing links were used as the gold standard for evaluation, despite the fact that they are not always appropriate because they are manually created by unspecified contributors. For example, there is a biology-related article containing an anchor "translation" linking to an article "Translation" describing language translation, not to another article "Translation (biology)." Such an incorrect intra-language link may be detected using a similar method as the proposed one. In the above example, suppose the Japanese counterpart article contains an anchor "翻訳" linking to an article "翻訳 (生物学)." Two anchors "translation" and "翻訳" correspond to each other but their destination articles are not linked via an ILL. This inconsistency may be evidence for an inappropriate intra-language link. Note that which of the English and Japanese links is inappropriate cannot be easily determined. How to estimate the appropriateness of intra-language links is a problem to be solved.

6 Related Work

Wikification, which aims at linking mainly non-Wikipedia articles to Wikipedia articles, can be naturally applied to linking between Wikipedia articles. There has been much research on wikification, most of which focused on disambiguation of destination articles (Milne and Witten, 2008a; Fogarolli, 2009; Ratinov et al., 2011). Determining an appropriate destination article for an anchor term is a special case of WSD. Although a variety of ideas for WSD have been adapted to wikification, their performance is not satisfactory and there is room for further improvement. Another important issue with wikification is anchor selection, although most literature on wikification avoids the issue by selecting every term that is used as an anchor in any Wikipedia article. Anchor selection is a keyword extraction problem, which has been tackled using syntactic, statistical, and/or machine learning techniques but remains room for further improvement (Jacquemin and Bourigault, 2003). It should be added that our proposed method avoids both disambiguation and anchor selection problems by exploiting link information in another language edition of Wikipedia.

Adafre and de Rijke (2005) proposed a method for finding "missing intra-language links" in a Wikipedia article by assuming that an intra-language link represents the relatedness between concepts described by the linked articles. Their method adds intra-language links to an article by using articles with similar link structures as that of the article in question. Similar methods that use the Wikipedia's link structures as a semantic network have been proposed for entity linking (Milne and Witten, 2008b; Fogarolli, 2009; Ratinov et al., 2011). These still remain monolingual methods; the availability of other language editions cannot be assumed.

A bilingual approach to improving quality of Wikipedia articles has also been studied. Sorg and Cimiano (2008) proposed a method for finding new ILLs by using a classifier whose features include

the number of ILLs between articles pointed by an article in one language and those pointed by an article in another language. Wang et al. (2013) improved the classifier by extending the intra-language links to increase the number of features. Both methods and our proposed method exploit the comparability between intra-language links in different language editions. However, while the former find new ILLs, the latter finds new intra-language links.

7 Conclusion

We proposed a method for enriching intra-language links in Wikipedia articles. It transfers intra-language links between a pair of different language articles linked by an inter-language link through the following two steps: first, determine destination articles to which the target-language article should be linked by following a source-language intra-language link and an ILL successively from each of the anchors in the source-language article; second, determine an anchor for each of the destination articles by searching the target-language article for possible anchors and selecting the most appropriate one according to the anchor translation probability criterion if two or more possible anchors are found. Unlike usual wikification methods, our method avoids anchor selection and disambiguation problems by exploiting the comparability of intra-language links between different language editions of Wikipedia.

We conducted an experiment of transferring intra-language links from the English edition to the Japanese edition to evaluate the effectiveness of our method. The method increased the number of intra-language links in Japanese articles by 40.9%, and the accuracy of anchors selected was estimated to be 96.3%. Future work includes an extension to multiple language combination and a variation for inappropriate intra-language link detection.

References

Adafre, Sisay Fissaha and Maarten de Rijke. 2005. Discovering missing links in Wikipedia. In *Proceedings of the 3rd International Workshop on Link Discovery: Issues, Approaches and Applications*, pages 90–97.

Fogarolli, Angela. 2009. Word sense disambiguation based on Wikipedia link structure. In *Proceedings of 2009 IEEE International Conference on Semantic Computing*, pages 77–82.

Gale, William A., Kenneth W. Church, and David Yarowsky. 1992. One sense per discourse. In *Proceedings of HLT '91 Workshop on Speech and Natural Language*, pages 233–237.

Jacquemin, Christian and Didier Bourigault. 2003. Term extraction and automatic indexing. In Ruslan Mitkov (Ed.), *The Oxford Handbook of Computational Linguistics*, pages 599–615. Oxford University Press.

Levenshtein, Vladimir I. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Cybernetics and Control Theory*, 10(8):707–710.

Mihalcea, Rada and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, pages 233–242.

Milne, David and Ian H. Witten. 2008a. Learning to link with Wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 509–518.

Milne, David and Ian H. Witten. 2008b. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceedings of the Wikipedia and AI Workshop of AAAI*, pages 25–30.

Ratinov, Lev, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1375–1384.

Sorg, Philipp and Philipp Cimiano. 2008. Enriching the crosslingual link structure of Wikipedia - a classification-based approach. In *Proceedings of the AAAI 2008 Workshop on Wikipedia and Artificial Intelligence*.

Wang, Zhichun, Juanzi Li, and Jie Tang. 2013. Boosting cross-lingual knowledge linking via concept annotation. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 2733–2739.

Chinese Irony Corpus Construction and Ironic Structure Analysis

Yi-jie Tang and Hsin-Hsi Chen
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan
tangyj@nlg.csie.ntu.edu.tw,hhchen@ntu.edu.tw

Abstract

Non-literal expression recognition is a challenging task in natural language processing. An ironic expression implies the opposite of the literal meaning, causing problems in opinion mining and sentiment analysis. In this paper, ironic messages are collected from microblogs to form an irony corpus based on the use of emoticons, linguistic forms, and sentiment polarity. Five linguistic patterns are mined by using the proposed bootstrapping approach. We also analyze the linguistic structure and elements used to convey irony. Based on our observations, ironic words/phrases and contextual information are the necessary elements in irony, while the contextual information can be hidden in linguistic forms. A rhetorical element, which is optional in irony, can also be used to help strengthen the effects and understandability of an ironic expression. The ironic elements in each instance of our irony corpus are labelled based on this structure. This corpus can be used to study the usage of ironic expressions and the identification of ironic elements, and thus improve the performance of irony recognition.

1 Introduction

Dealing with non-literal meaning is a challenging task in natural language processing. Linguistic context and background knowledge are required to interpret non-literal utterances properly. An ironic expression, where the meaning is the opposite of what is literally expressed, is one of the indirect and non-literal linguistic forms that cannot be easily processed and detected. One cannot capture the real meanings of opinions and sentiments expressed in a document or conversation if irony is not taken into account.

The challenges of irony processing involve the following issues: (1) No comprehensive irony corpus is available. (2) Irony analysis is related to semantics, pragmatics and discourse studies, which are the most challenging in natural language processing. (3) Contextual information and background knowledge are necessary, but they are hard to obtain and process. (4) Non-linguistic or non-verbal factors, e.g., intonations, gestures and talking speed in speech, and spaces, punctuations and typography in writing, have to be considered.

This paper focuses on irony corpus construction, ironic pattern mining, and ironic structure analysis. Messages were collected from a microblogging platform based on emoticons, and ironic messages and patterns were extracted to build an irony corpus. The structure of ironic expressions and the clarification of the uses of ironic elements were also analyzed. Labels representing the ironic elements are added to each message in the irony corpus. To the best of our knowledge, this is the first Chinese irony corpus available for research.

This paper is organized as follows. Section 2 surveys the related work. Section 3 proposes a methodology to construct an irony corpus. Section 4 presents the patterns mined from the corpus. Section 5 discusses the results of ironic expressions collected from a different type of corpus. Section 6 makes the error analysis. Section 7 analyzes linguistic structure of Chinese irony. Section 8 concludes the remarks.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1269–1278, Dublin, Ireland, August 23-29 2014.

2 Related Work

Sarcasm and irony have been studied by linguistics and cognitive scientists (Giora and Fein, 1999; Gibbs and Colston, 2007) for years, but there has been no concrete claim on the linguistic structure of irony. Some studies have started focusing on the processing of sarcasm and irony recently, but it is still not clear whether sarcasm and irony differ significantly or represent the same concept.

The research of non-literal expression identification has drawn attention in recent years. Katz and Giesbrecht (2006) use meaning vectors for literal and non-literal expression classification. Li and Sporleder (2010) focus on distinguishing literal and non-literal usages of idioms.

Filatova (2012) uses crowdsourcing to generate an irony and sarcasm corpus. Veale and Hao (2010) construct a corpus of ironic similes using the wildcarded query "as * as a *" on a search engine. Davidov et al. (2010) collect messages from Twitter and product reviews from Amazon.com using the Mechanical Turk service. The #sarcasm hashtag is used as ground truth, and a k-nearest neighbor strategy is used for classification. González-Ibáñez et al. (2011) also make use of hashtags in Twitter as labels to build a sarcasm corpus. In their study, both human classification and automatic classification achieve low accuracy in sarcasm detection. Reyes et al. (2012) analyze humor and irony based on the user-generated tags, such as "#humor" and "#irony", in twitter. Lukin and Walker (2013) use a bootstrapping method to improve the performance of the classifiers for identifying sarcastic and nasty utterances in online dialogues.

The hashtag-based approaches are not always suitable for irony corpus construction for all the languages. As of March 9, 2014, only 113 messages are found to contain the hashtag #反諷 (#irony) in Weibo, the largest Chinese language microblogging platform. This paper differs from the previous work in that we employ negative emoticons and positive words as clues to capture the irony. The linguistic patterns mined from the irony corpus can be used to detect if a sentence is ironic.

3 Irony Corpus Generation from Microblogs

This section introduces a bootstrapping methodology to construct an irony corpus and mine irony patterns. While Lukin and Walker (2013) also used a bootstrapping method to improve sarcasm and nastiness classifiers, this paper, in contrast, focuses on irony pattern mining and corpus construction.

3.1 An Emotion-Tagged Corpus

The traditional definition of verbal irony is adopted, where the speaker says something that seems to be the opposite of what they mean (Gibbs and Colston, 2007). Under this definition, texts annotated with polarity information that expresses the actual meaning should be collected, and the literal meanings of words in the texts should be identified. If any disagreement exists between the actual meaning and literal meaning, then we say the text contains irony.

Nowadays, emoticons are used quite often in social media to express the feelings of the posters. The tagged emoticons specify their actual meanings in some sense. Based on this idea, messages were collected from Plurk[1], a microblogging platform similar to Twitter. It lets users post messages limited to 140 characters, and allows them to use graphical emoticons in their messages.

It was assumed that these emoticons can represent the poster's sentiments, and, therefore, be regarded as sentiment labels of the messages. Among 35 emoticons, 23 are categorized into positive, and 12 are categorized into negative. Collected messages are dated from Jun 21, 2008 to Nov 7, 2009, and all of them are in Traditional Chinese.

On the other hand, the literal meanings of the posted messages need to be known. Many sentiment analysis algorithms (Liu, 2012) can be explored. A lexicon-based approach was adopted. The NTU Sentiment Dictionary, or NTUSD (Ku and Chen, 2007), was employed to determine the sentiment of a word. This dictionary provides 21,056 positive and 22,751 negative words. Most of these words are in Traditional Chinese.

[1] http://www.plurk.com

3.2 Candidates Extraction

Possible irony messages were extracted from the Plurk corpus by using NTUSD. Since the typical social function of irony is expressing negative meaning with positive words, as mentioned in Gibbs and Colston (2007), focus was directed on those messages with negative emoticons and positive words. A total of 3,178,372 messages was found containing at least one negative emoticon. Among them, 304,754 messages with at least one positive word are found and form an irony candidate dataset.

Discourse relation determines how two discourse units cohere to each other. Sentiment transition of two clausal arguments is identified based on their discourse relation (Zhou et al., 2011; Wang et al., 2012; Huang et al., 2013). In the sentence "he is nice but not attractive," positive opinion in the beginning is transformed to a negative one by the discourse connective "but." Both the positive word "nice" and the negative phrase "not attractive" are used literally. Thus, it was necessary to filter out messages containing such connectives.

Messages are removed only when the positive word occurs earlier than the discourse connectives with a comparison function, due to Chinese grammatical structure. The Chinese discourse connectives used here include "但", "但是", "可是", "只是", "不過" (all the above are equivalent to the English word *but*), "然而" (however), "卻" (comparatively), "可惜" (unfortunately), "偏偏" (contrarily), "反而" (oppositely), and "倒是" (on the contrary). A total of 254,836 messages remains after this process.

3.3 Pattern Mining

Although irony can be used without any customary linguistic patterns, some ironic expressions do exhibit specific forms of language use. Colston and O'Brien (2000) suggest that both irony and hyperbole create contrasts between expected and ensuing events. It was assumed that exaggerated expressions could be used with irony to strengthen the effects of the speech act. In the expression 我真是太幸運啦！ (I am really and extremely lucky!), the adverbs *really* and *extremely* are used to strengthen the ironic effect. Thus, combinations of degree adverb phrases and a positive adjective are used as patterns to find possible irony expressions automatically in the candidate dataset.

Not all degree adverbs in Chinese are used because some of them are mostly used in formal texts and not frequently present in microblogs. The degree adverb phrases used here include the combinations of the adverbs "還" (*hái*), "也" (*yě*), "未免" (*wèimiǎn*), "可" (*kě*) and "實在" (truly) and the degree adverbs "真" (really), "太" (extremely) and "非常" (very).

The following bootstrapping procedure was used to find more patterns.

(1) Which patterns should be used is decided. At the very beginning of the bootstrapping procedure, the [degree adverb + positive adjective] pattern mentioned above is used.

(2) Messages containing the patterns in step (1) are automatically retrieved from the candidates. NTUSD is used to determine sentiment polarity, and CKIP parser is used to get parts of speech[2].

(3) Messages retrieved in step (2) were reviewed by the annotator to decide which of them are actually ironic.

(4) If the annotator finds new irony patterns in the reviewed messages, then the procedure starts again from step (1) and uses the patterns to repeat the process.

This process was repeated for four times. After the fourth iteration, no more new patterns were found by the annotator. Finally, 2,825 messages are found to have any of the patterns, and 1,005 of them are confirmed to be ironic and make up the NTU Irony Corpus.[3] Examples of these patterns and ironic messages are shown in Section 4.

4 Irony Patterns

All the patterns mined by the approach used in Section 3 are categorized into the following five groups.

4.1 Degree Adverbs + Positive Adjective

In this pattern, the following two components must exist:

[2] http://ckipsvr.iis.sinica.edu.tw.

[3] The NTU Irony Corpus is available at http://nlg.csie.ntu.edu.tw/nlpresource/irony_corpus/.

(a) Degree adverb phrase + positive adjective phrase
(b) Negative context

The negative context can occur either before or after the component (a). For example, the following expression is used when someone has to wait for a long time to start ordering in a restaurant. In total, 13.03% of all the messages in the corpus contain this pattern.

(s1) 點餐都要等半小時，服務還真是好阿
I have to wait for half an hour to order. The service is definitely really good.

The underlined expression is the contextual information described in (b), and the double-underlined expression is the linguistic form described in (a). In the second clause the adverbs "還" (*hái*) and "真" (really) are combined to form a degree adverb phrase for intensification or hyperbole. Although the positive word *good* is used, the speaker means the opposite. The first clause indicates why they think the service is not good, and, therefore, provides the contextual information.

4.2 The Use of Positive Adjective with High Intensity

In this pattern, the following two components must exist:

(a) Positive adjective with high intensity
(b) Negative context

Specific positive adjectives with high intensity are used to form ironic expressions with or without other rhetorical elements. Since the context is negative, the positive adjective is used to express non-literal meanings. The adjectives we found in the corpus include "偉大" (*great*), "了不起" (r*emarkable*) and "天才" (*genius*). Only 2.09% of the messages in the corpus contain this pattern. For example, the word *great* is used in the following message:

(s2) 我的 plurk「又」發生不明錯誤了...這真是這世紀最偉大的發明啊
My Plurk account encountered an unknown error 'again'... This is indeed the greatest invention in the century.

4.3 The Use of Positive Noun with High Intensity

In this pattern, the following two components must exist:

(a) Positive noun with high intensity
(b) Negative context

Specific nouns that represent highly positive meanings are also used to express irony. These nouns include "巨星" (*superstar*), "大禮" (*big gift*) and "境界" (*wonderful state*). When they are used with a negative context, an ironic expression is formed. This is pattern is not found frequently in the corpus. Only 2.00% of the messages in the corpus contain this pattern. An example is listed below:

(s3) 中秋節收到的大禮是.....長了一堆肉
The big gift I received in the Mid Autumn Festival was…… a lot of fat in my body.

4.4 The Use of "很好" (very good)

In this pattern, the following two components must exist:

(a) Sentence boundary + 很好 + punctuation
(b) Negative context

A sentence boundary occurs before the word "很好" (very good) because there is no subject. Multiple punctuations, and particularly exclamation marks and ellipses, can be used after "很好" to increase the intensity. In the following example, exclamation marks are used:

(s4)感冒... 很好!! 我的假期飛了
I caught a cold... Very good!! My vacation is gone.

Sometimes this pattern is followed by an exclamation word, such as "啊" (*a*), "呀" (*ya*), and "嘛" (*ma*). These exclamations, like punctuations, can help strengthen the level of the speaker's feelings. In our irony corpus, this pattern is used in 50.84% of all ironic messages. Obviously, this is a common way when people want to express their negative feelings with an ironic expression.

4.5 "可以再…一點" (It's okay to be worse)

In this pattern, the following expression must exist:

可以再 + negative adjective + 一點
(It is okay to be more + negative adjective)

This pattern literally states that it is okay for something to become worse and is a commonly used pattern to express irony in our corpus. It can be found in 33.53% of the messages in the corpus. In most cases, even when no proper contextual information is present, the listener can tell the literal meaning is not meant because it violates most people's inclinations. Thus, the use of this pattern is usually non-literal and ironic. An example is shown below.

(s5)零下十一度...你可以再冷一點
It's -11°C...It is okay to be colder

A message can contain more than one pattern, causing the sum of the percentages of the above five patterns to be greater than 100%. For example, both patterns 4.4 and 4.5 are used in the following message:

(s6)很好!!!!我可以再笨一點 再笨一點阿...
Very Good!!!! It is okay for me to be more idiotic…

The patterns in Sections 4.4 and 4.5 are mainly based on their linguistic forms and frequently used in ironic expressions. We argue that these patterns are more static than the others, and we call them the *customary patterns*. On the other hand, the patterns in Sections 4.1, 4.2 and 4.3 are called *non-customary patterns*.

5 Collecting Ironic Expressions from Blogs

In order to understand how irony is conveyed in different types of media, we use the methodology and mined patterns described in Sections 3 and 4 to collect irony expressions from the Yahoo Kimo Blogs corpus.

5.1 The Yahoo Blog Corpus

The Yahoo Kimo Blog corpus, referred to the Yahoo corpus in the following sections, contains blog articles from November 1, 2005 to August 20, 2007 (Yang, Lin and Chen, 2009). Out of all the posts in the dataset, 2,764,202 posts have at least one emoticon. The articles posted in July 2006 are used here, and they are divided into 341,932 smaller units by the full stop symbol. All articles are in Traditional Chinese.

Since the Plurk platform can be used as an instant messaging system, and readers of the message are usually on the author's friend list, these messages are usually conversational. On the other hand, Ya-

hoo blogs are not limited in length and a blog article itself is not part of the conversation. Thus, the blog articles are usually more formal compared to microblog messages.

Although the articles are separated by a full stop into shorter units, these units are not necessarily identical to sentences due to the conventional usage of the Chinese period symbol. They can consist of multiple sentences and thus contain a discourse structure, which makes them suitable for this corpus study.

5.2 Extract Ironic Expressions

A similar approach to the steps described in Section 3.3, is used to collect ironic expressions from the Yahoo corpus, but four patterns of irony found in Plurk are used to perform step (1). These patterns, as listed below, are adopted because they are the most frequently used ones in our Plurk irony corpus. They can also reflect the uses of customary and non-customary irony patterns as the first two patterns are customary, and the last two are non-customary. Pattern 1 and Pattern 2 are the same patterns as mentioned in Section 4.4 and Section 4.5, respectively. Pattern 3 and Pattern 4 are two forms from the pattern described in Section 4.1. Only step (1) to step (3) are performed, and step (4) is bypassed; that is, the process is not repeated.

Pattern 1:
(a) Sentence boundary + 很好 + punctuation
(b) Negative context
Pattern 2:
可以再 + negative adjective + 一點
Pattern 3:
(a) 還真 + positive adjective
(b) Negative context
Pattern 4:
(a) 真是 + positive expression
(b) Negative context

5.3 Results and Discussion

A total of 36 ironic texts is obtained. All the four irony patterns seen in Plurk can be found in Yahoo. The final results are shown in Table 1.

	Number of Ironic Expressions	Percentage
Pattern 1	14	38.89%
Pattern 2	10	27.78%
Pattern 3	5	13.89%
Pattern 4	7	19.44%

Table 1: Ironic texts found for the four Patterns in Yahoo.

The proportions of the four patterns in Plurk and Yahoo are also compared. The percentages are calculated by dividing the occurrence of each pattern by the occurrence of all four patterns in the same datasets. As can be seen in Figure 1, the proportions of patterns (1) and (2) in Plurk are significantly higher than in Yahoo, and the proportions of patterns (3) and (4) in Plurk are significantly lower than in Yahoo ($p<0.05$ according to the t-test). This suggests that patterns (1) and (2) tend to be used in informal and conversational texts while patterns (3) and (4) tend to be used in formal articles to convey irony. Also, this may suggest that customary patterns are more likely to be used in conversations, and authors of formal articles prefer an indirect way to express irony, although more data are required for further studies in the future.

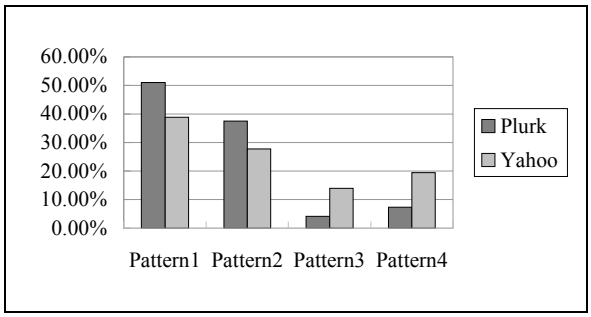


Figure 1: Comparison of the proportion of the four patterns in Plurk and Yahoo.

6 Error Analysis

In this section, we analyze why non-ironic messages were retrieved by the automatic processes. The 1,820 wrong messages specified in Section 3.3 are classified into the following two categories.

(1) Sentiment identification

Using the patterns to find possible ironic messages involves the correct sentiment identification. NTUSD does not cover some new words used on Internet informal conversations. The sentiment of a word can also be changed depending on its context. For example, "太強" (so strong) is listed as a positive term in NTUSD. However, it is used to indicate a negative condition in the example (s7).

(s7) 止痛藥的副作用也太強了吧，昏睡一整晚
The *side effect of the pain reliever was so strong, making me sleep through the whole night.*

(2) Opinion targets

In a Plurk message, even though the message poster is talking about the same topic, more than one entity with associated opinions can be present. For example:

(s8) 最近公司生意很好，好累ㄛ
The business of our company is running so well. I am so tired.

The poster expresses negative sentiment by using the word "tired." Although the positive word "很好" (very good) is also used, it modifies the word "business" rather than the poster's condition. That is, the opinion targets of the two words are different, and this causes problems when automatically retrieving ironic messages.

7 Linguistic Structure of Irony

In this section, the linguistic structure of irony is analyzed based on our observations on the corpus.

7.1 Ironic Word

As described, the literal meaning of an ironic word or phrase is opposite to the actual meaning. An ironic word/phrase is necessary to separate irony from regular utterances. If the ironic word of an utterance is reverted, the speaker's actual sentiment or intention is reconstructed.

However, it is not easy to identify the ironic word in an utterance. Sometimes more than one word can be an ironic word. In our corpus, 94.93% of the ironic words are adjectives, while others are used as adverbs, verbs or nouns. The recognition of ironic word/phrase is a challenging task, but other ironic elements described in Sections 7.2 and 7.3 can be analyzed side by side to help improve the performance.

7.2 Contextual Information

Contextual information is usually provided as part of ironic utterances to help convey irony. For example, the underlined sentence in the following utterance is crucial for irony interpretation:

(s9) 我掛彩了，真是太好運了
I was injured. I was really lucky.

Without the first sentence, it is hard to tell if *lucky* is actually meant. Although a speaker can still use ironic words/phrases without providing contextual information, this can be an ineffective way to communicate the actual meanings of irony. According to the cooperative principle proposed by Grice (1975), the speaker must give enough information in order to enable successful communication and implicatures. The four maxims of the cooperative principle include:

(1) **Maxim of Quantity**: The speaker should make their contribution as informative as is required. Do not make the contribution more informative than is required.
(2) **Maxim of Quality**: The speaker should not say what they believe to be false, and should not say that for which they lack adequate evidence.
(3) **Maxim of Relation**: The speaker should be relevant.
(4) **Maxim of Manner**: The speaker should avoid obscurity of expression, avoid ambiguity, be brief and be orderly.

Based on Grice's maxims, it is assumed *enough, correct, relevant,* and *understandable* contextual information should be provided with ironic expressions. However, the speaker sometimes assumes the listener already knows about the conditions where the irony takes place and has the required background knowledge; thus the contextual information is hidden in the ironic utterance.

Four types of context can be used to interpret irony:

(1) **Linguistic context**: The linguistic context refers to the words that are expressed before and/or after the irony words in a sentence or discourse. It is easier to obtain and analyze than the other three types of context.
(2) **Physical context**: Physical context refers to what is actually present and/or happening in the environment or circumstance where the conversation is taking place. It is also related to the timing. In online conversations, participants are not usually in the same location, but they can be aware of the same ongoing events and situations. It is not necessary for the speaker to provide physical context information if they assume the objects or situations are noticeable to the listeners.
(3) **Epistemic context**: The background knowledge shared by the participants in a conversion can also be used to interpret the irony. This type of context does not change over time. For example, people know rocks are hard, so they can understand the expression *the bed is as soft as a rock* is not literal.
(4) **Social context**: Social relationship can be important for expressing and interpreting irony, especially in online messages.

We argue that at least one type of contextual information must exist, but it can be hidden if the speaker thinks the listener is already aware of it. Physical, epistemic and social context can be hidden, while linguistic contextual information must be present.

7.3 Rhetoric

As shown in Section 4, degree adverbs, punctuations and exclamations can be used to convey irony. Some of them can even be repeated to intensify the effects. These elements increase contradiction and strengthen the degree of negative opinions. Unlike ironic words and context, rhetoric elements are not necessary to convey irony.

Liebrecht et al. (2013) call the words used to strengthen evaluative utterances *intensifiers*. In their experiments, non-hyperbolic sarcastic messages often contain an explicit marker on Twitter. They argue that sarcasm is often signaled by hyperbolic words, including intensifiers and exclamations, and sarcastic utterances with hyperbolic words are easier to identify by listeners/readers than sarcastic utterances without hyperbolic words. It can be seen that adverbs, adjectives, punctuations and exclamations with high intensity observed in our irony patterns have very similar effects.

Among the 113 messages containing the #反諷 (#irony) hashtag in Weibo, which was mentioned in Section 2, 83.19% do not exhibit hyperbole or uses of intensifiers. This observation is similar to the argument suggested in Liebrecht et al. (2013) and is one of the reasons why the hashtag is not suitable

for the irony pattern mining task in this study. In comparison, this methodology helps find more clues of irony that can be seen from their linguistic forms.

7.4 Corpus Labeling

To increase the usefulness of the corpus, ironic element tags are added to each message. An example is shown in Figure 2.

<context sentiment="pos">才剛買的書，竟然掉頁了，</context>這品質<rhetoric>也太</rhetoric><ironic sentiment="neg">好</ ironic>了<rhetoric>吧</rhetoric>.

English translation:
<context sentiment="pos">The book I just bought has fallen apart.</context> The quality is <rhetoric>just extremely</rhetoric> <ironic sentiment="neg">good</ ironic>*le*<rhetoric>*ba*</rhetoric>.

Figure 2: An example message with ironic element tags.

As can be seen in the example, "好" (good) is the word that is used in the opposite way, so it is marked with the ironic word/phrase label <ironic>. The preceding sentence states what actually happened, and is marked with the label <context>. The message poster also uses the degree adverb "太" (extremely) and used the exclamation "吧" (*ba,* a sentence-final partical). These two words are marked with the <rhetoric> label. The sentiment polarity marks of the ironic word and contextual information, shown as either *pos* or *neg*, are also added.

8 Conclusion

In this paper, five types of irony patterns are mined, and an irony corpus is constructed based on linguistic forms and sentiment classification. Four verbal forms in Plurk and Yahoo were further examined. The former platform restricts short text conversation, and the latter platform allows for the long text description. The experimental results show that the customary forms tend to be used in informal and conversational texts while the non-customary forms tend to be used in formal articles to convey irony. The three basic elements that form a successful ironic speech act were also analyzed. These elements, including the words/phrases with reversed meanings, contextual information and rhetorical words, should be identified first in order to properly process ironic expressions and perform linguistic analysis. In the mined patterns, it was found that hyperbole was frequently present. In future work, we will explore other opinion mining and sentiment analysis algorithms, and focus on automatic recognition of hyperbole and the ironic elements.

Acknowledgements

This research was partially supported by National Taiwan University under grant 103R890858. We are also very thankful to the anonymous reviewers for their helpful comments to revise this paper.

References

Herbert L. Colston and Jennifer O'Brien. 2000. Contrast of Kind Versus Contrast of Magnitude: the Pragmatic Accomplishments of Irony and Hyperbole. *Discourse and Processes*, 30(3):179-199.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon, In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning* (*CoNLL-2010*), pages 107-116, Uppsala, Sweden.

Elena Filatova. 2012. Irony and Sarcasm: Corpus Generation and Analysis Using Crowdsourcing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation* (*LREC 2012*), pages 392-298, Istanbul, Turkey.

Raymond W. Gibbs and Herbert L. Colston. 2007. *Irony in Language and Thought*. Lawrence Erlbaum Associates, New York.

Rachel Giora and Ofer Fein. 1999. Irony: Context and Salience. *Metaphor and Symbol,* 14:241-257.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying Sarcasm in Twitter: A Closer Look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Short Papers*, pages 581-586, Portland, Oregon, USA.

H. P. Grice. 1975. Logic and Conversation. In P. Cole and J. J. Morgan, eds. *Syntax and Semantics, 3: Speech Acts.* New York: Academic Press.

Hen-Hsen Huang, Chi-Hsin Yu, Tai-Wei Chang, Cong-Kai Lin and Hsin-Hsi Chen. 2013. Analyses of the Association between Discourse Relation and Sentiment Polarity with a Chinese Human-Annotated Corpus. In *Proceedings of ACL 2013 7th Linguistic Annotation Workshop & Interoperability with Discourse*, pages 70-78, Sofia, Bulgaria.

Graham Katz and Eugenie Giesbrecht. 2006. Automatic Identification of Non-compositional Multi-word Expressions Using Latent Semantic Analysis. In *Proceedings of the ACL/COLING-06 Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, pages 12-19, Sydney, Australia.

Lun-Wei Ku and Hsin-Hsi Chen. 2007. Mining Opinions from the Web: Beyond Relevance Retrieval. *Journal of American Society for Information Science and Technology, Special Issue on Mining Web Resources for Enhancing Information Retrieval,* 58(12):1838-1850.

Linlin Li and Caroline Sporleder. 2010. Linguistic Cues for Distinguishing Literal and Non-Literal Usages. In *Proceedings of 23rd International Conference on Computational Linguistics* (*COLING 2010*), *Poster Volume*, pages 683-691, Beijing, China.

Christine Liebrecht, Florian Kunneman, and Antal van den Bosch. 2013. The Perfect Solution for Detecting Sarcasm in Tweets #not. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 29-37, Atlanta, Georgia.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers.

Stephanie Lukin and Marilyn Walker. 2013. Really? Well. Apparently Bootstrapping Improves the Performance of Sarcasm and Nastiness Classifiers for Online Dialogue. In *Proceedings of the Workshop on Language Analysis in Social Media,* pages 30-40, Atlanta, Georgia.

Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From Humor Recognition to Irony Detection: The Figurative Language of Social Media. *Data & Knowledge Engineering*, 74:1-12.

Tony Veale and Yanfen Hao. 2010. Detecting Ironic Intent in Creative Comparisons. In *Proceedings of the 19th European Conference on Artificial Intelligence* (*ECAI 2010*)*,* pages 765-770, Lisbon, Portugal.

Fei Wang, Yunfang Wu, and Likun Qiu. 2012. Exploiting Discourse Relations for Sentiment Analysis. In *Proceedings of the 24th International Conference on Computational Linguistics* (*COLING 2012*)*: Posters*, pages 1311-1320, Mumbai, India.

Changhua Yang, Kevin Lin, and Hsin-Hsi Chen 2009. Writer Meets Reader: Emotion Analysis of Social Media from both the Writer's and Reader's Perspectives. In *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence* (*WI 2009*), pages 287-290, Milan, Italy.

Lanjun Zhou, Binyang Li, Wei Gao, Zhongyu Wei, and Kam-Fai Wong. 2011. Unsupervised Discovery of Discourse Relations for Eliminating Intro-Sentence Polarity Ambiguities. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing* (*EMNLP 2011*), pages 162-171, Edinburgh, Scotland, UK.

Global Methods for Cross-lingual Semantic Role and Predicate Labelling

Lonneke van der Plas
Institute for NLP
Pfaffenwaldring 5B
70569 Stuttgart, Germany
vdplasme@ims.uni-stuttgart.de

Marianna Apidianaki
LIMSI-CNRS
Rue John von Neumann
91405 Orsay Cedex, France
marianna@limsi.fr

Chenhua Chen
Institute for NLP
Pfaffenwaldring 5B
70569 Stuttgart, Germany
cch.chenhua@googlemail.com

Abstract

We address the problem of transferring semantic annotations to new languages using parallel corpora. Previous work has transferred these annotations on a token-to-token basis, an approach that is sensitive to alignment errors and translation shifts. We present a global approach to transfer that aggregates information across the whole parallel corpus and leads to more robust labellers. We build two global models, one for predicate labelling and one for role labelling, each tailored to the task at hand. We show that the combination of direct and global methods outperforms previous results.

1 Introduction

With the proliferation of the Internet in non-English speaking countries, the need for multilingual processing becomes more and more pressing. Various efforts have focused on developing language-independent NLP tools and extending to other languages tools that had been exclusive to English. Furthermore, several annotation efforts have been devoted to developing resources for different languages, needed for supervised learning (Hajič et al., 2009). However, there is still a large number of languages for which corpora with semantic annotations do not exist. Since manual annotation is a costly and time-consuming approach to resource development, cross-lingual annotation transfer offers an alternative.

Semantic parsing or semantic role labelling (SRL) is the task of automatically labelling predicates and arguments with predicate-argument structure. This level of analysis provides a more stable semantic representation across syntactically different sentences. The example sentences (1a) and (1b) illustrate how the semantic annotation remains stable across the locative alternation of the verb *load*.

(1) a. [AGENT Jessica] [REL-LOAD.01 loaded] [THEME boxes] [DESTINATION into the wagon].
b. [AGENT Jessica] [REL-LOAD.01 loaded] [DESTINATION the wagon] [THEME with boxes].

Also in the cross-lingual setting, the predicate-argument structure of a sentence is considered to be more stable than its syntactic form as the English sentence in (2a) and its French translation in (2b) show:

(2) a. [EXPERIENCER Mary] [REL-LIKE.01 liked] [CONTENT the idea]. (English)
b. [CONTENT L'idée] a [REL-LIKE.01 plu] [EXPERIENCER à Marie]. (French)

This is why several pieces of work have transferred semantic annotations from a source language, for which semantic annotations exist, to a target language using parallel corpora (Padó, 2007; Basili et al., 2009; Annesi and Basili, 2010). These transfer methods rely on the assumption of semantic equivalence of the original and the translated sentences, but also on correct and complete alignments between words

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1279–1290, Dublin, Ireland, August 23-29 2014.

or constituents in those sentences. We will refer to these traditional methods as direct transfer because the semantic annotations are transferred directly from token to token. Although direct transfer methods are straightforward and easy to implement, they are vulnerable to missing or incorrect alignments which lead to missing and erroneous annotations in the target language. Consequently, non-literal translations and translation shifts present major problems for these methods.

In this paper we propose a global approach to the cross-lingual transfer of PropBank (Palmer et al., 2005) semantic annotations that aggregates information at the corpus level and, as a consequence, is more robust to non-literal translations and alignment errors. Our global approach involves two steps: in the learning step, two global models are learned on the basis of role and predicate annotations in the source language (English). In the labelling step, these models assign labels to verbs and their arguments in the target language (French) without consulting any parallel data. Contrary to previous work, we build separate models for the transfer of semantic role and predicate annotation because predictors for the two models are different in nature. We model cross-lingual transfer of predicate labels as a cross-lingual word sense disambiguation (WSD) task because this fits well with the lexical nature of the task: annotating French verbs with English predicate labels. Our approach to predicate labelling needs word alignments but instead of relying on local (token-to-token) correspondences like the direct method, it exploits alignment information gathered from the whole corpus thus avoiding transfer errors caused by local misalignments. Our model for cross-lingual semantic role labelling[1] is based on syntactic-semantic mappings learned from a gold annotated monolingual corpus. The SRL method does not need aligned data. Our methods are knowledge-lean as our predicate labelling method only needs a part of speech (PoS) tagger in the two languages and no syntactic information on either side, in contrast to previous work. For SRL, a syntactic parser for the target language is needed, but no joint semantic-syntactic parsing framework as was the case in previous work (van der Plas et al., 2011). The requirements of the global annotation transfer methods in terms of data and annotations, and their differences from direct transfer, are illustrated in Figure 1.

Our contribution is three-fold. First, we present a global approach to semantic annotation transfer that corrects token-level mistakes found in traditional direct transfer methods. We show the strengths and limitations of global vs. direct transfer and explain how the two can be combined. Second, in contrast to previous work, we address the two tasks of cross-lingual predicate labelling and cross-lingual semantic role labelling by building two separate models tailored to the task at hand. We show how the predicate labels produced by our high-coverage and knowledge-lean model for cross-lingual predicate labelling are successfully used as predictors for semantic role labelling. Third, due to its knowledge-lean and flexible character, our method adapts relatively easily to other language pairs without requiring semantic lexicons in the target language.

In the next section, we present related work on cross-lingual annotation transfer. In Section 3 we present the data used in our experiments and in Section 4 we briefly discuss direct transfer. The two global methods proposed in this paper are presented in Section 5. We report and discuss our results in Section 6, before concluding.

2 Related work

Transferring annotation from one language to another in order to train monolingual tools for new languages was first introduced by Yarowsky and Ngai (2001). In their approach, token-level part-of-speech (PoS) and noun phrase bracketing information was projected across word-aligned bitext and this partial annotation served to estimate the parameters of a model that generalized from the noisy projection in a robust way. In more recent work, Das and Petrov (2011) propose a graph-based framework for projecting syntactic information across languages. They create type-level tag dictionaries by aggregating over projected token-level information extracted from bi-text and use label propagation on a similarity graph to smooth and expand the label distributions. A different approach to cross-lingual PoS tagging is proposed

[1]Most unsupervised approaches consider argument identification as a separate task that is omitted (Lang and Lapata, 2010) or performed heuristically (Lang and Lapata, 2011). We focus on semantic role labelling in this paper and consider argument identification as given.

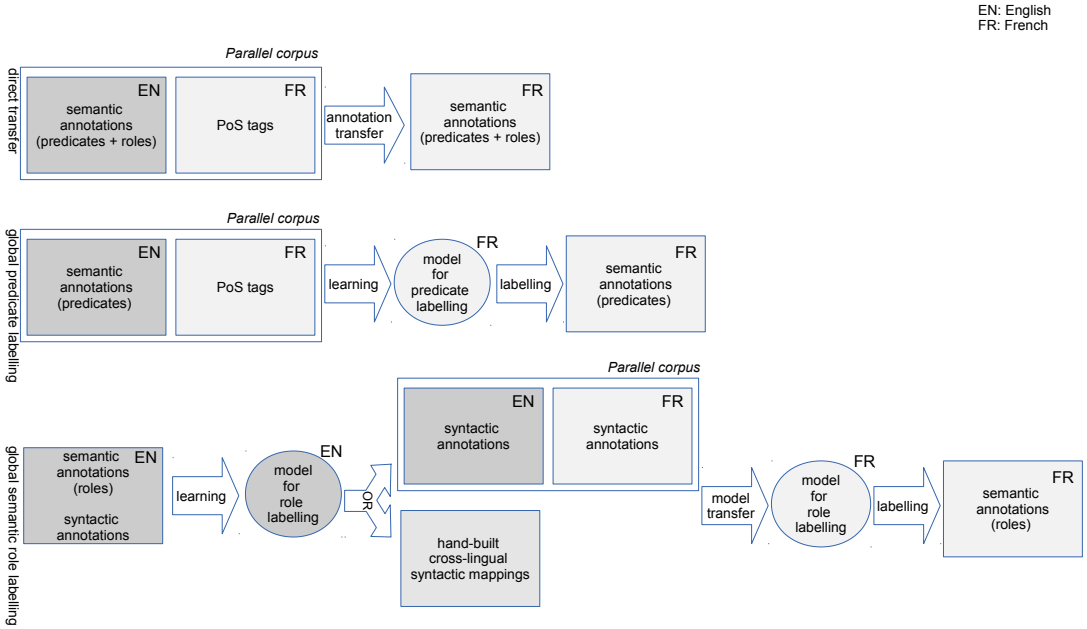


Figure 1: Direct vs. global cross-lingual transfer of semantic annotations.

by Täckström et al. (2013) who couple token and type constraints to guide learning. Our approach to cross-lingual semantic role labelling follows this vein. Instead of solely relying on token-level information acquired from word-alignments, we combine this with type-level information captured by our global methods which are trained on the entire corpus. We however are concerned with semantic annotations and not PoS.

Transfer of semantic annotation has started off with direct transfer of FrameNet semantic annotations (Padó, 2007; Basili et al., 2009; Annesi and Basili, 2010). With the addition of a learning step and the use of PropBank data, Van der Plas et al. (2011) have scaled up previous efforts. They show that a joint semantic-syntactic parser trained on the output of direct transfer produces better parses than the input it received by aggregating information across multiple examples. In their work, transfer of predicate labels and semantic role labels is done in one step. The model needs an aggressive filter to compensate for missing annotations on the predicate level after direct transfer. This filter successively leads to drops in performance for the role labellings. Here, we build two separate global models that complement direct transfer instead of relying on it.

The same emphasis on learning is found in cross-lingual model transfer where source language models are adapted to work on the target language directly. For semantic role labelling, Kozhevnikov and Titov (2013) use shared feature representations (syntactic and lexical) to adapt a source model to a target-language model. The ideas behind their cross-lingual model adaptation resemble the ideas behind our global method for semantic role labelling. However, in contrast to their work we do not consider the predicate labelling as given because, as manual annotations show (van der Plas et al., 2010), this task is not trivial. We first build a tailored global model for cross-lingual predicate labelling and then use the predicted predicate labels for semantic role labelling.

3 Data

In our experiments, we use the English-French part of the Europarl corpus (Koehn, 2005). The dataset is tokenised and lowercased and only sentence pairs corresponding to a one-to-one sentence alignment with lengths ranging from one to 40 tokens on both French and English sides are considered. Furthermore, because translation shifts are known to pose problems for the automatic projection of semantic roles across languages (Padó, 2007), we select only those parallel sentences in Europarl that are direct

translations from English to French or vice versa. In the end, we have a parallel corpus of 276-thousand sentence pairs.

The English part of the parallel corpus is annotated by a freely-available syntactic-semantic parser (Henderson et al., 2008; Titov et al., 2009) trained on the CoNLL 2009 training set (the Penn Treebank corpus (Marcus et al., 1993) merged with PropBank labels (Palmer et al., 2005) and NomBank labels (Meyers, 2007)). The probabilistic model is a joint generative model of syntactic and semantic dependencies that maximises the joint probability of the dependencies while building two separate structures.

The WSD classifier used for predicate labelling is trained on the parallel training corpus tagged with semantic roles on the English side. The candidate predicate labels that are considered by the classifier for each French verb are the labels of its English translations in the training corpus. Verbs on the English side are replaced by the corresponding predicate label where available. Then both parts of the corpus are lemmatized and tagged by part of speech (Schmid, 1994) and the parallel files are rebuilt (one sentence per line) by replacing words on both sides by the corresponding 'lemma_PoS tag' pair. The corpus is then word aligned in both directions using GIZA++ (Och and Ney, 2003) and a lexicon is built from intersecting alignments. Lexicon entries for French verbs contain the English predicate labels to which they were aligned in the training corpus. The entry for the verb *encourager*, for instance, contains seven predicate labels: {urge.01, foster.01, stimulate.01, promote.02, encourage.01, encourage.02, renew.01}, two of which correspond to the same English verb (*encourage*). We keep labels with an alignment confidence score above 0.01 according to GIZA++.

Contrary to our predicate labelling model, the role labelling model needs syntactic information in the target language. For parsing French, we use the dependency parser described in Titov and Henderson (2007). We train the parser on the dependency version of the French Paris 7 treebank (Candito et al., 2009), achieving 87.2% labelled accuracy on this data set. The French Treebank (Abeillé et al., 2003) is a treebank of 21,564 sentences annotated with constituency annotation. We use the automatic dependency conversion of the French Treebank into dependency format (Candito et al., 2009) to train the French syntactic parser that is used to annotate the French part of the parallel corpus.

For testing, we use the hand-annotated data described in Van der Plas et al. (2010). We randomly split those 1000 sentences into test and development set containing 500 sentences each. We use the development set for the current experiments, which contains 1,917 core roles in total. We limit our experiments to verbal predicates because the semantic annotations on French test sentences are limited to verbal predicates.

4 Direct cross-lingual transfer

Before explaining the global methods, we present the direct semantic transfer (DST) method proposed by Van der Plas et al. (2011) that we use for comparisons and combinations throughout this paper. The method is based on the Direct Correspondence Assumption for syntactic dependency trees proposed by Hwa et al. (2005). The transfer proceeds as follows: For any pair of sentences E and F that are translations of each other in the parallel corpus, we transfer the semantic relationship $R(x_E, y_E)$ to $R(x_F, y_F)$ if and only if there exists a word-alignment between x_E and x_F and between y_E and y_F, and we transfer the semantic property $P(x_E)$ to $P(x_F)$ if and only if there exists a word-alignment between x_E and x_F.

The relationships that are transferred are semantic role dependencies and the properties are predicate senses. These are transferred from the English part of the parallel training corpus that is automatically annotated with syntactic-semantic analyses, as explained in the previous section.

5 Global cross-lingual transfer of semantic annotations

In contrast to direct transfer where annotations are transferred on a token-to-token basis in word-aligned sentences, we propose two global methods for cross-lingual transfer, one for predicates and one for semantic roles, that both consist of a learning and a labelling step. Our methods are globally defined and as a consequence rely less on local translation correspondences than previous methods, which makes them less vulnerable to missing and incorrect alignment links.

5.1 Global cross-lingual predicate labelling

In cross-lingual predicate labelling, our aim is to put predicate labels that originate from the English side of the parallel corpus on the French verbs in the other side of the corpus. The predicate labels contain the English verb and its sense. For example, "give.01" stands for the first sense of the verb *give*. As the predicate label contains a lot of lexical information, putting the correct English predicate label on a French verb is very close to Word Sense Disambiguation (WSD), the task of automatically identifying the meaning of words in context (Navigli, 2009). In the cross-lingual variant of this task, the candidate senses are the words' translations in other languages and WSD aims at predicting semantically correct translations for words in context (Resnik and Yarowsky, 2000; Ng et al., 2003; Carpuat and Wu, 2007; Apidianaki, 2009). The main difference between cross-lingual WSD and our cross-lingual transfer of predicate labels is that we do not search for correct translations of French words but for the most appropriate predicate labels in context (i.e. verbs disambiguated with a predicate sense).

The global predicate labelling method consists of a learning step and a labelling step. During learning, we compute estimates for annotation transfer on the basis of the word alignments between English and French predicates over the entire parallel training corpus. At labelling time, we label French verbs with English predicate labels without the need for parallel data or alignments. The method is language-independent and only requires minimal linguistic resources (PoS information).

In terms of coverage, a predicate label is provided for all French verbs in the test set for which information was retained during training and not only for aligned ones, in contrast to direct transfer. We expect to augment the recall when using global estimates and hope that the effect on precision is not too negative.

Learning

For each French verb (v) in the lexicon built as described in Section 3, we want to be able to identify its correct predicate label in a new context by choosing one among its candidate labels (L) retained from the training corpus. A feature vector is built for each candidate label L_i ($1 \leq i \leq |L|$) found for the verb v in the lexicon, following the procedure described in Apidianaki et al. (2012). For each candidate label, we extract the content word co-occurrences of the verb v in the French sentences where it translates an English verb tagged with this label in the training corpus. The retained French words constitute the features of the vector built for that label. Let N be the number of features retained for each label L_i of the verb v from the corresponding French contexts. Each feature F_j ($1 \leq j \leq N$) receives a total weight with the label ($\text{tw}(F_j, L_i)$) which is learned from the data and defined as the product of the feature's global weight ($\text{gw}(F_j)$) and its local weight with that label ($\text{lw}(F_j, L_i)$). The global weight of a feature F_j is a function of the number n of candidate labels of v to which F_j is related, and of the probabilities (p_{ij}) that F_j co-occurs with instances of the verb v corresponding to each of the labels:

$$\text{gw}(F_j) = 1 - \frac{\sum_{i=1}^{n} p_{ij} \log(p_{ij})}{n} \tag{1}$$

Each p_{ij} is computed as the ratio of the co-occurrence counts of F_j with v when it is aligned to a label L_i to the total number of features (N) seen with this candidate label:

$$p_{ij} = \frac{cooc\_count(F_j, L_i)}{N} \tag{2}$$

The local weight between feature F_j and label L_i ($\text{lw}(F_j, L_i)$) directly depends on the number of times they occur together:

$$\text{lw}(F_j, L_i) = \log(\text{cooc\_count}(F_j, L_i)) \tag{3}$$

The intuition underlying this weighting scheme is that if an interesting semantic relation exists between a feature F_j and a specific predicate label L_i of a verb v, then we expect the probability (p_{ij}) of the feature F_j occurring in the contexts where v is translated by this label to be larger than if they were independent. In other words, a feature gets a high total weight (tw) with a label when it appears frequently in the corresponding French contexts and rarely in the contexts of the other labels.

Labelling

Predicate identification on the French side is done by selecting verbs based on the PoS labels provided by the tagger and subsequently filtering out modals and instances of the verb *être* (*be*).[2] The most suitable predicate labels are then assigned to the retained French verbs by the disambiguation classifier. The context of a new instance of a French verb is compared to the weighted feature vectors (V_i's) built for its candidate labels as described above, and an association score is assigned to each label. To facilitate comparison with the vectors, the new contexts (sentences) are lemmatised and PoS tagged on the fly (with TreeTagger) and the content word co-occurrences of the French verb are gathered in a bag of words. If common features (CFs) are found between the new context and the vector of a label (L_i), their association score corresponds to the mean of the weights of their shared features with L_i found in the corresponding vector. In Equation 4, $(CF_j)_{j=1}^{|CF|}$ is the set of common features between a label vector V_i and the new context C and tw is the total weight of a CF with label L_i, computed as explained in the previous section.

$$\text{assoc\_score}(V_i, C) = \frac{\sum_{j=1}^{|CF|} \text{tw}(CF_j, L_i)}{|CF|} \quad (4)$$

The label that receives the highest association score with the new context is returned and serves to annotate the corresponding French verb.

5.2 Global cross-lingual role labelling

For role labelling, we adopt a different strategy. Whereas predicate labels include a lot of lexical information, role labels do not. However, for role labels there is another source of information that helps to define global estimates: the correlation between syntax and semantics.

Previous work in monolingual unsupervised semantic role induction (Grenager and Manning, 2006; Lang and Lapata, 2010; Lang and Lapata, 2011) showed that mapping rules that assign semantic roles to arguments of a verb based on the syntactic functions of these arguments, represent a baseline that is very hard to beat. This strong correlation between syntactic labels and semantic role labels in the PropBank annotation has been shown in detail by Merlo and Van der Plas (2009). In contrast to previous work on monolingual unsupervised semantic role induction, we add the predicate label as a predictor. The core arguments of the verb, that are the numbered labels in PropBank, are known to be verb-specific. We have access to predicate labels assigned by the cross-lingual predicate labelling method described in the previous section and exploit them for role labelling.

For a given predicate, diathesis alternations are the major source of variation in propositions. They give rise to different syntactic structures, while the semantic roles remain stable. For example, the sentence "I gave the book to Jean" is syntactically different from "I gave Jean the book", but semantic roles on the three arguments stay the same. We will show in a feasibility study that the effect of diathesis alternations on the correlation between syntax and semantics is limited. In a cross-lingual setting, structural divergences (Dorr, 1994) are expected to reduce the correlation between syntax and semantics. An example is the difference in syntactic structure between the sentences "Tu me manques" vs. "I miss you", which are translations of each other, however the semantic roles are the same across languages.

As our global method is not restricted to alignments at labelling time, we are able to classify all given arguments[3] and not just those that are aligned in a parallel corpus. In this way, we believe that the negative effect of structural divergences and diathesis alternations is limited. Moreover, we show how mild supervision from the partial annotations that result from the direct transfer can potentially remedy these difficulties.

Learning syntactic-semantic mappings

The syntactic-semantic mapping rules that are exploited by our model for role labelling are extracted from gold-annotated monolingual data. As a consequence, the extracted rules are of high quality which

[2] We exclude the verb *être* because its English counterpart (*be*) is not annotated in the CoNLL-2009 data used in our experiments.

[3] We focus on the classification of core semantic roles because diathesis alternations and cross-lingual divergences mainly involve these roles.

would not be the case if parallel data was used. Manually annotated parallel corpora are very sparse and automatic parsing introduces errors which might be propagated by the direct transfer methods and result in noisy annotations. Using gold-standard monolingual data thus ensures the quality of the mappings exploited by our global model.

We build a model that determines the most suitable semantic role label r for a given argument of a given predicate p, based on its syntactic dependency label d.[4] We simply compute the maximum likelihood estimates (MLE) and count occurrences of the following triples $< p, d, r >$ in a large body of English gold semantically and syntactically annotated data.

$$P_{MLE}(r|p,d) = \frac{count(p,d,r)}{count(p,d)} \tag{5}$$

In the cross-lingual setting, the mapping rules extracted from the English training data are applied to French. We learn the correspondences between English and French syntactic labels in a data-driven way by syntactically annotating both sides of our parallel training corpus. We base the cross-lingual syntactic mapping on alignment counts between syntactic labels in the parallel corpus parsed syntactically both on the English and the French side (cf. Section 3). An alternative that needs no parallel data is to study the annotation guidelines for the two languages and determine the cross-lingual correspondences between syntactic labels by hand.

The syntactic label set used for French (Candito et al., 2009) is less fine-grained than the English labels (20 versus 36). As a consequence, the mapping from English syntactic labels to French treebank labels is for the most part a many-to-one mapping, which leads to information loss but suffices for our purpose as will be shown in the next section.

Once the correspondences between the syntactic labels of the two syntactic annotation frameworks are discovered, the cross-lingual transfer of syntactic-semantic mappings consists in substituting the English syntactic labels with their French counterparts to adapt the model described above.

Labelling

For role labelling, we use estimates derived from the training data (see Equation 5) to determine the most suitable role of a given argument. Because a particular triple in the test set might not have been seen during training, we backoff to 2-tuples that discard the predicate label, and backoff to A1 if neither the dependency label nor the predicate has been seen in training.

To treat the R-suffix, which takes care of anaphoric arguments, we use the following simple rule: for the monolingual setting all arguments with PoS-tags "WDT", "WP", and "WRB" receive the R-suffix. In the cross-lingual setting, we translate the PoS tags to the single French PoS tag "PROREL". We do not treat the C-prefix, which takes care of discontinuous arguments, because there were only a few examples.

We do not accept duplicate semantic roles, a constraint that leads to valid role configurations in general (Punyakanok et al., 2008). We expect the more prominent semantic roles, such as A0 and A1, to appear earlier in the sentence than semantic roles with higher numbers. We therefore attribute semantic roles of a predicate from left to right.

5.3 Combining direct and global cross-lingual transfer

Direct transfer methods generally have low recall, we however expect them to be more precise than the global methods. In our combined method, we use the annotations assigned by direct transfer as the backbone and fill missing labels by the global methods. The annotations from direct transfer restrict the possible roles the global method adds. We expect, as an additional benefit of this combination, that the partial annotations from direct transfer together with the no-duplicate-role constraint described above will remedy problems related to diathesis alternations. Although the probabilities computed will favour the canonical alternation in general, the partial annotations may prevent a canonical analysis in a particular proposition. Consider the following alternation example: *Mary presented the flowers to John* vs. the less canonical alternation *Mary presented John with the flowers*. Although the most probable role

[4] We chose not to include the complete dependency path from predicate to argument because of data sparseness. We select the dependency label on the arc that points to the argument under discussion.

Predicate identification and labelling							
		Labelled			Unlabelled		
		Prec	Rec	F	Prec	Rec	F
1	Direct	51	29	37	93	57	71
2	Global	45	39	42	95	83	89
3	Combined	45	45	**45**	92	91	**91**
4	Plas11	68	25	37	98	36	53
5	Plas11(f)	56	46	**51**	97	80	**87**
6	Manual	61	57	59	97	89	93

Table 1: Percent recall, precision and F-measure for predicate identification and labelling.

Cross-lingual semantic role labelling		
1	Direct	35
2	Global	68
3	Combined	73
4	Most frequent semantic role	48

Table 2: Percent accuracy for semantic role labelling

for the *prep* relation would be A2, based on the canonical alternation, partial annotations on *Mary* (A0) and *John* (A2) in combination with the no-duplicate-role constraint would rule that out and the next most probable label would be put on *with*: A1.

6 Results and discussion

We ran experiments using the two global methods described in Section 5 separately and combined with direct transfer. In this section, we present the results and compare to several baselines and upper bounds from manual annotations and previous work.

6.1 Cross-lingual predicate labelling

Table 1 shows the results of cross-lingual predicate labelling (Labelled) and identification (Unlabelled). The first row shows the results from using the traditional direct transfer method. The second row presents results from the global method where we use cross-lingual WSD to label predicates. The third row combines direct and global transfer, as explained in Section 5.3. For comparison, we present results when using the parser from Van der Plas et al. (2011) on our test data: the fourth row contains results when using all (unfiltered) data, the fifth row when using data filtered for incomplete predicate labellings. We show an upper bound in the last row which corresponds to the inter-annotator agreement for manual annotation on a random set of 100 sentences (van der Plas et al., 2010).

Overall the figures, including the upper bound from manual annotations, are not very high. Annotating French verbs with English predicate labels is a hard task. When we look at the differences between the three automatic methods, we see that recall is very low (29%) for the direct method. From the recall figures for unlabelled predicates, we see that the direct method leaves many predicates without a label.

The global method has a much better recall, 39%, and a slightly lower precision. The best results are however attained when the two methods are combined, that is, when global transfer is used to fill in missing predicates from direct transfer. We get an F-measure of 45% which is a big improvement over the baseline of direct transfer, which attained 37%. These results show that the global method for predicate labelling improves recall without sacrificing precision too much.

We compare these results also to the results obtained by Van der Plas et al. (2011)'s three step model, where a parser trained on transferred annotations annotates in turn the test sentences. We see that the current method gives better results (recall and F-measure) when the parser is trained on unfiltered data. An aggressive filter, that removes more than half of the data and leads to a big drop in performance for argument labelling (recall that argument and predicate labelling is done in parallel in this model) finally leads to a result that outperforms ours. This result is not surprising because the parser has access to much more expressive syntax. Note that our global method only needs a PoS tagger in the source language and no syntactic information nor joint semantic-syntactic parsing frameworks. It is thus knowledge-lean and easier to apply to languages without a parser, a difference that should be taken into account in the interpretation of the results. However, we can learn from these results that structural information is beneficial. In future work, we plan to include word position information in our cross-lingual WSD method. This will give the method access to structural information while keeping it knowledge-lean.

In Figure 2, we give an example that illustrates the contribution of the global method. In this example,

English (automatic): There is in particular one amendment, let [let.01] me point [point.02] out, concerning [concern.01] the energy sector, which, in my capacity as rapporteur, I see [see.01] as particularly important.
Transfer: Il y a notamment un amendement, je le souligne, concernant [concern.01] le secteur de l'énergie, qui me paraît en tant que rapporteur particulièrement important.
CLWSD: Il y a notamment un amendement, je le souligne [stress.01], concernant [concern.01] le secteur de l'énergie, qui me paraît [seem.01] en tant que rapporteur particulièrement important.

Figure 2: Predicate label addition and correction using CLWSD.

the cross-lingual WSD method annotates more verbs than the direct transfer approach: labels [stress.01] and [seem.01] assigned during disambiguation, are missing from the first sentence after transfer. Even with a high quality word alignment, it would not be possible to get these labels from the English source sentence through direct transfer because they are simply not there, due to the non-literal translation. This example shows the limitations of token-to token direct transfer and how the global method compensates for that by using information aggregated across the whole parallel corpus.

6.2 Global cross-lingual role labelling

Though already supported by previous work (Grenager and Manning, 2006; Lang and Lapata, 2010; Lang and Lapata, 2011), we tested the hypothesis that syntactic-semantic mappings provide good approximations for semantic role labelling, especially when adding predicate information. We therefore first ran a monolingual feasibility study by collecting counts from the CoNLL 2009 training set and testing on the CoNLL 2009 test set. The accuracy attained with this simple method is 79%. This shows that in a monolingual setting, the predicate label combined with the syntactic label of the argument are good predictors for the semantic role of the argument. This number can serve as a baseline for semantic role labelling given the correct predicate label.

In previous sections, we discussed diathesis alternations as problematic for using syntactic-semantic mapping rules. To measure the importance of diathesis alternations we need to measure the variation in a large corpus. By applying the mapping rules learned from the training data on the same data we get an idea of the amount of variation. We get an accuracy of 86%. Although the 14% probably contains the most interesting examples from a linguistic point of view, these results on monolingual data show that predicate-centered syntactic-semantic mapping rules are a promising direction for improving recall in direct transfer methods.

Table 2 shows the results[5] for semantic role labelling for the three cross-lingual transfer methods and the baseline of applying the most frequent semantic role label 'A1'. For the global and the combined methods we use the predicate labels provided by the cross-lingual WSD method. The numbers in Table 2 provide performance numbers given the predicate from the cross-lingual WSD method. We discussed in Subsection 5.2 that, when applying syntactic-semantic mapping rules cross-lingually, differences in annotation framework and cross-lingual divergences are at play. We see indeed that when applying syntactic-semantic mapping rules cross-lingually the accuracy drops from 79 to 68%. This drop in performance when applying to French the syntactic-semantic mappings that were learned on English data is not too important. This accuracy number is, in any case, much better than the results from direct transfer. This is mainly due to the low recall of direct transfer which results in very few but rather precise (87%) semantic roles. It is therefore very useful to use the direct transfer method as a backbone that restricts the labels we get from global transfer by imposing consistency with the available annotation (no-duplicate-argument-constraint). By combining the two methods, we get 73% accuracy that is not far from the 79% in the monolingual setting. In future work, we would like to investigate whether the drop in performance between the monolingual and cross-lingual setting is larger for languages that are less related.

We also compare our results to previous work on cross-lingual transfer of semantic roles. Kozhevnikov and Titov (2013) evaluate on the full test set described in Subsection 3 (1000 sentences), they use gold predicates instead of predicted predicates and evaluate on both core roles and adjuncts. The authors shared with us their results for core roles only: 74% and 77%, when using original and transferred

[5] As we focus on argument labelling (and not identification) we provide accuracy scores.

syntax, respectively. We use original syntax and should therefore compare to the 74%. When we use gold predicate annotations as Kozhevnikov and Titov (2013) did, instead of the predicate labels obtained through cross-lingual WSD, and test on all 1000 sentences, we attain 75% for the combined method and 71% for the global method. These results compare favourably with their results. This is encouraging because their model uses a larger feature set that includes (cross-lingual) lexical features, the unlabelled dependency graph and PoS information. Interestingly, they attain better scores when they use a transferred syntactic model instead of the original syntax. This result seems in line with our discussion on the loss of information when trying to map the English syntactic label inventory to the French inventory. We keep syntactic model transfer in mind for future work.

Because we consider the arguments as given, while Van der Plas et al. (2011) do both argument identification and labelling for all core roles and adjuncts, and provide precision and recall given the predicate only, we cannot directly compare to their results. We however include their results for the sake of completeness. Their parser results in 65% F-score.

Applying A1 (the most frequent semantic role) to the entire data set gives us 48% accuracy. That is much higher than results from transfer, again due to the low recall of the direct transfer method, but much lower than the results of the combined and global methods.

7 Conclusion

We have introduced a global approach to transfer that aggregates information at the corpus level thereby correcting and complementing the annotations from traditional direct transfer methods that suffer from token-level mistakes. We show that the combination of direct transfer (a high-precision method) and global methods (high in recall) outperforms previous results.

In contrast to previous work, we transfer predicate annotations and semantic role annotations by building two separate models tailored to the task at hand. We show how the predicate labels produced by our high-coverage model for cross-lingual predicate labelling are successfully used as predictors for semantic role labelling.

In future work, we would like to feed structural information to the cross-lingual WSD method such as information about word position, which would preserve its knowledge-lean character without need for syntactic parsing. Furthermore, we intend to use cross-lingual WSD for labelling adjuncts (non-core semantic roles) since this task is also rather lexical in nature. Last but not least, we want to add argument identification which will allow to propose a complete SRL annotation framework based on global information.

Acknowledgements

This research was funded and supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) as part of the SFB 732.

References

A. Abeillé, L. Clément, and F. Toussenel. 2003. Building a treebank for French. In *Treebanks: Building and Using Parsed Corpora*. Kluwer Academic Publishers.

P. Annesi and R. Basili. 2010. Cross-lingual alignment of FrameNet annotations through Hidden Markov Models. In *Proceedings of CICLing*.

M. Apidianaki, G. Wisniewski, A. Sokolov, A. Max, and F. Yvon. 2012. WSD for n-best reranking and local language modeling in SMT. In *Proceedings of the Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 1–9, Jeju, Republic of Korea, July. Association for Computational Linguistics.

M. Apidianaki. 2009. Data-driven Semantic Analysis for Multilingual WSD and Lexical Selection in Translation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL-09)*, pages 77–85, Athens, Greece.

R. Basili, D. De Cao, D. Croce, B. Coppola, and A. Moschitti, 2009. *Computational Linguistics and Intelligent Text Processing*, chapter Cross-Language Frame Semantics Transfer in Bilingual Corpora, pages 332–345. Springer Berlin / Heidelberg.

M.-H. Candito, B. Crabbé, P. Denis, and F. Guérin. 2009. Analyse syntaxique du français : des constituants aux dépendances. In *Proceedings of la Conférence sur le Traitement Automatique des Langues Naturelles (TALN'09)*, Senlis, France.

M. Carpuat and D. Wu. 2007. Improving Statistical Machine Translation using Word Sense Disambiguation. In *Proceedings of the Joint EMNLP-CoNLL Conference*, pages 61–72, Prague, Czech Republic.

D. Das and S. Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 600–609, Portland, Oregon, USA, June. Association for Computational Linguistics.

B. Dorr. 1994. Machine translation divergences: A formal description and proposed solution. *Computational Linguistics*, 20(4):597–633.

T. Grenager and C. Manning. 2006. Unsupervised discovery of a statistical verb lexicon. In *Proceedings of EMNLP*.

J. Hajič, M. Ciaramita, R. Johansson, D. Kawahara, M. A. Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štepánek, P. Straňák, M. Surdeanu, N. Xue, and Y. Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009)*.

J. Henderson, P. Merlo, G. Musillo, and I. Titov. 2008. A latent variable model of synchronous parsing for syntactic and semantic dependencies. In *Proceedings of CONLL 2008*, pages 178–182.

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

P. Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of MT Summit X*, pages 79–86, Phuket, Thailand.

M. Kozhevnikov and I. Titov. 2013. Crosslingual transfer of semantic role models. In *In Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, August. Association for Computational Linguistics.

J. Lang and M. Lapata. 2010. Unsupervised induction of semantic roles. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947, Los Angeles, California, June. Association for Computational Linguistics.

J. Lang and M. Lapata. 2011. Unsupervised semantic role induction via split-merge clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*.

M. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Comp. Ling.*, 19:313–330.

P. Merlo and L. van der Plas. 2009. Abstraction and generalisation in semantic role labels: PropBank, VerbNet or both? In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 288–296, Suntec, Singapore.

A. Meyers. 2007. Annotation guidelines for NomBank - noun argument structure for PropBank. Technical report, New York University.

R. Navigli. 2009. Word Sense Disambiguation: a Survey. *ACM Computing Surveys*, 41(2):1–69.

H. T. Ng, B. Wang, and Y. S. Chan. 2003. Exploiting Parallel Texts for Word Sense Disambiguation: An Empirical Study. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 455–462, Sapporo, Japan.

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

S. Padó. 2007. *Cross-lingual Annotation Projection Models for Role-Semantic Information*. Ph.D. thesis, Saarland University.

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31:71–105.

V. Punyakanok, D. Roth, and W. Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

P. Resnik and D. Yarowsky. 2000. Distinguishing Systems and Distinguishing Senses: New Evaluation Methods for Word Sense Disambiguation. *Natural Language Engineering*, 5(3):113–133.

H. Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK, September. http://www.ims.uni-stuttgart.de/~schmid/.

O. Täckström, D. Das, S. Petrov, R. McDonald, and J. Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. In *Transactions of the ACL*. Association for Computational Linguistics, March.

I. Titov and J. Henderson. 2007. A latent variable model for generative dependency parsing. In *Proceedings of the International Conference on Parsing Technologies (IWPT-07)*, pages 144–155, Prague, Czech Republic.

I. Titov, J. Henderson, P. Merlo, and G. Musillo. 2009. Online graph planarisation for synchronous parsing of semantic and syntactic dependencies. In *Proceedings of the twenty-first international joint conference on artificial intelligence (IJCAI-09)*, Pasadena, California, July.

L. van der Plas, T. Samardžić, and P. Merlo. 2010. Cross-lingual validity of PropBank in the manual annotation of French. In *In Proceedings of the 4th Linguistic Annotation Workshop (The LAW IV)*, Uppsala, Sweden.

L. van der Plas, P. Merlo, and J. Henderson. 2011. Scaling up cross-lingual semantic annotation transfer. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics and the Human Language Technologies conference*.

D. Yarowsky and G. Ngai. 2001. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Multilingual Semantic Parsing : Parsing Multiple Languages into Semantic Representations

Zhanming Jie
University of Electronic Science & Technology of China
allanmcgrady@gmail.com

Wei Lu
Information Systems Technology and Design
Singapore University of Technology and Design
luwei@sutd.edu.sg

Abstract

We consider *multilingual semantic parsing* – the task of simultaneously parsing semantically equivalent sentences from multiple different languages into their corresponding formal semantic representations. Our model is built on top of the *hybrid tree* semantic parsing framework, where natural language sentences and their corresponding semantics are assumed to be generated jointly from an underlying generative process. We first introduce a variant of the joint generative process, which essentially gives us a new semantic parsing model within the framework. Based on the different models that can be developed within the framework, we then investigate several approaches for performing the multilingual semantic parsing task. We present our evaluations on a standard dataset annotated with sentences in multiple languages coming from different language families.

1 Introduction

Semantic parsing, the task of parsing natural language sentences into their formal semantic representations (Mooney, 2007) is one of the most important tasks in the field of natural language processing and artificial intelligence. This area of research recently has received a significant amount of attention (Zettlemoyer and Collins, 2005; Kate and Mooney, 2006; Wong and Mooney, 2006; Lu et al., 2008; Jones et al., 2012b). Consider these example sentence-semantics pairs:

English:	Which states have points that are higher than the highest point in Texas ?
Semantics:	$answer(state(loc_1(place(higher_2(highest(place(loc_2(stateid('TX'))))))))$
English:	What rivers do not run through Tennessee ?
Semantics:	$answer(exclude(river(all), traverse_2(stateid('TN'))))$

In the typical setting, the semantic parser learns from a collection of such sentence-semantics pairs a model that can parse novel input sentences into their respective semantic representations. Such semantic representations can then be used to interact with certain downstream components to perform interesting tasks. For example, retrieving of answers from an underlying database, or performing certain actions based on the generated executable semantic instructions.

Note that in the training data, although complete sentence-semantics pairs are given, specific word-level semantic information is not explicitly provided. The model therefore needs to automatically learn such latent mappings between natural language words/phrases and semantic units.

One natural assumption is that the semantics exhibit certain restricted structures, such as the recursive tree structures. Under such an assumption, one can convert the second semantics appeared above as the tree structure illustrated in Figure 1. More details about such tree structured representations will be given in Section 2.1.

Currently, researchers only focused on the semantic parsing task under a single language setting where the input is a sentence from one particular language. However, natural language is highly ambiguous, and identifying the correct semantics associated with words with limited background information is a

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1291–1301, Dublin, Ireland, August 23-29 2014.

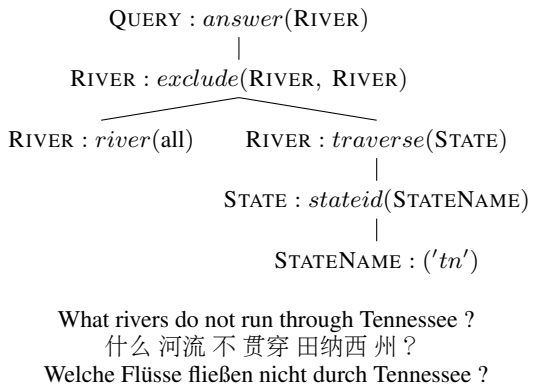


Figure 1: An example tree-structured semantic representation (above) and its corresponding natural language sentences (in English, Chinese and German).

challenging task. Researchers resorted to performing context-dependent semantic parsing to alleviate such an issue (Zettlemoyer and Collins, 2009).

On the other hand, researchers have successfully exploited parallel texts for improved word-level semantic processing (Chan and Ng, 2005). This is because words from different languages that convey the same semantics can be used to disambiguate each other's semantics. In fact, texts from different languages that convey the same semantic information becomes increasingly available nowadays. Web crawlers such as Google and Yahoo! are able to rapidly aggregate a large volume of news stories every day. One crucial fact is that many such news articles written in different languages are actually all discussing the same underlying story and therefore convey similar or identical semantic information. To build better automatic systems for improved natural language understanding, it is therefore helpful to develop algorithms that can simultaneously process the underlying semantic information associated with all these documents coming from different language sources together. For example, consider the following example taken from the multilingual version of the dataset, which shows semantically equivalent sentences from three different languages and their corresponding semantics:

English:	What rivers do not run through Tennessee ?
Chinese:	什么 河流 不 贯穿 田纳西？
German:	Welche Flüsse fließen nicht durch Tennessee ?
Semantics:	$answer(exclude(river(all), traverse_2(stateid('TN'))))$

As a step towards the above-mentioned goal, this work focuses on the development of an automated system that is capable of simultaneously parsing semantically equivalent natural language texts in different languages into their underlying semantics.

Specifically, in this work, we first introduce a new variant of a semantic parsing model under an existing framework. This new variant can be used together with other models for jointly making semantic parsing predictions, leading to an improved multilingual semantic parsing system. We demonstrate the effectiveness of this new variant through experiments. Although bilingual parsing has been extensively studied in fields such as statistical machine translation (Wu, 1997; Chiang, 2007), to the best of our knowledge, bilingual or multilingual semantic parsing that focuses on parsing sentences from multiple different languages into their formal semantic representations has not yet been studied. We present the very first work on performing multilingual semantic parsing that simultaneously parses semantically equivalent sentences from multiple different languages into their semantics. We believe this line of work can potentially lead to further developments and advancements in areas such as multilingual semantic processing and semantics-based machine translations (Jones et al., 2012a).

2 Background

2.1 Semantics

Researchers have focused on various semantic formalisms for semantic parsing. Popular examples include the tree-structured semantic representations (Wong and Mooney, 2006; Kate and Mooney,

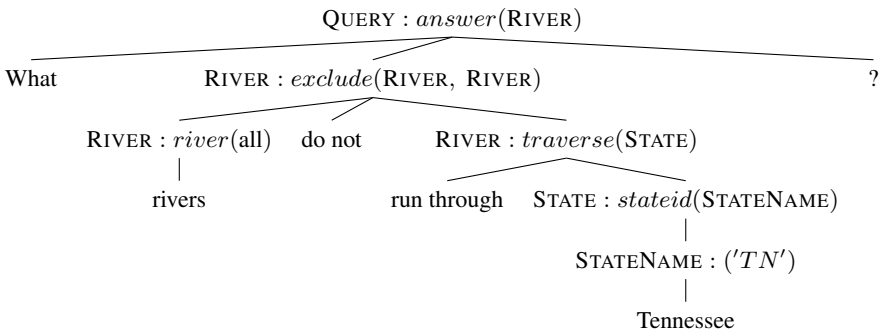


Figure 2: An example hybrid tree. Such a hybrid tree is generated from the generative process, and captures the correspondences between natural language words and semantic units.

2006), the lambda calculus expressions (Zettlemoyer and Collins, 2005; Wong and Mooney, 2007), and dependency-based semantic representations (DCS) (Liang et al., 2013). In this work, we specifically focus on the tree-structured representations for semantics.

Each semantic representation consists of semantic units as its tree nodes, where each semantic unit is of the following form:

$$m_a \equiv \tau_a : p_\alpha(\tau_b*) \tag{1}$$

Here m_a is used to denote a complete semantic unit, which consists of its semantic type τ_a, its function symbol p_α, as well as a list of types for argument semantic units τ_b* (here $*$ means 0, 1, or 2; we assume there are at most two arguments for each semantic unit). In other words, each semantic unit can be regarded as a function which takes in other semantic representations of specific types as arguments, and returns a new semantic representation of a particular type. For example, in Figure 1, the semantic unit at the root has a type QUERY, a function name $answer$, and a single argument type RIVER.

2.2 Related Work

Substantial research efforts have focused on building monolingual semantic parsing systems. We survey in this section several of them.

WASP (Wong and Mooney, 2006) is a model motivated by statistical synchronous parsing-based machine translation (Chiang, 2007), which essentially casts the semantic parsing problem as a phrase-based translation problem (Koehn et al., 2003). KRISP (Kate and Mooney, 2006) makes use of Support Vector Machines with string kernels (Lodhi et al., 2002) to recursively map contiguous word sequences into semantic units to construct a tree structure. The SCISSOR model (Ge and Mooney, 2005) performs integrated semantic and syntactic parsing. The model parses natural language sentences into semantically augmented parse trees whose nodes consist of both semantic and syntactic labels and then builds semantic representations based on such augmented trees. The *hybrid tree* model (Lu et al., 2008; Lu et al., 2009), whose code is publicly available, makes the assumption that there exists an underlying generative process for jointly producing both the language and semantics. The model employs efficient dynamic programming algorithms for learning a distribution over the latent *hybrid trees* which jointly encode both language and semantics. An example hybrid tree representation is shown in Figure 2. Jones et al. (2012b) recently proposed a framework that performs semantic parsing with tree transducers. The model learns representations that are similar to the hybrid tree structures using a generative process under a Bayesian framework.

Besides these approaches, recently there are also several works that take alternative learning approaches for semantic parsing which do not require annotated semantic representations (Poon and Domingos, 2009; Clarke et al., 2010; Goldwasser et al., 2011; Liang et al., 2013; Artzi and Zettlemoyer, 2013). Most of such approaches rely on either weak supervision or certain forms of indirect supervision. Some of these works also focus on optimizing specific downstream tasks rather than the semantic parsing task itself.

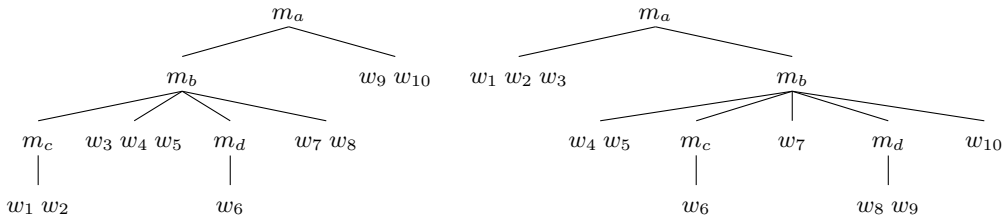


Figure 3: Two example hybrid trees. Their leaves are natural language words, and the internal nodes are semantic units. Both hybrid trees correspond to the same $\mathbf{n}$-$\mathbf{m}$ pair $\langle \overline{w_1\ w_2\ w_3\ w_4\ w_5\ w_6\ w_7\ w_8\ w_9\ w_{10}}, m_a(m_b(m_c, m_d))\rangle$. Thus they can be viewed as two different ways of generating such a pair from the joint generative process.

We note there also exist various multilingual or cross-lingual semantic processing works. Most of such works focus on semantic role labeling(SRL), the task of recovery of shallow meaning. Examples include multilingual semantic role labeling (Björkelund et al., 2009), multilingual joint syntactic and semantic dependency parsing (Henderson et al., 2013), and cross-lingual transfer of semantic role labeling models (Kozhevnikov and Titov, 2013). Researchers also looked into exploiting semantic information for bilingual processing such as machine translations (Chan et al., 2007; Carpuat and Wu, 2007; Jones et al., 2012a).

In this work, we focus on the task of multilingual semantic parsing under the setting where the input consists of semantically equivalent sentences from multiple different languages, and the outputs are formal semantic representations. We specifically focus on the hybrid tree model, a state-of-the-art framework for semantic parsing. We first make an extension to the model, and investigate methods for performing such a multilingual semantic parsing task by aggregating a few variants of the models under such a framework.

3 Approach

In this section, we first discuss the hybrid tree model of Lu et al. (2008), and introduce a novel extension. Next we discuss the approach used for multilingual semantic parsing.

3.1 The Hybrid Tree Model

For a given $\mathbf{n}$-$\mathbf{m}$ pair (where $\mathbf{n}$ is a complete natural language sentence, and $\mathbf{m}$ is a complete semantic representation), the hybrid tree model assumes that both $\mathbf{n}$ and $\mathbf{m}$ are generated from an underlying generative process in a top-down, left-to-right, level-by-level, recursive manner. The joint generative process for the pair results in a new tree-structured representation called a *hybrid tree*, which consists of natural language words as leaves, and semantic units as internal nodes.

There are three types of model parameters involved in the generative process. The meaning representation model parameters (ρ) are used for generating one semantic unit from its parent semantic unit. The hybrid pattern parameters (ϕ) are used for deciding how natural language words and semantic units are organized together to form the next level of the nodes of the hybrid tree structure. The emission parameters (θ) are used for generating natural language words from its corresponding semantic unit.

For a given $\mathbf{n}$-$\mathbf{m}$ pair, there are multiple possible hybrid trees that can jointly represent such a pair. See Figure 3 for two possible hybrid trees that contain the same $\mathbf{n}$-$\mathbf{m}$ pair. Consider the first example hybrid tree illustrated there. The probability of generating such a hybrid tree $\mathbf{h}$ (i.e., jointly generating both the natural language sentence $\mathbf{n}$ and the semantics $\mathbf{m}$) is:

$$\begin{aligned} P(\mathbf{n}, \mathbf{m}, \mathbf{h}) = {} & \rho(m_a) \times \phi(\overline{\mathbf{X}\mathbf{w}}|m_a) \times \theta(\mathbf{X}|m_a, \Lambda) \times \theta(w_9|m_a, \Lambda) \times \theta(w_{10}|m_a, \Lambda) \\ & \times \rho(m_b|m_a, \arg = 1) \times \phi(\overline{\mathbf{X}\mathbf{w}\mathbf{Y}\mathbf{w}}|m_b) \times \theta(\mathbf{X}|m_b, \Lambda) \times \theta(w_3|m_b, \Lambda) \\ & \times \theta(w_4|m_b, \Lambda) \times \theta(w_5|m_b, \Lambda) \times \theta(\mathbf{Y}|m_b, \Lambda) \times \theta(w_7|m_b, \Lambda) \times \theta(w_8|m_b, \Lambda) \\ & \times \rho(m_c|m_b, \arg = 1) \times \phi(\overline{\mathbf{w}}|m_c) \times \theta(w_1|m_c, \Lambda) \times \theta(w_2|m_c, \Lambda) \\ & \times \rho(m_d|m_b, \arg = 2) \times \phi(\overline{\mathbf{w}}|m_d) \times \theta(w_6|m_d, \Lambda) \end{aligned} \quad (2)$$

Note that $\overline{\mathbf{Xw}}$ refers to a pattern which says the next level of the hybrid tree is expected to consist of the first child semantic unit, followed by a contiguous sequence of natural language words. Similar definitions can be given to the patterns $\overline{\mathbf{XwYw}}$ and $\overline{\mathbf{w}}$, where $\mathbf{X}$ and $\mathbf{Y}$ refer to the first and second child semantic unit, respectively. The symbols $\mathbf{X}$ and $\mathbf{Y}$ appear in emission parameters are used to denote placeholders for the first and second child semantic unit, respectively.

The hybrid tree model then focuses on the learning of these model parameters from the training data using maximum likelihood estimation. In other words, the model tries to maximize:

$$\sum_i \log P(\mathbf{n}_i, \mathbf{m}_i; \rho, \phi, \theta) = \sum_i \log \sum_{\mathbf{h}} P(\mathbf{n}_i, \mathbf{m}_i, \mathbf{h}; \rho, \phi, \theta) \tag{3}$$

Since the correct hybrid tree associated with $\mathbf{n}$-$\mathbf{m}$ pair is unknown, we marginalize over the hidden variable $\mathbf{h}$. The model parameters will then be estimated using the Expectation-Maximization (EM) algorithm (Dempster et al., 1977). Specifically, an inside-outside style algorithm (Baker, 2005) is used where an additional layer of dynamic programming algorithms are used for efficient inference (Lu et al., 2008). The complexity of the inference algorithm is $O(mn^3)$, where m is the size of the semantic representation (number of semantic units), and n is the number of words in the input sentence.

Note that the generation of natural language words involves the context Λ. Specifically, if the context is empty, the model is regarded as the *unigram model*. If the context is the previously generated word, the model is called a *bigram model*. For example, consider the generation of the natural language word w_4 in the left hybrid tree in Figure 2. The probability for generating this word is $\theta(w_4|m_b)$ and $\theta(w_4|m_b, w_3)$, under the unigram and the bigram model, respectively. In Lu et al. (2008), the mixgram model (an interpolation between the unigram model and the bigram model) was also considered when parsing novel sentences, which yielded a better performance.

Once the model parameters are learned, we will be able to use them to parse novel sentences. Specifically, for each novel input sentence, we first find the most probable hybrid tree that contains the sentence $\mathbf{n}$, and then extract its internal nodes to form the semantic representation. Efficient dynamic programming algorithms similar to the ones used for training can also be employed here. In addition, the algorithm can also be extended to support exact top-k decoding, which will be useful later for combining multiple lists of outputs with rank aggregation (to be discussed in Sec. 3.3).

3.2 The Backward Bigram Model

One assumption associated with the original hybrid tree model is that nodes at each level of the hybrid tree are generated from the left to the right. An alternative assumption would be that the nodes at each level are generated in the reverse order – from the right to the left. While this alternative assumption will not introduce any difference in the unigram model (since each node is generated from its respective parent semantic unit only, regardless of its context), such a new assumption will lead to a completely new generative process under the bigram assumption.

To see this, again consider the emission probability for generating the word w_4 in the hybrid tree on the left of Figure 3. Under the assumption of our new model, the probability of generating this word is $\theta(w_4|m_b, w_5)$, since now the context Λ becomes the word to the right of the current word. The parameter estimation and parsing (decoding) procedures are largely similar to those of the original bigram model, where similar efficient dynamic programming algorithms can be employed.

3.3 Multilingual Semantic Parsing

In multilingual semantic parsing, the input consists of multiple semantically equivalent sentences, each of which is from a different language. One approach for building such a multilingual semantic parsing system is to develop a joint generative process from which both the semantic representations and the sentences in different languages are generated simultaneously. However, building such a joint model is non-trivial. Typically, sentences from different languages exhibit very different syntactic structures and word orderings. It is also non-trivial to design efficient dynamic programming algorithms for this case where multiple languages are involved in the joint generative process. Furthermore, the difficulty of building such a joint generative model becomes higher as the number of input languages increases.

Previous research efforts show that it can be beneficial to learn individual models independently, and then combine the learned models only during the inference stage (Punyakanok et al., 2005; Chang et al., 2012). Motivated by this, we take the approach that learns a separate semantic parser for each different language first. Next, we combine these semantic parsers for different languages into a single multilingual semantic parser only during the inference stage.

One common approach for combining different outputs from different systems is to perform *majority voting* based on optimal predictions from each parser. We first obtain the best output semantic representation from each individual semantic parser, and then count the number of occurrences for each possible output. The most frequent output semantic representation is returned as the final output of our system. Naturally, this approach is only applicable when there are at least three systems/models.

An alternative approach is to allow each system to produce a ranked list of k most probable outputs, each is associated with a score. Our system then aggregates these ranked lists to select the best output. This problem is known as *rank aggregation* and has been extensively studied in fields such as data mining and information retrieval (Dwork et al., 2001; Gleich and Lim, 2011; Li, 2011). For our task, we first let each semantic parser (for each language) generate a ranked list of the top-k most probable outputs (hybrid trees) for the given input. Next, based these hybrid trees we find a ranked list of most probable semantic representations. Each such semantic representation is also associated with a score, which is the log-likelihood of the hybrid tree, i.e., $\log P(\mathbf{n}, \mathbf{m}, \mathbf{h})$. Note that for each semantic representation, we only consider the score associated with the most probable hybrid tree that contains such a semantic representation. We use the standard approach for combining two ranked lists with scores. Consider a ranked list from the j-th model/system that consists of n distinct items. Let's use $s_i^{(j)}$ to denote the original score associated with the i-th semantic representation in the j-th ranked list. We normalize the score $s_i^{(j)}$ in the following way to obtain the new score $\tilde{s}_i^{(j)}$ (normalized score, divided by the standard deviation associated with the sample):

$$\tilde{s}_i^{(j)} = \frac{s_i^{(j)}}{n\mu^{(j)}\delta^{(j)}} \quad \text{where} \quad \mu^{(j)} = \frac{1}{n}\sum_{k=1}^{n} s_k^{(j)}, \quad \delta^{(j)} = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(s_k^{(j)} - \mu^{(j)})^2}$$

Such new scores will then be used for aggregating the results to form a new ranked list. How do we find the best output from multiple lists? Two useful sources of information that we may use include: 1) the number of times each output appears in these lists; 2) the combined score $\sum_j \tilde{s}^{(j)}$ for each output s. We believe the more frequent an output appears in these lists (i.e., more systems/models predict such an output in their top-k lists), the more likely it can be a good candidate. Therefore we first find the set of most frequent outputs, next from such a set we select the output with the highest overall score $\sum_j \tilde{s}^{(j)}$ as the final output of our system.

4 Experiments

4.1 Data and Setup

We conducted our experiments on the multilingual GEOQUERY dataset released by Jones et al. (2012b). This dataset consists of 880 instances of natural language queries related to US geography facts. Each query is coupled with its corresponding semantic representation originally written in Prolog. The original GEOQUERY dataset (Wong and Mooney, 2006; Kate and Mooney, 2006) contains natural language queries in English only. Additional Chinese annotations were provided by Lu and Ng (2011) when performing a natural language generation task. Jones et al. (2012b) further provided the following three additional language annotations to this dataset: German, Greek and Thai. Thus, this dataset is now fully annotated with five different languages, two of which (Chinese, Thai) are Sino-Tibetan languages, and the rest are all Indo-European languages.

Following previous works on semantic parsing (Kwiatkowski et al., 2010; Jones et al., 2012b), we split the dataset into two portions. The training set consists of 600 instances, and we report evaluation results on the portion consisting of the remaining 280 instances. We used the identical split provided by Jones et al. (2012b) for all the experiments. Following previous works, we used the standard approach for

	EN	DE	EL	TH	CN
Unigram	70.0	59.6	70.0	68.9	68.9
Bigram	75.4	56.1	65.4	70.7	68.9
Bigram (inv)	74.3	57.1	65.4	71.1	66.8
Mixgram	76.1	**62.5**	69.3	73.2	70.7
Voting (u,b,m)	76.1	61.1	70.4	73.6	70.0
Voting (u,b,bi)	76.4	61.4	71.8	**74.3**	72.1
Aggregation	**78.6**	60.0	**72.1**	71.4	**73.2**

Table 1: Monolingual semantic parsing results on all five languages (EN:English, DE:German, EL:Greek, TH:Thai, CN:Chinese.). We report accuracy percentages in this table.

	ENDE	ENEL	ENTH	ENCN	DEEL	DETH	DECN	ELTH	ELCN	THCN
Unigram	74.6	76.1	76.4	75.0	76.8	72.1	74.3	80.4	79.6	74.0
Bigram	80.0	77.9	**87.1**	78.2	72.1	75.0	76.4	81.4	76.8	79.6
Bigram (inv)	78.2	76.8	86.4	75.7	72.5	75.7	76.1	82.1	75.7	79.3
Mixgram	77.9	76.4	82.5	81.1	76.1	75.7	74.3	81.1	80.7	77.9
Voting (u,b)	80.0	79.6	83.6	82.1	**77.1**	74.6	74.6	82.1	78.6	79.6
Voting (u,b,bi)	**82.1**	79.3	86.4	82.1	76.8	**77.1**	76.4	**85.4**	78.9	**80.7**
Aggregation	78.9	**82.1**	85.7	**83.6**	76.4	73.6	**76.8**	83.9	**81.4**	79.3

Table 2: Semantic parsing results when two different input languages are considered (for example, the column ENDE gives the results when each input to our system consists of a pair of semantically equivalent sentences written in English and German.). Scores are accuracy percentages.

evaluation on the multilingual GEOQUERY dataset. Specifically, we first let our semantic parsers produce semantic representations from multilingual input sentences. The resulting semantic representations are then converted into Prolog queries in a deterministic manner, which can be used to interact with the underlying knowledge base to retrieve answers. A predicted semantic representation is considered correct if and only if it retrieves identical results as the correct reference semantic representation when both are used for retrieving answers from the underlying database.

4.2 Results and Discussions

We performed experiments on the conventional monolingual semantic parsing task first. We report accuracy scores, which are defined as the number of correctly parsed inputs (i.e., the total number of correct semantic representations) divided by the total number of input sentences. Baseline results for unigram, bigram, and mixgram models, which are originally introduced in Lu et al. (2008) are reported under "Unigram", "Bigram", and "Mixgram" respectively in Table 1. The results for backward bigram models are reported under "Bigram(inv)".

To assess the effectiveness of our methods for combining different outputs, we first conducted experiments on voting over the outputs from the three models originally introduced in the work of Lu et al. (2008) (Voting(u,b,m)). Next we performed voting over outputs from unigram model, bigram model, as well as the backward bigram model introduced in this paper (Voting(u,b,bi)). These voting-based approaches yielded better results over the first voting-based approach. Specifically, we compared this new voting-based approach against the previous best model reported in Lu et al. (2008) – mixgram model, which was also based on a combination of unigram and bigram models. We used the paired t-test to assess the significance of the overall improvements across different languages when using our new method. When comparing the approach "Voting(u,b,m)" over "Mixgram", we obtained a one-tailed p-value of 0.40. When comparing the approach "Voting(u,b,bi)" over "Mixgram", we obtained a one-tailed p-value of 0.11. We also investigated the effectiveness of the aggregation-based approach. This approach is based on aggregating the two top-100 lists generated by unigram, bigram and backward bigram models. When comparing this approach over "Mixgram", we obtained a one-tailed p-value of

	EnDe El	EnDe Th	EnDe Cn	EnEl Th	EnEl Cn	EnTh Cn	DeEl Th	DeEl Cn	DeTh Cn	ElTh Cn
Unigram	79.6	78.2	79.3	83.2	83.2	79.3	81.8	79.6	77.1	81.4
Bigram	82.1	85.7	81.8	87.5	81.8	86.4	82.5	80.7	79.6	83.6
Bigram (inv)	82.9	85.4	79.6	86.8	81.1	85.4	82.1	80.4	78.9	83.2
Mixgram	81.4	83.2	81.8	85.0	83.2	84.3	82.9	80.7	79.3	82.9
Voting (u,b)	83.2	85.0	84.3	87.9	84.0	85.0	84.0	**83.6**	**81.1**	84.6
Voting (u,b,bi)	**84.0**	**86.1**	**85.4**	**89.6**	84.3	**86.8**	**85.0**	82.5	**81.1**	84.6
Aggregation	83.6	85.0	**85.4**	88.9	**87.1**	85.7	82.5	82.5	80.0	**85.4**

Table 3: Semantic parsing results when three different input languages are considered (for example, the column EnDeEl gives the results when each input to our system consists of three semantically equivalent sentences, which are written in English, German and Greek, respectively.). Scores are accuracy percentages.

	EnDe ElTh	EnDe ElCn	EnDe ThCn	EnEl ThCn	DeEl ThCn	EnDeEl ThCn
Unigram	82.9	82.1	81.1	85.0	82.1	84.0
Bigram	86.1	83.6	84.3	87.1	85.0	86.1
Bigram (inv)	86.4	82.5	84.0	86.8	85.4	85.0
Mixgram	84.0	82.1	83.2	86.4	84.0	85.7
Voting (u,b)	87.5	86.1	86.4	89.6	**86.4**	89.3
Voting (u,b,bi)	**88.6**	86.8	**87.1**	**90.0**	85.7	**89.6**
Aggregation	87.1	**87.1**	86.1	88.9	86.1	88.6

Table 4: Semantic parsing results when four or five different input languages are considered (for example, the column EnDeElTh gives the results when each input to our system consists of four semantically equivalent sentences, which are written in English, German, Greek, and Thai respectively.). Scores are accuracy percentages.

0.29 under the paired t-test. These results indicate that the approach based on voting over the unigram, bigram and backward bigram models gives the most promising results for monolingual semantic parsing, demonstrating the usefulness of our proposed backward bigram model.

Next we move to the multilingual setting where we would like to simultaneously process more than two languages. Specifically, we considered multilingual semantic parsing where there are two, three, four and five input languages. Table 2, Table 3, and Table 4 summarize these results. Table 2 shows the results for bilingual semantic parsing where we have two different input languages. The results reported under "Unigram" are based on the aggregation approach over unigram models. Similarly for "Bigram", "Bigram(inv)", and "Mixgram" (we also tried the voting-based approach for combining such baseline systems, which yielded slightly worse results). From this table we can see that generally speaking by considering two different languages as the input, our system is able to do better semantic parsing. We compared the voting-based approaches against the baseline approaches. For the approach "Voting(u,b)" (we excluded mixgram models in voting since now we have four models, two from each language, which are sufficient for voting, and preliminary results show that the inclusion of the mixgram models is not helpful), it does not outperform the bigram baseline approach (which is the most competitive amongst all baseline approaches) significantly ($p = 0.19$). When comparing the aggregation approach against the bigram baseline approach, we obtain $p = 0.04$. In contrast, the approach "Voting(u,b,bi)" outperforms all the baseline systems significantly ($p < 0.005$). These results again demonstrate the effectiveness of our newly proposed backward bigram model.

We can see from the results presented in Table 3 and Table 4 that, in general, the performance of the multilingual semantic parser tends to improve as the number of input languages increases. However this is not always the case. For example, consider the final system where we use all five languages as the input (refer to the results in the column of EnDeElThCn in Table 4); interestingly, when we remove German (De) from the inputs, we are able to build a better system in terms of accuracy (refer to the results in the column of EnElThCn). We believe this is partly due to the fact that the monolingual semantic parsing

task with German as the input language (see DE in Table 1) is relatively more challenging. Nevertheless, when all the languages are considered, the overall system is able to obtain an accuracy of 89.6% with the voting-based approach where our proposed backward bigram model is incorporated. This is significantly higher than any other monolingual system's performance reported in the literature. According to Jones et al. (2012b), the results of state-of-the-art monolingual semantic parsing systems on four of these five languages considered here are: 82.1%(EN), 75.0%(DE), 75.4%(EL), and 78.2%(TH). Note that to date, no single system reported in the literature can dominate all other systems across all these languages on this dataset in terms of accuracy performance. We hypothesize that this is because semantic information conveyed by the sentences from a single language tends to be highly ambiguous, and various linguistic phenomenons can be difficult to capture under a monolingual setting for any existing monolingual semantic parsing system. The multilingual semantic parsing system introduced in this work, in contrast, can exploit richer information from multiple languages to successfully disambiguate the semantics associated with the inputs for improved semantic parsing.

5 Conclusions

In this work, we focused on *multilingual semantic parsing*, the task of simultaneously parsing sentences from various different languages into their corresponding formal semantic representations. Our work is built on top of the *hybrid tree* framework where different generative process can be developed for jointly modelling the generation of both language and semantics. We first introduced a variant of the generative process, leading to a new semantic parsing model. Next we presented methods for combining and aggregating outputs from different models within the framework to build our multilingual semantic parsing system. Our results demonstrate the effectiveness of our approaches for such a task. To the best of our knowledge, this is the first work that tackles such a multilingual semantic parsing task which simultaneously parses sentences from multiple languages into formal semantic representations. Future work include explorations on applications of our system in areas such as multilingual semantic processing, cross-lingual semantic processing, and semantics-based machine translations (Jones et al., 2012a).

Acknowledgements

We would like to thank the anonymous reviewers for comments. This work was conducted during the first author's internship at SUTD. This work was supported by SUTD grant SRG ISTD 2013 064.

References

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *TACL*.

James K Baker. 2005. Trainable grammars for speech recognition. *The Journal of the Acoustical Society of America*, 65(S1):S132–S132.

Anders Björkelund, Love Hafdell, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of CONLL'09*, pages 43–48.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of EMNLP-CoNLL '07*, pages 61–72.

Yee Seng Chan and Hwee Tou Ng. 2005. Scaling up word sense disambiguation via parallel texts. In *Proceedings of AAAI '05*, pages 1037–1042.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of ACL '07*.

Ming-Wei Chang, Lev Ratinov, and Dan Roth. 2012. Structured learning with constrained conditional models. *Machine learning*, 88(3):399–431.

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

James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world's response. In *Proceedings of CONLL '10*, pages 18–27.

Arthur P Dempster, Nan M Laird, Donald B Rubin, et al. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal statistical Society*, 39(1):1–38.

Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. 2001. Rank aggregation methods for the web. In *Proceedings of WWW '01*, pages 613–622.

Ruifang Ge and Raymond J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of CONLL '05*, pages 9–16.

David F Gleich and Lek-Heng Lim. 2011. Rank aggregation via nuclear norm minimization. In *Proceedings of KDD '11*, pages 60–68.

Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *Proceedings of ACL '11*, pages 1486–1495.

James Henderson, Paola Merlo, Ivan Titov, and Gabriele Musillo. 2013. Multi-lingual joint parsing of syntactic and semantic dependencies with a latent variable model. *Computational Linguistics*, 39(4).

Bevan Jones, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, and Kevin Knight. 2012a. Semantics-based machine translation with hyperedge replacement grammars. In *Proceedings of COLING '12*, pages 1359–1376.

Bevan Keeley Jones, Mark Johnson, and Sharon Goldwater. 2012b. Semantic parsing with bayesian tree transducers. In *Proceedings of ACL '12*, pages 488–496.

Rohit J. Kate and Raymond J. Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proceedings of COLING/ACL '06*, pages 913–920.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL '03*, pages 48–54.

Mikhail Kozhevnikov and Ivan Titov. 2013. Cross-lingual transfer of semantic role labeling models. In *Proceedings of ACL '13*.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of EMNLP'10*, pages 1223–1233.

Hang Li. 2011. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 4(1):1–113.

Percy Liang, Michael I Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *The Journal of Machine Learning Research*, 2:419–444.

Wei Lu and Hwee Tou Ng. 2011. A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. In *Proceedings of EMNLP '11*, pages 1611–1622.

Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *Proceedings of EMNLP '08*, pages 783–792.

Wei Lu, Hwee Tou Ng, and Wee Sun Lee. 2009. Natural language generation with tree conditional random fields. In *Proceedings of EMNLP '09*, pages 400–409.

Raymond J. Mooney. 2007. Learning for semantic parsing. In *Computational Linguistics and Intelligent Text Processing*, pages 311–324. Springer.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of EMNLP '09*, pages 1–10.

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. 2005. Learning and inference over constrained output. In *Proceedings of IJCAI '05*, pages 1124–1129.

Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of HLT/NAACL '06*, pages 439–446.

Yuk Wah Wong and Raymond J. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of ACL '07*.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Luke S Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of UAI '05*.

Luke S Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *Proceedings of ACL/IJCNLP '09*, pages 976–984.

Unsupervised Word Sense Induction using Distributional Statistics

Kartik Goyal
Carnegie Mellon Uniersity
kartikgo@cs.cmu.edu

Eduard Hovy
Carnegie Mellon University
hovy@cmu.edu

Abstract

Word sense induction is an unsupervised task to find and characterize different senses of polysemous words. This work investigates two unsupervised approaches that focus on using distributional word statistics to cluster the contextual information of the target words using two different algorithms involving latent dirichlet allocation and spectral clustering. Using a large corpus for achieving this task, we quantitatively analyze our clusters on the Semeval-2010 dataset and also perform a qualitative analysis of our induced senses. Our results indicate that our methods successfully characterized the senses of the target words and were also able to find unconventional senses for those words.

1 Introduction

Word Sense Induction (WSI) involves automatically determining the number of senses of a given word or a phrase and identifying the features which differentiate those senses. This task, although similar to the Word Sense Disambiguation (WSD) task, is fundamentally different because it does not involve any supervision or explicit human knowledge about senses of words. WSI has potential to be extremely useful in downstream applications because, apart from the savings on annotation costs, it also mitigates several theoretical conflicts associated with supervised WSD tasks, which generally involve deciding on the granularity of senses. Ideally, a WSI algorithm would be able to adapt to different tasks requiring different sense granularities. WSI algorithms can also be used to model the evolution of the senses of a word with time and hence can be much easier to maintain than existing fixed sense inventories like WordNet(Miller, 1995), Ontonotes(Hovy et al., 2006) etc. Automatic sense identification systems also have the potential to generalize well to large amounts of diverse data and hence be useful in various difficult domain independent tasks such as machine translation and information retrieval.

Several factors make the problem of word sense induction very challenging. Most importantly, it is not clear what should be the 'true' senses of a word. The semantic continuum makes it always possible to break a sense into finer grained subsenses. Thus, the problem is one of finding the optimal granularity for any given task. Even in a semi-supervised setting, it is unknown which sense inventories are most suited as starting points in a sense bootstrapping procedure.

Our unsupervised approach relies heavily on the distributional statistics of words which occur in the proximity of the target words. Hence, we first obtain the distributional statistics from a very large corpus to facilitate generalization and reliable estimation of different possible senses. Then we use these statistics in a novel manner to obtain a representation for the senses of the target word. In this paper, we discus the performance of induced senses on the Semeval 2010 WSD/WSI(Manandhar et al., 2010) task.

2 Related Work

Much of the work on word sense induction has been quite recent following the Semeval tasks on WSI in 2007(Agirre and Soroa, 2007) and 2010, but the task was recognized much earlier and various semi-supervised and unsupervised efforts were directed towards the problem.Yarowsky (1995) proposed a

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1302–1310, Dublin, Ireland, August 23-29 2014.

semi-supervised approach, which required humans to specify seed words for every ambiguous word and assumed one sense per discourse for an ambiguous word. The unsupervised approaches mainly focus on clustering the instances of the target words in a corpus, using first-order vectors, second-order vectors (Purandare and Pedersen, 2004)(Schütze, 1998) etc. Pantel and Lin (2002) used various syntactic and surface features for clustering the various occurences of a target word. Co-occurence graph-based approaches(Véronis, 2004) have also been used, which represent the words co-occuring with the target words as nodes and then identify the highly dense subgraphs or 'hubs' within this co-occurence graph. Brody and Lapata (2009) and Lau et al. (2012) proposed bayesian WSI systems which cluster the instances by applying Latent Dirichlet Allocation (LDA)(Blei et al., 2003), Hierarchical Dirichlet Processes (HDP)(Teh et al., 2006) etc. wherein each occurence of a target word is represented as a 'document' and its surrounding context as the 'observable content'. Choe and Charniak (2013) propose a 'naive bayes' model for WSI which assumes one sense per discourse and uses Expectation Maximization(EM) to estimate model parameters like the probability of generating an instance feature like a word in the context, given the sense of the target word in a particular instance. Reisinger and Mooney (2010) and Huang et al. (2012) have proposed sense dependent multiple prototypes for a word instead of the conventional one vector representation per word and have shown that this sense differentiation improves semantic similarity measurements between words.

3 Basic Motivation: Co-occurence graphs

Conventionally, each word is represented as a co-occurence vector which may contain frequency, point wise mutual information or some lower dimensional representation of context and this representation conflates all the senses of a word. These vectors can be viewed as a graph where words are nodes which have an edge between them if a word occurs in the distributional vector of another. Given a target ambiguous word w, we refer to those words as the 'first order' words(referred to by 'neighbors') which are directly connected to w. The 'second order' words are the words directly connected to the first order words and so on. This graph is cyclic and each node might have multiple senses conflated into it. In this work, we only consider the first and second order words, eg. a target word like 'bank' will have words like 'river' ,'money' etc in it's first order and the second order vectors will be the words from the context of the first order words like 'river':'flood','plains' etc, 'money': 'currency', 'economy' etc. Essentially, these second order words characterize the first order words and hence are very informative for clustering the first order words into different senses. Essentially, we use the second order words as features of the first order words and use them to cluster the first order words into different senses.It must be noted that the first order words themselves might have multiple senses and ideally, those words should also be disambiguated but in the current work we only focus on disambiguating the 'target' words.

4 Methodology

For clustering the neighbors of the target words, we implement and compare two methods which differ significantly in their technical details and employ distribtutional statistics of the neighbors differently, which we describe in the sections below. For obtaining the distributional statistics on a large scale, we used the 5-gram data of Google N-gram corpus(Michel et al., 2011) which effectively lets us use as 10 word window. No lemmatization or case normalization was performed because the large corpus size ameliorated the problem of sparseness. Only nouns, verbs, adjectives and adverbs were employed for the statistical estimation because our pilot studies suggested that these words were most informative.

4.1 Latent Dirichlet Allocation

LDA(Blei et al., 2003) is a well known bayesian generative topic model which models every 'document' as a mixture of latent 'topics' and all its 'words' as multinomial draws from those latent topics. In topic model parlance, a 'corpus' consists of various 'documents'. Each 'document' has a collection of tokens which is treated like a bag of words, where each word is drawn from a latent 'topic'. The topics are shared across documents thus giving each document a topic proportion based upon the topic assignment of the tokens in a document. The priors on topic proportions and the topic multimonial paramers are

dirichlet parameters. An important characteristic of LDA is its clustering property which makes the model inclined to enforce sparseness with small dirichlet priors.
It is important to note that we employ LDA in a significantly different manner than the previous approaches which have used LDA or other related topic models for word sense induction. Other topic modelling based approaches for WSI represent each instance of the target word as a 'document' and the immediate context as the 'bag of words' for that 'document'. Unlike these approaches, we represented a target ambiguous word as the 'corpus' in the topic modelling parlance. Then we found out all the 'first order' words co-occuring with the target word within a 10 word window. Each 'first order' word/type is considered a 'document' in our LDA based approach. The latent 'topics' for each 'document' are the latent 'senses' and each first order type comprises of a 'sense distribution' which is indicative of its tendency to induce a particular sense in the target word. The 'second order' types are all the words occuring in a 10 word window of every 'first order' word. These types along with their frequency, form the 'bag of words' for the 'first order' type(LDA document). Hence, in our model, the latent senses are shared across all the first order neighbors of the target word and the second order tokens play the role of 'words' in our LDA based model. After getting the sense distributions for each first order type, we perform k-means over all the sense distribution vectors such that every first order neighbor gets assigned a cluster.
We posit that the distributional statistics of a large corpus helps in improving the coverage of second or-

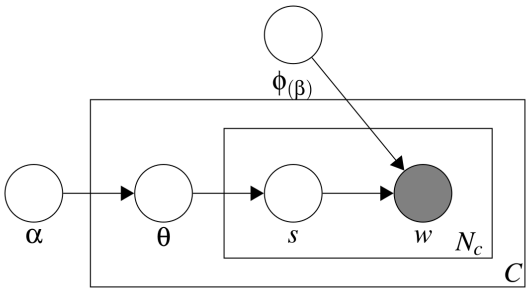


Figure 1: Figure1: s is the latent sense variable. θ is the sense distribution of a first order neighbor. w is a second order neighbor of a first order word. ϕ is the sense multinomial with a dirichlet prior β. α is a dirichlet prior on the sense proportion of a first order type.

der words which are essential for reliable clustering of the first order words. However, the large number of occurences and a large vocabulary make it intractable to run LDA using the original frequency of the second order words. To overcome this computational hurdle, we posit that with a diverse representation of the second order words, LDA based parameter estimation relies more upon the relative distribution of the these words across all the first order words rather than their actual distributions. Hence, we decided to scale down the actual counts for each word so that we could run LDA with the finite resources available. An important parameter in this model is the number of latent topics/senses to use, which is specified to be the actual number of senses specified in the Ontonotes sense inventory. This is an idealized case in which the number of senses are known. The α hyperparameter is chosen to be small with respect to the average 'document lengths' we encounter. This has the effect of pushing most of the probabilistic weight to one topics instead of diluting it among many topics. We also decided to analyze the effect of part of speech tags of the second order words in clustering the first order words. The various configurations we experimented with were:

- All: Considered nouns,verbs,adjectives and adverbs in second order bag of words.
- Nouns: Only considered second order words which were nouns to study the effect of Nouns on clustering.
- Verbs: Only considered second order words which were verbs.

- Nadj: Considered both nouns and adjectives to study the effect of Noun phrases over clustering.

- Vadv: Considered both verbs and adjectives for second order bag of words.

4.2 Spectral Clustering

Spectral Clustering(Ng et al., 2002) is a clustering technique which uses a pairwise similarity matrix, L, to find out clusters such that the seperation between the entities in two seperate clusters is maximum while implicitly taking into account the distances between groups of points instead of considering them individually. The aim is to find the eigenvectors of $D^{-1}L$ corresponding to smallest eigenvalues to minimize the similarity across two clusters. Here D is a diagonal matrix with degree of node i on entry D_{ii}. For k clusters, k eigenvectors ordered by their eigenvalues are found out. These k eigenvectors are used to form a $n \times k$ matrix where n is the number of datapoints. Each row of this matrix is considered a datapoint with a vector of length k, thus effectively reducing the dimension of the datapoints to k most prominent dimensions according to the similarity matrix decomposition. Finally, k-means is performed on the n vectors to assign a cluster to each datapoint.
We cluster the first order neighbors for each target word using spectral clustering. The crux of this algorithm lies in using appropriate pairwise distance matrices. For constructing the pairwise distance matrices of first order types, we used two vectorial representations of the first order words:

- **Senna embeddings**: The word embeddings trained by a neural network by (Weston et al., 2012)

- **Distributional vectors** comprised of the frequencies of the second order words.

Then we used these vectors to calculate mutual pairwise distance matrices(we experimented with Euclidean and Cosine distances), which were converted into similarity matrices by using Gaussian kernels. These matrices were used as input to the spectral clustering algorithm.
We chose to ignore very low frequency words for making word vectors. This cutoff was decided by analyzing the distributional frequency vs. rank curves of the words, which were heavy tailed. Again, we use the same number of clusters as the number of senses in Ontonotes sense inventory, so that we can study the correspondence between our clusters and the Ontonotes senses.

5 Quantitative Analysis

In this paper, we discus our systems' performances on the Semeval-2010 word sense induction/disambiguation dataset, which contains 100 target words: 50 nouns and 50 verbs. The test data is a part of OntoNotes (Hovy et al., 2006) and contains around 9000 instances of usage of the target words. For annotating a particular test instance, we first filtered the surrounding context to retain only salient Nouns, Verbs, Adverbs, and Adjectives. We report a mixture of senses for each instance, where the weight for each sense was proportional to the number of filtered surrounding words belonging to that sense/cluster. As mentioned earlier, we experimented with a variety of settings for spectral clustering and LDA based methods. The performance with different settings was generally similar and hence, we report our best results here. For a better insight into how our models in different settings performed, we also report the full tables for paired F-score. The performance trend of various systems is similar for other measures. We compare our results to three baselines:

- Most Frequent Sense (MFS) baseline: assigns all the test instances to the most frequent sense of the target word.

- Brown University's system results (Choe and Charniak, 2013).

- Lau (LDA) (Lau et al., 2012), who provide only the results for one of the three measures. In particular, we compare our system to their results obtained by a model that was based on LDA and used the gold standard number of senses as the number of topics to be used.

System	V-measure			Paired F-score			Supervised F-score			#cl
	all	nouns	verbs	all	nouns	verbs	all	nouns	verbs	
LDA	4.4	5.2	3.2	60.7	53.2	71.7	60.9	55.2	69.2	2.45
Spectral	4.5	4.6	4.2	61.5	54.5	71.6	60.7	55.1	68.8	1.87
MFS	0.0	0.0	0.0	63.5	57.0	72.7	58.7	53.2	66.6	1.00
Brown	18.0	23.7	9.9	52.9	52.5	53.5	65.4	62.6	69.5	3.42
Lau	-	-	-	-	-	-	64.0	60.0	69.0	-

Table 1: Performance on Paired F-score and supervised F-score. LDA and Spectral are the two methods proposed in this paper. Lau is the baseline in which LDA system of (Lau et al., 2012) is considered. It should be noted that in their paper, (Lau et al., 2012) did not report their performance on Paired F-score.

The Semeval-2010 task provides us with 3 evaluation metrics: V-measure, Paired F-score and Supervised F-score. It was noticed (Manandhar and Klapaftis, 2009) that V-measure tends to favour systems that produce a higher number of clusters than the gold standard and hence is not a reliable estimate of the performance of WSI systems. But, we report our results on V-measure too as it gives useful insight about the nature of data and the WSI algorithms.
It is important to note that all the measures treat Ontonotes sense annotations as the gold standard, which makes this task unfit for our evaluation purposes. As mentioned earlier, our argument is that several decisions related to the granularity of senses and definition of senses are a topic of dispute, and hence we believe that instead of relying upon a pre-annotated sense inventory, it should be more effective to induce senses automatically in an unsupervised manner using a large and unbiased corpus, and tune the granularity governing parameters for different downstream tasks which require sense disambiguation. But our performance on these annotations still provides us with valuable information about the agreement between Ontonotes senses and our systems' senses. In our experiments, we have not tried to tune the hyperparameters or perform agglomerative clustering to better fit our clusters to the gold standard clusters by using training/development set at all, because we wanted to analyze the performance of our algorithms in the most general setting.

5.1 V-Measure

The V-measure defines the quality of a cluster to be the harmonic mean of homogeneity and coverage. These can be viewed as precision and recall of the element-wise assignment to clusters, where homogeneity measures the 'pureness' of the clusters and coverage measures the 'cohesiveness'. It was noticed (Manandhar and Klapaftis, 2009) that V-measure tends to favour systems producing a higher number of clusters than the gold standard and hence is not a reliable estimate of the performance of WSI systems. In addition, the number of induced clusters in our systems is bounded at the top by the Gold Standard number of senses because of our choice of hyperparameters in both spectral clustering and LDA based approaches.

From the results, we realized that the number of senses induced in the test set by our system is quite low compared to the baselines and other systems that participated in Semeval-2010. This hurts our V-measure. Our systems perform better on nouns than verbs generally according to this measure. Also, LDA-based approaches with the number of topics equal to the number of gold-standard senses perform the best. For spectral clustering, euclidean distances seem to perform better.

5.2 Paired F-score

The paired F-score is the harmonic mean of precision and recall on the task of classifying whether the instances in a pair belong to the same cluster or not. This measure also penalizes the systems if the number of induced senses is not equal to the number of senses in the gold standard. It must be noted that in our approach, the induced number of senses on the test dataset is not equal to the original number of senses although we clustered with the number of clusters specified by Ontonotes, because our clusters are different from Ontonotes senses. MFS has a recall of 100% which makes it a very hard baseline to

P F-score(%)	all	nouns	verbs	#cl
CD20	60.5	53.1	71.3	2.12
CD15	57.9	50.8	68.2	2.26
CD10	58.5	50.7	69.7	2.27
ED20	61.5	54.5	71.6	1.87
ED15	60.6	53.1	71.5	2.12
ED10	60.0	52.3	71.3	2.45
CS15	59.6	52.9	69.4	2.25
CS10	60.1	51.9	72.0	2.07
ES15	59.8	52.9	71.3	2.15
ES10	60.8	53.5	71.4	2.21
MFS	63.5	57.0	72.7	1.00
Brown	52.9	52.5	53.5	3.42

Table 2: General trend for the various settings: Paired F-Score Evaluation: Spectral Clustering: 'C':cosine distance, 'E': Euclidean Distance, 'D': Second order Distributinal counts, 'S':Senna embeddings and the adjacent numbers are the number of nearest neighbors(in 1000s) considered for the distance matrix.

P F-score(%)	all	nouns	verbs	#cl
all	60.7	53.2	71.7	2.47
noun	59.6	52.1	70.7	2.32
verb	60.0	52.4	71.0	2.25
nadj	59.7	52.6	70.1	2.3
vadv	59.3	52.27	69.6	2.25
MFS	63.5	57.0	72.7	1.00
Brown	52.9	52.5	53.5	3.42

Table 3: General trend for the various settings: Paired F-Score Evaluation: LDA: 'all': All POS tags considered in the first order neighborhood, 'noun': Only nouns considere, 'verbs': Only verbs considered, 'nadj': nouns and adjectives considered, 'vadv':verbs and adverbs considered

beat. Semeval-2010 results show that none of the systems outperform the MFS baseline. Both of our systems perform better than other systems on this measure and are comparable to the performance of the MFS baseline.

5.3 Supervised F-score

For the supervised task, the test data is split into two parts: one for mapping the system senses to the gold standard senses, and the other for evaluation based upon the mapped senses. We report our performance on the 80% mapping and 20% evaluation split. The mapping is done automatically by the program provided by the organizers which is based upon representing the gold standard clusters as a mixture of the system senses.

Our different systems perform similarly on the supervised evaluation. We outperform the tough MFS baseline and perform competitively against other systems. We observe that other systems outperform us on the target nouns whereas our performance on verbs is similar to that of other systems. This can be attributed to the fact that our methods induce a small number of senses in general over the test set but according to the test data based upon Ontonotes, the senses of nouns have a much higher resolution than verbs.

5.4 Discussion on Quantitative Results

In general, we found our performance to be competetive with the other systems. Also, we perform significantly better than other Semeval-2010 systems on the paired F-score metric. In our experiments,

Sense	Cluster Words
1	Engineers,Presbyterian,Service,Jewish,Police,Ethnicity,Independent,Movements
2	membrane,complicated,surgical,hypothalamic,potassium,lymphatic,electron,tumor
3	Cynthia,Armstrong,Tracy,Marilyn,Stella,Abbot,Gustavus,Clark,Stewart,Monica
4	heels,noses,haze,hand,drooping,galloped,nakedness,pallid,anguish,palms
5	night,burdens,gut,assassins,witness,results,celestial,visual,deep,Hell
6	lifted,hastily,hovering,guiding,sinner,tendency,developing,sacrificed,condemned

Table 4: Example words in the clusters of 'body.n'

we found that for spectral clustering, Euclidean distances tend to perform better than Cosine distances. Also, the distributional counts of the second order words tend to perform better than Senna vectors which is not surprising because the Senna vectors are trained with the philosophy of a language model, which results in words often being clustered according to their POS tags rather than their semantic closeness. Spectral methods, yield slightly better results on two metrics than LDA based clustering which suggests that similarity matrices give us a better idea about interactions between groups of words than simple occurence frequencies of the words. But a bigger advantage of spectral clustering techniques is the speed of computing SVD which is much better than that of slow inference algorithms of LDA based models.

For LDA based models, we also note that different settings focusing on different POS tags, performed very similarly and did not indicate any strong preference for any POS tag for the task of WSI using LDA. Finally, both our methods tend to induce a small number of senses in the test data, which suggests that the induced senses are relatively coarse-grained. Further splitting of coarse clusters using hierarchical clustering methods might be helpful if a task requires finer-grained senses.

6 Qualitative Analysis

In this section, we present some deductions drawn from the qualitative analysis of clusters generated by our methods which support our hypothesis. In particular, we discus the nature of clusters generated by the spectral clustering algorithm using the second order distributional vectors for obtaining the similarity matrix based on Euclidean distance.

A preliminary analysis of cluster sizes revealed that in almost all the cases, one of the clusters was very large(about 3 times larger than the second largest cluster) and this largest cluster seemed to conflate a lot of senses. Other clusters were generally similar sized and most of them represented a sense of the target word on their own. The results in general look very promising and many clusters can be easily interpreted as different senses of the target word.

In Table 4, we show the top few words for the word 'body.n'. Some senses very clearly represent themselves : 1. Body as in organization, 2. Biological terms related to body, 4. Body in a more informal sense. Sense 5 seems like a mixture of two senses of body, one related to celestial bodies and other related to dead bodies/murder. Interestingly, sense 3 comprises proper nouns i.e. people whose bodies have been mentioned in the corpus. This is not a conventional sense listed in any of the sense inventories but based upon the requirements of a task, one might be interested in differentiating between general mentions of 'bodies' and mentions of 'bodies' which appear when mentioning famous people or celebrities. This sort of clustering can be incredibly useful in tasks like Machine Translation and Information Retreival which require us to model semantics of rare words such as important proper nouns.

7 Discussion and Future Work

We used a large corpus and its distributional statistics to perform word sense induction for a set of 100 target words. We proposed two algorithms which cluster the salient words surrounding the target word by using the distributions of surrounding words. Both LDA based algorithm and the spectral clustering algorithm yielded similar clusters. We believe that these clusters can be employed in downstream tasks and can be further broken into smaller fine grained clusters automatically if needed by the application.

We also evaluated our clusters arising from the distributional statistics, in the Semeval-2010 tasks without any tuning and showed that they perform competetively with other approaches.
We argue that treating existing sense inventories as gold standards for WSI tasks is not an appropriate measure for WSI systems because these inventories would not be able to measure two very important characteristics of WSI systems which make them more advantageous than supervised WSD systems:a) coverage and b) discovery of new senses.
Hence, the Semeval-2010 experiments are not an accurate reflection of the capabilities of WSI systems because they rely on the Ontonotes sense inventory for the Gold Standard judgements, which are admitted even by the OntoNotes builders to be only 85% reliable on average (Hovy et al., 2006). Our competetive performance on these tasks show that our methods can be compliant with standard word sense disambiguation tasks but more importantly, our qualitative analysis showed that our techniques can discover new unconventional senses too, which might not be present in the sense inventories but could be very useful in tasks requiring differentiations. Unfortunately, no metrics exist that can help us quantify the coverage of senses and their novelty. An ideal metric to evaluate the WSI systems in a better manner, would be their performance on extrinsic tasks like Machine Translation, Information Retreival, Machine Reading etc., which require differentiation of senses at different granular levels. WSI techniques have a potential of eliminating sense annotation costs hence enabling wider use of sense differentiation in a more generalized setting.
Our techniques resulted in coarse-grained senses. A major challenge in this task is to determine the appropriate number of senses to induce. To overcome this problem, non-parametric methods could be conceived to identify the ideal number of clusters automatically. In future, the WSI systems like ours can also be used to analyze the evolution of senses over a period of time or geographical variation of senses. As mentioned earlier, the co-occurence graph consists of many canonical representation of words which must be split according to their different senses. In our experiments, we considered a small number of target words and did not take into account the multiplicity of senses in the representation of 'first' and 'second' order neighbors. A more sophisticated iterative approach involving making several passes over a co-occurence graph and refining senses of different words in each pass can ameliorate the problem associated with a single canonical representation of neighboring words. Finally, designing extrinsic tasks to measure the efficacy of WSI systems will be extremely helpful in development of more robust and useful WSI systems.

Acknowledgments

This research was supported in part by DARPA grant FA8750-12-2-0342 funded under the DEFT program.

References

Eneko Agirre and Aitor Soroa. 2007. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 7–12. Association for Computational Linguistics.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 103–111. Association for Computational Linguistics.

Do Kook Choe and Eugene Charniak. 2013. Naive bayes word sense induction. In *EMNLP*, pages 1433–1437.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: the 90% solution. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, pages 57–60. Association for Computational Linguistics.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.

Jey Han Lau, Paul Cook, Diana McCarthy, David Newman, and Timothy Baldwin. 2012. Word sense induction for novel sense detection. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 591–601. Association for Computational Linguistics.

Suresh Manandhar and Ioannis P Klapaftis. 2009. Semeval-2010 task 14: evaluation setting for word sense induction & disambiguation systems. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 117–122. Association for Computational Linguistics.

Suresh Manandhar, Ioannis P Klapaftis, Dmitriy Dligach, and Sameer S Pradhan. 2010. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68. Association for Computational Linguistics.

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. 2011. Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176–182.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. 2002. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856.

Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–619. ACM.

Amruta Purandare and Ted Pedersen. 2004. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the Conference on Computational Natural Language Learning*, pages 41–48. Boston.

Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123.

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical dirichlet processes. *Journal of the american statistical association*, 101(476).

Jean Véronis. 2004. Hyperlex: lexical cartography for information retrieval. *Computer Speech & Language*, 18(3):223–252.

Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. 2012. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics.

Group based Self Training for E-Commerce Product Record Linkage

Wayne Xin Zhao[1,2], **Yuexin Wu**[2], **Hongfei Yan**[2] **and Xiaoming Li**[2]
[1]School of Information, Renmin University of China, China
[2]School of Electronic Engineering and Computer Science, Peking University, China
batmanfly@gmail.com, wuyuexin@gmail.com,
yhf1029@gmail.com, lxm@pku.edu.cn

Abstract

In this paper, we study the task of product record linkage across multiple e-commerce websites. We solve this task via a semi-supervised approach and adopt the self-training algorithm for learning with little labeled data. In previous self-training algorithms, the learner tries to convert the most confidently predicted unlabeled examples of each class into labeled training examples. However, they evaluate the confidence of an instance only based on the individual evidence from the instance. The correlation among data instances is rarely considered.

To address it, we develop a novel variant of the self-training algorithm by leveraging the data characteristics for the task of product record linkage. We joint consider a candidate linked pair and its corresponding correlated pairs as a group at the selection of pseudo labeled data. We propose a novel confidence evaluation method for a group of instances, and incorporate it as a re-ranking step in the self-training algorithm. We evaluate the novel self-training algorithm on two large datasets constructed based on real e-commerce Websites. We adopt several competitive methods as comparisons and perform extensive experiments. The results show that our method outperforms these baselines that do not consider data correlation.

1 Introduction

Recent years have witnessed the rapid development of online e-commerce business, e.g. Amazon and eBay, which raises the need for better storing, organizing and analyzing the large amount of product records. An important task is how to effectively link product records across multiple databases or websites. This task serves as a fundamental step for many applications. For example, it will be useful to provide entity-oriented search and product comparison analysis in eBay, where record linkage can help to unify the corresponding records (i.e. records from different sellers) given a product. Record linkage has been shown to be important in many fields, including biology (Needleman and Wunsch, 1970), database (Neiling, 2006) and text mining (Goiser and Christen, 2006; Bilenko and Mooney, 2003). In this paper, we mainly focus on the task of product record linkage for online e-commerce websites, but our method is easy to be extended to other data sources and tasks.

Early studies on record linkage were mainly based on the classical probabilistic approach developed by Fellegi and Sunter (1969), furthermore it was improved by the application of the expectation-maximization (EM) algorithm (Winkler, 1988) and the use of approximate string comparison algorithms (Christen, 2006; Winkler, 2006). The early work was not flexible to incorporate rich information. The development of machine learning techniques in the late $1990s$ provides a new approach for record linkage, and it has become the mainstream methodology for this task. The task of record linkage is usually re-casted as the record pair classification problem, i.e. whether a record pair refers to the same entity or not (Elfeky et al., 2002; Neiling, 2006; Tejada et al., 2002; Nahm et al., 2002). Supervised methods can also be used to learn distance measures for approximate string comparisons (Bilenko and Mooney, 2003;

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1311–1321, Dublin, Ireland, August 23-29 2014.

Cohen et al., 2003). Although supervised techniques often achieve good linkage quality, they are largely limited by the availability of the training data.

To address this problem, semi-supervised learning approaches aim to make good use of a small portion of labeled and a large amount of unlabeled data to build a better classifier (Yarowsky, 1995). Self-training is a commonly used algorithm for semi-supervised learning, where in each iteration the learner converts the most confidently predicted unlabeled examples of each class into labeled training examples. It has been successfully applied to many tasks, such as sentiment analysis (He and Zhou, 2011; Riloff et al., 2003) and object detection from images (Rosenberg et al., 2005).

In this paper, we solve the task of product record linkage via a semi-supervised approach and adopt the flexible self-training framework for learning with little labeled data. We propose a novel variant of the self-training algorithm by incorporating the correlation existing in the data instances, which is rarely studied in previous studies. To introduce our idea, we first present an illustrative example in Figure 1. There are two databases $\mathcal{D}$ and $\mathcal{D}'$, and we have three records $r_1, r_2, r_3 \in \mathcal{D}$ and another three records $r'_1, r'_2, r'_3 \in \mathcal{D}'$. Furthermore, we assume r_1 and r'_1 refer to the same product. We can see that r_1 is involved in three candidate pairs, i.e. (r_1, r'_1), (r_1, r'_2) and (r_1, r'_3). Similarly, r'_1 is involved in three candidate pairs, i.e. (r'_1, r_1), (r'_1, r_2) and (r'_1, r_3). Usually, each individual database does not contain duplicate records, once we know r_1 is linked to r'_1, we can infer the rest candidate pairs should not be linked. In other words, only if we are confident that no pair in the set $\{(r_1, r'_2), (r_1, r'_3), (r_2, r'_1), (r_3, r'_1)\}$ is not linked, r_1 is likely to be linked with r'_1.

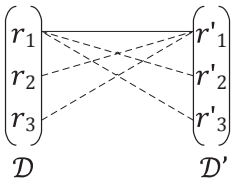


Figure 1: An illustrative example for correlation among record pairs. The real line denotes the real linkage relation and the dash line denotes the candidate linkage relation.

For the task of record linkage, the number of positive instances (i.e. linked record pairs) are usually much less than that of negative instances. We mainly consider the confidence evaluation of the candidate positive instance. By following the above idea, given a *candidate* linked pair, we treat all the correlated record pairs together as a group and evaluate the linkage confidence based on the evidence of all record pairs in this group, i.e. *group confidence evaluation*. We incorporate the group confidence evaluation into the self-training algorithm as a re-ranking step. Interestingly, once we have identified a linked pair, the rest correlated record pairs can be naturally judged as negative instances. We evaluate the novel self-training algorithm on two large datasets constructed based on real e-commerce Websites. We adopt several competitive methods as comparisons and perform extensive experiments. The results show that our method outperforms these baselines that do not consider data correlation.

2 Related Work

We have briefly described the supervised approaches for record linkage in the introduction. Now we discuss other related studies, including unsupervised clustering techniques, genetic programming based approaches and linking based on more complex constraints.

Unsupervised clustering techniques have been investigated both for improved blocking (Cohen and Richman, 2002; McCallum et al., 2000) and for automatic record pair classification (Elfeky et al., 2002). Usually, such techniques do not perform not as well as supervised approaches.

Most recently, genetic programming (GP) (Koza et al., 1999) has also been utilized to the task of record linkage. GenLink (Isele and Bizer, 2012) is a GP-based supervised learning algorithm in order to learn linkage rules from a set of existing reference links, which also suffers from the problem of lack of labeled data. Ngomo and Lyko (2013) evaluated linear and boolean classifiers against classifiers

computed by using genetic programming for the record linkage problem. Their experiments showed that both approaches did not perform well on real data.

Some other studies exploit more complex constraints that include relationships between different entity types to link all types of entities in coordination (Bhattacharya and Getoor, 2007; Dong et al., 2005; On et al., 2007). The usage of such constraints can indeed help to get better linkage results, but is in many cases domain-dependent. We try to develop an approach which can be applicable across domains.

In order to address the problem of limited labeled data, we mainly consider the semi-supervised approaches. There are rarely semi-supervised approaches specially for the record linkage problem. Some studies on improving self-training algorithms are related to our work. Self-training with editing (Li and Zhou, 2005) can help to reduce mislabeled pseudo training examples, and reserved self-training (Guan and Yang, 2013) is designed for handling imbalanced data. We have very different focus with theirs, i.e. incorporating the instance correlations into learning algorithms, which can applied to other self-training variants.

3 Problem Definition

In this section, we first introduce the preliminary related to our task. Then we formally define our studied task.

Product record. A product record r is characterized by a referred product entity e and a set of attribute values $\mathcal{V} = \{(v_i)\}_i$, where v_i denotes the value of the ith attribute in r. We use $r.e$ and $r.\mathcal{V}$ to index the product entity and attribute value set of the record r respectively. A product record corresponds to a unique product entity but a product entity can map to multiple product records across multiple databases. Attribute values are represented as strings, i.e. a sequence of characters. An attribute of a product might correspond to different descriptive text across websites.

Product record linkage. The task of product record linkage is to judge whether two product records refer to the same product entity. Given two product records r and r', we aim to judge whether $r.e$ is the same to $r'.e$. Usually, r and r' come from different product databases. Although different product databases can have different attributes for the same product and different attribute names for the same attribute, we make an assumption about the task: *candidate record pairs share the same set of attributes*. It is relatively easy to automatically identify common attributes and align attributes (Härder et al., 1999; Rundensteiner, 1999; Hassanzadeh et al., 2013), which is not our focus in this paper. We mainly study product record linkage under the same set of attributes, and this assumption makes our study more focused. If r and r' refer to the same product entity, denoted by $r \sim r'$; otherwise, we denote it by $r \not\sim r'$.

4 A General Machine Learning based Approach

Given a product type, as we mentioned above, we assume that it corresponds to a specific set of attributes, and all the product records share the same set of attributes but possibly with different descriptive text for attribute values. In this section, we further present a general supervised approach with similarity features.

4.1 Defining the similarity function

Given two product records r and r', we can obtain the similarity between their descriptive text of an attribute by using a similarity function. The major intuition is that if two records refer to the same product, they should have similar text for the same attribute, i.e. the similarity function should return a large similarity value. Let $f(\cdot, \cdot)$ denote a similarity function, which takes two text strings and returns a similarity value within the interval $[0, 1]$ for these two strings. As revealed in (Bilenko and Mooney, 2003), different attributes or fields may need different similarity functions to achieve best similarity evaluation. Thus, instead of fixing a single similarity function, we consider using the following widely used similarity functions: 1) Exact match; 2) Cosine similarity; 3) Jaccard coefficient; 4) K-Gram similarity (Kondrak, 2005); 5) Levenshtein similarity (Levenshtein, 1966); 6) Affine Gap similarity (Needleman and Wunsch, 1970).

4.2 The learning framework

Based on these similarity functions, we propose a general learning framework for product record linkage by using similarity values of different fields as features.

Given a product type, we assume that there are A attributes and K similarity functions. For two records r and r', we can obtain a similarity feature vector $\mathbf{x} = [x_{a,k}]_{i=1,k=1}^{A,K}$, which is indexed by an attribute and a similarity function: $x_{a,k}$ denotes the similarity of the ath attribute between r and r' by using the kth similarity function. Furthermore, each feature vector $\mathbf{x}$ will correspond to a unique binary label y which indicates that r and r' refer to the same product entity. Given a set of record pairs and their linkage labels $\{(\mathbf{x}, y)\}$, we can learn a classifier which is able to predict the linkage label given the similarity feature vector of two records. To this end, we have reformulated the task of product record linkage as a binary classification problem. Any classifiers can be used for this task. In what follows, we will use *instances* and *candidate pairs* alternatively.

5 Group based Self-Training

In the above, we have presented a supervised learning approach for product record linkage. The approach is easy to apply in practice, however, the performance is largely limited by the availability of training data. For our current task, i.e. product record linkage, the generation of labeled data becomes even much harder: there are usually many product types and it is infeasible to create a large amount of labeled data for each type. Although it is difficult to obtain labeled data, we can easily obtain sufficient unlabeled data. Thus, in this paper, we study the task of product record linkage in a semi-supervised setting by leveraging both the learning ability of the classifiers and the usefulness of the large amount of unlabeled data. We propose a novel group based self-training algorithm for product record linkage. Before introducing our method, we first introduce the general self-training algorithm.

5.1 The general self-training algorithm

Self-training is a semi-supervised learning algorithm. It starts training on labeled data only, after each iteration, the most confidently predicted unlabeled samples would be incorporated as new labeled data, i.e. pseudo labeled data, decided by confidence scores from the classifier. After several iterations, it is expected to get a better classifier trained with both labeled data and pseudo labeled data. The general procedure of self-training algorithm is summarized in Algorithm 1.

Algorithm 1: The general procedure of the self-training algorithm.

```
Input: labeled dataset L, unlabeled dataset U, the classifier C.
U' ← S randomly selected examples from U, S is usually set to 0.5 × |U|;
repeat
    Training the classifier: Use L to train C, and label the examples in U';
    Selecting pseudo labeled data: Select T most confidently classified examples from U' and add them to L;
    Filling unlabeled data: Refill U' with examples from U, to keep U' at a constant size of S examples.
until I iterations or U = ∅;
return The extended labeled dataset L and the trained classifier C.
```

We can see that self-training is a wrapper algorithm by taking a classifier as the learning component, and it has three major steps in an iteration: 1) training classifier; 2) selecting pseudo labeled data; and 3) filling unlabeled data. Among the three steps, the most important step is the pseudo labeled data selection. Previously, the most commonly used method is to select the top confident instances of the classifier, and it is easy to see that the performance of self-training relies on the learning ability of the embedded classifier.

5.2 Group confidence evaluation

Recall that each instance is a pair of product records (r, r') and their label indicates whether they should be linked or not. Let $P^L(r, r')$ denote the confidence that r and r' refer to the same product entity (linked

confidence), and $P^N(r, r')$ denote the confidence that r and r' refer to different product entities (non-linked confidence). $P^L(r, r')$ and $P^N(r, r')$ can be estimated by the confidence scores from the classifier. In the task of product record linkage, there are usually more negative instances, i.e. the number of non-linked pairs is much more than that of linked pairs. Thus, we mainly study the confidence of a candidate positive instance. The standard self-training algorithm selects top ranked positive instances according to the confidence scores estimated by the classifier, i.e. we select the instances with large linked confidence $P^L(\cdot, \cdot)$. However, when applied to product record linkage, it ignores important characteristics underlying the data, which will be potentially helpful to the task.

Let us examine the illustrative example in Figure 1. Recall that r_1 and r'_1 refer to the same product, i.e. $r_1 \sim r'_1$. We can see that r_1 is involved in three candidate pairs, i.e. (r_1, r'_1), (r_1, r'_2) and (r_1, r'_3). Similarly, r'_1 is involved in three candidate pairs, i.e. (r'_1, r_1), (r'_1, r_2) and (r'_1, r_3). We totally have a set of five candidate pairs, i.e. $\{(r_1, r'_1), (r_1, r'_2), (r_1, r'_3), (r_2, r'_1), (r_3, r'_1)\}$. Here we follow the assumption of the one-to-one mapping, i.e. given two databases, a product record can link to at most one record in the other database. By leveraging the correlation among candidate pairs, with $r_1 \sim r'_1$, we can infer the rest four candidate pairs must not be linked, i.e. $r_1 \not\sim r'_2, r_1 \not\sim r'_3, r_2 \not\sim r'_1, r_3 \not\sim r'_1$. Next, we formally characterize the above idea and present the algorithm. Given two databases $\mathcal{D}$ and $\mathcal{D}'$, let $\mathcal{C} \subset \mathcal{D} \times \mathcal{D}'$ denote the candidate pair set where two product records in a pair come from $\mathcal{D}$ and $\mathcal{D}'$ respectively. Consider a candidate pair $(r, r') \in \mathcal{C}$, where $r \in \mathcal{D}, r' \in \mathcal{D}'$. We consider the following two sets: $\mathcal{S}_r = \{(r, b)|(r, b) \in \mathcal{C}, b \in \mathcal{D}' \text{ and } b \neq r'\}$ and $\mathcal{S}_{r'} = \{(a, r')|(a, r') \in \mathcal{C}, a \in \mathcal{D} \text{ and } a \neq r\}$. Intuitively, if we know $r \sim r'$, then all the pairs in both $\mathcal{S}_r$ and $\mathcal{S}_{r'}$ must not be linked. Thus, we define **the conflicting set** of pair (r, r') as $\mathcal{S}_{cfl}^{r,r'} = \mathcal{S}_r \cup \mathcal{S}_{r'}$.

With the definition of the conflicting set, let us reconsider the pseudo labeled data selection. The straightforward way is to evaluate each instance with their linked confidence $P^L()$ from the classifier. However, it oversimplifies the data dependence and does not make use of the correlated characteristics. Consider an instance, which is a record pair (r, r'), we can have the following two properties:

- If $r \sim r'$, then $\forall (a, b) \in \mathcal{S}_{cfl}^{r,r'}$, we have $a \not\sim b$;
- If $\exists (a, b) \in \mathcal{S}_{cfl}^{r,r'}$ and $a \sim b$, then we have $r \not\sim r'$.

The above properties suggest that it should be helpful to consider the correlation among instances when evaluating the confidence of a positive instance, i.e. a candidate linked record pair. Intuitively, if two records refer to the same product entity, they should have large linked confidence and their conflicting pairs should have large non-linked confidence. We propose to use the following method to evaluate the linkage confidence between r and r'

$$\text{Conf}(r, r') = P^L(r, r') \Big(\prod_{(a,b) \in \mathcal{S}_{cfl}^{r,r'}} P^N(a, b) \Big)^{1/M}, \tag{1}$$

where $M = |\mathcal{S}_{cfl}^{r,r'}|$, $P^L(\cdot, \cdot)$ and $P^N(\cdot, \cdot)$ are positive and negative confidence scores estimated by the classifier respectively. Note that we take the geometric mean of the non-linked confidence of these conflicting pairs, which is to reduce the affect of large outlier values and the varying size of the conflict sets. We treat a candidate linked pair and all the candidate pairs in its conflicting set as a group. The group confidence evaluation consists of two intuitions: 1) the confidence that two records should be linked; 2) the confidence that any pair of records in the conflicting set must not be linked. We have taken these two aspects into a unified evaluation score.

5.3 The proposed self-training algorithm

In this part, we present the novel self-training algorithm based on the group confidence evaluation. We have the similar steps with the general self-training algorithm in Algorithm 1. The major focus is to modify the step of *pseudo labeled data selection*. As mentioned above, we mainly consider the confidence evaluation of positive instances. Our method for pseudo labeled data selection is three-step process:

- Select top T' most confidently classified positive examples by the classifier;
- Rerank these T' examples by the group confidence scores defined in Equation 1;
- Select top T examples from the reranked T' examples ($T \leq T'$) as pseudo positive instances and their corresponding conflicting instances in the conflicting sets as pseudo negative instances.

We select positive instances not only based on the instance itself but also their corresponding conflicting instances: if we have high confidence about a positive instance, then the confidence of their conflicting instances being negative should be high, too. Next, we present the detailed group based self-training algorithm in Algorithm 2.

Algorithm 2: The procedure of the group based self-training algorithm.

Input: labeled dataset $\mathcal{L}$, unlabeled dataset $\mathcal{U}$, the classifier $\mathcal{C}$.
$\mathcal{U}' \leftarrow S$ randomly selected examples from $\mathcal{U}$;
repeat
  Training the classifier: Use $\mathcal{L}$ to train $\mathcal{C}$, and label the examples in $\mathcal{U}'$;
  Selecting pseudo labeled data selection:
  - Select T' most confident positive examples from $\mathcal{U}'$ and add them to $\mathcal{L}$;
  - Calculate the group confidence scores for the T' examples according to Equation 1.
  - Rerank these T' examples by their group confidence scores and add top T examples to $\mathcal{L}$ as the pseudo positive instances.
  - For each of the T examples, add their conflicting instances to $\mathcal{L}$ into as the pseudo negative instances.

  Filling unlabeled data: Refill $\mathcal{U}'$ with examples from $\mathcal{U}$, to keep $\mathcal{U}'$ at a constant size of S examples.
until *I iterations or $\mathcal{U} = \emptyset$*;
return *The extended labeled dataset $\mathcal{L}$ and the trained classifier $\mathcal{C}$.*

On one hand, our group based self-training algorithm naturally exploits the correlation among data instances and evaluate the confidence scores in a broader view, which avoids the decision conflicts caused by the data dependence. On the other hand, we focus on evaluating the confidence of being a positive instance, which further reduces the bias from imbalanced data distribution. Thus, it is expected to achieve better performance in the task of product record linkage.

Most classifiers can provide the estimated confidence scores $P^L()$ (i.e. for a positive instance) and $P^N()$ (i.e. for a negative instance): Maximum-Entropy models output the conditional probabilities of an instance for each class (Berger et al., 1996); the Decision Tree C4.5 algorithm is also able to compute the probability distribution over different classes for each instance (Quinlan, 1993).

6 Experiments

6.1 Construction of the test collection

We test our method on two real e-commerce datasets respectively from Jingdong[1] and eTao[2]. Jingdong is the largest B2C e-commerce company and eTao is one of the largest product search portals in China. Due to the extremely large product databases, it is infeasible to generate training data on each product type for these two product databases. We consider two popular kinds of products: laptop and camera. These two kinds of products cover a considerable amount of brands and models, especially suitable for the test of record linkage. Both Jindong and eTao have set up specific categories for these two kinds of products respectively, thus we can easily crawl the product records under the corresponding category label. To generate linked record datasets, we first manually align attributes (i.e. fields) for these two kinds between Jindong and eTao. We summarize the numbers of aligned fields and some example fields in Table 1. Not all the records contain the information for all the fields, we set the value of the empty field to a "*NULL*" string.

[1] http://www.jd.com
[2] http://www.etao.com

We adopt a blocking approach (Baxter et al., 2003) to automatically generate a set of candidate pairs, i.e. a record in Jindong is to be linked with a record in eTao. This approach consider all pairwise links between Jindong records and eTao records for the same kind of product. If there exists at least one common word in the field of *brand* or *model* between a record pair, we consider it to be a candidate pair. The automatic method generates 20,094 candidate pairs and 12,157 candidate pairs respectively for *LAPTOP* and *CAMERA*. Then we invite professional workers from an e-commerce company to link records across these two product databases. Instead of examining all the candidate pairs, the labeling process adopts a product-oriented way to generate the gold standard. Given a product record of a database, the annotator first identifies the product entity that the record refers to, then she looks for the corresponding record in another database. In the annotation process, Web access is available all the time. Annotators can make use of the search engines of Jindong and eTao to accelerate the product lookup. A linked record pair is treated as a positive instance. Finally, we identify 501 linkable products (i.e. 501 positive instances) in LAPTOP dataset, and 478 linkable products (i.e. 478 positive instances) in CAMERA dataset. All the other candidate pairs are automatically labeled as *negative*. We present the the data statistics in Table 1.

Dataset	# positive instances	# negative instances	# fields	Example fields
LAPTOP	501	19593	10	OS, screen size, CPU type, ram size
CAMERA	478	11679	11	lens type, sensor type, focal length, aperture size

Table 1: Basic statistics of datasets.

6.2 Experimental setup

For each kind of product, we divide the dataset into two parts, i.e. a training set and a test set. In order to examine different methods in a semi-supervised setting, we keep a small amount of instances in the training set, and we assume all the methods can use of the data (without labels) in the test set. There are more negative instances, we mainly consider the amount of positive instances, and the number of positive instances is called as *the number of seeds*. We randomly generate the training set with the given number of seeds. Once we add one positive instance into the training set, we add all the its conflicting instances into the training set. This is to reduce the correlation between training instances and testing instances for a fair comparison. In later experiments, given the seed number, we will generate ten random training sets and take the average of ten runs as the final performance. In later experiments, we do not explicitly report the number of negative instances unless needed.

We adopt three widely used evaluation metrics for the classification task: Precision, Recall and the F-measure [3].

We compare the following methods for the task of product record linkage:

- *Supervised Classifier* (**SC**): the standard supervised classifier, which does not consider the unlabeled data at all.
- *Traditional Self-Training* (**t-ST**): the traditional self-training method in Algorithm 1 which adds an equal amount of samples of each class in pseudo labeled selection at each iteration.
- *Proportional Self-Training* (**p-ST**): the traditional self-training method in Algorithm 1 but add samples according to the class distribution at each iteration.
- *Simple Group Based Self-Training* (**s-ST**): a simplified version of our approach without the group confidence valuation, which directly selects samples of high confidence scores estimated from the classifier together with their conflicting pairs as negative samples at each iteration.
- *Group Based Self-Training* (**g-ST**): the proposed group based self-training algorithm in Algorithm 2, which uses the group confidence evaluation method to select pseudo positive instances.

[3] http:/en.wikipedia.org/wiki/Precision_and_recall

Recall all the methods rely on the wrapped classifier. We select two classic but very different classifiers: the Maximum Entropy model (***MaxEnt***) and the Decision Tree $C4.5$ (***Tree***). We implement these two classifiers using the machine learning toolkit Weka[4]. We use the six similarity functions to obtain similarity values between two records on each field as features. All the self-training based methods run ten iterations and at each iteration they add the same number of positive instances, i.e. 30. Different methods select pseudo *negative instances* differently. *t-ST* does not consider the correlation between data instances, and it adds top 30 confident negative instances. *p-ST* adds top $30 \times \frac{\#\text{negative instances}}{\#\text{positive instances}}$ confident negative instances. Both *p-ST* and *g-ST* take all the conflicting instances of the selected pseudo positive instances as the negative instances. We present the average numbers of pseudo negative instances at an iteration in Table 2. As will be revealed later, although *p-ST* adds more negative instances, *g-ST* performs much better than *p-ST*, which indicates simply adding more negative instances might not lead to better performance. We do not perform specific preprocessing steps to make the data balanced (e.g. under-sampling or over-sampling), and we find the data distribution does not significantly affect the performance of the classifiers on our dataset.

Dataset	t-ST	p-ST	s-ST	g-ST
LAPTOP	30	950	845	854
CAMERA	30	655	569	584

Table 2: Average numbers of pseudo negative instances selected at each iteration.

6.3 Results and analysis

Overall performance comparison. To test the performance under weak supervision, we first set the seed number to 30, which nearly takes up a proportion of 5% of the labeled data. We present the results of different methods in Table 3 and Table 4. We first examine the performance of the baselines. We can see that semi-supervised learning is very effective to improve over the the supervised classifier when the amount of training data is small. It is interesting to see that *s-ST* performs best among all the baselines. Recall that the major difference between *s-ST* and other baselines is that it select the conflicting pairs of the pseudo positive instances as the negative instances. It indicates that it is important to consider the correlation among the data instances. In addition, Decision Tree seems to be more competitive than Maximum Entropy Model for product record linkage. Then we take our group based self-training algorithm into comparison. In terms of F1 measure, we can see that it is consistently better than all the baselines on two datasets respectively by using two different classifiers. It is worth looking into the performance comparison on precision and recall. We can see that (1) *s-ST* and *g-ST* yield better results in terms of precision while the other baselines yield better results in terms of recall; (2) our method *g-ST* largely improves over the best baseline *s-ST*. It is not surprising to have these observations since that our group evaluation method is more careful at the selection of pseudo positive instance: it considers the evidence from the conflicting instances.

Methods	MaxEnt			Decision Tree		
	P	R	F1	P	R	F1
SC	0.246	0.910	0.382	0.301	**0.931**	0.454
t-ST	0.264	**0.925**	0.411	0.328	0.921	0.484
p-ST	0.350	0.831	0.487	0.412	0.887	0.539
s-ST	**0.979**	0.632	0.767	0.909	0.754	0.823
g-ST	0.936	0.742	**0.826**	**0.912**	0.843	**0.876**

Table 3: Results on LAPTOP dataset.

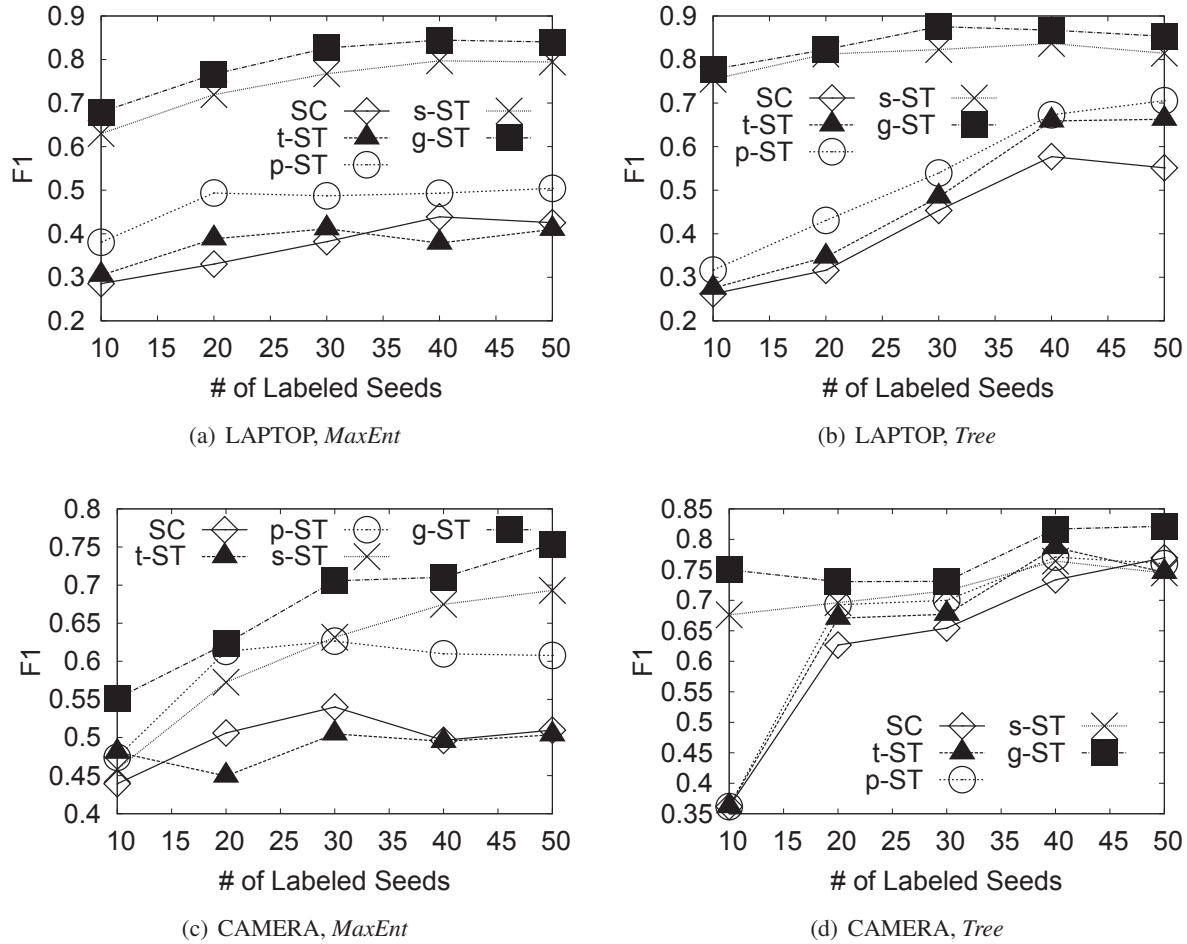
Parameter tuning. In the above, we have shown the results of different methods with 30 positive instances. The number of seeds is particularly important for self-training algorithms, and we want to ex-

[4]http://www.cs.waikato.ac.nz/ml/weka

Methods	MaxEnt			Decision Tree		
	P	R	F1	P	R	F1
SC	0.387	0.891	0.540	0.493	**0.965**	0.652
t-ST	0.352	**0.892**	0.504	0.537	0.963	0.677
p-ST	0.501	0.871	0.626	0.573	0.942	0.700
s-ST	**0.931**	0.479	0.632	0.962	0.570	0.716
g-ST	0.917	0.574	**0.706**	**0.965**	0.588	**0.731**

Table 4: Results on CAMERA dataset.

amine how it affects the performance of these methods. By varying the number of seeds from 10 to 50 with a step of 10, we present the F1 results in Figure 2 on two datasets by using two classifiers. We can see that our method is consistently better than baselines with the varying of the seed number. Especially, our method still works well when there is little labeled data, i.e. $\#seeds = 10$. With a weaker classifier, i.e. *MaxEnt*, our method yields more improvement than that with *Tree*. Besides the seed number, there are another two factors which potentially affect the performance: (1) the iteration number and (2) the number of pseudo positive instances selected at each iteration. We also examine the tuning results of these two parameters and find our method is consistently better than *s-ST* with the varying of these two factors. These results show that our method is very effective and it is of high stability and practicability.



(a) LAPTOP, *MaxEnt*

(b) LAPTOP, *Tree*

(c) CAMERA, *MaxEnt*

(d) CAMERA, *Tree*

Figure 2: Performance comparison with varying seed numbers (i.e. # of positive instances).

7 Conclusion

In this paper, we develop a novel variant of the self-training algorithm by leveraging the data characteristic for the task of product record linkage. We joint consider a candidate linked pair and its corresponding correlated pairs as a group, at the selection of pseudo labeled data. We propose a confidence evaluation method for a group of instances, and incorporate it as a re-ranking step in the self-training algorithm. We evaluate the novel self-training algorithm on two large datasets constructed based on real e-commerce Websites. We adopt several competitive methods as comparisons and perform extensive experiments. The results show that our method outperforms these baselines that do not consider data correlation. We also carefully examine the affects of various parameters, and the tuning results indicate the stability and robustness of our method.

The major contribution and novelty of this paper is the novel group confidence evaluation to model the correlation existing in data. Although we develop the idea in the setting of self-training algorithms, it will be promising to be applied in other learning algorithms, i.e. active learning.

Acknowledgements

We thank the anonymous reviewers for his/her thorough review and highly appreciate the comments. This work was partially supported by the National Key Basic Research Program (973 Program) of China under grant No. 2014CB340403, 2014CB340405 and NSFC Grant 61272340. Xin Zhao was supported by MSRA PhD fellowship. Xin Zhao and Yuexin Wu contributed equally to this work and should be considered as joint first authors. Xin Zhao is the corresponding author.

References

Rohan Baxter, Peter Christen, and Tim Churches. 2003. A comparison of fast blocking methods for record linkage. In *ACM SIGKDD*, volume 3, pages 25–27. Citeseer.

Adam L Berger, Vincent J Della Pietra, and Stephen A Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Indrajit Bhattacharya and Lise Getoor. 2007. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):5.

Mikhail Bilenko and Raymond J Mooney. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 39–48. ACM.

Peter Christen. 2006. A comparison of personal name matching: Techniques and practical issues. In *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, pages 290–294. IEEE.

William W Cohen and Jacob Richman. 2002. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 475–480. ACM.

William W Cohen, Pradeep D Ravikumar, Stephen E Fienberg, et al. 2003. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, volume 2003, pages 73–78.

Xin Dong, Alon Halevy, and Jayant Madhavan. 2005. Reference reconciliation in complex information spaces. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 85–96. ACM.

Mohamed G Elfeky, Vassilios S Verykios, and Ahmed K Elmagarmid. 2002. Tailor: A record linkage toolbox. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 17–28. IEEE.

Ivan P Fellegi and Alan B Sunter. 1969. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210.

Karl Goiser and Peter Christen. 2006. Towards automated record linkage. In *Proceedings of the fifth Australasian conference on Data mining and analystics-Volume 61*, pages 23–31. Australian Computer Society, Inc.

Zhiguang Liu Xishuang Dong Yi Guan and Jinfeng Yang. 2013. Reserved self-training: A semi-supervised sentiment classification method for chinese microblogs.

Theo Härder, Günter Sauter, and Joachim Thomas. 1999. The intrinsic problems of structural heterogeneity and an approach to their solution. *The VLDB Journal*, 8(1):25–43.

Oktie Hassanzadeh, Ken Q Pu, Soheil Hassas Yeganeh, Renée J Miller, Lucian Popa, Mauricio A Hernández, and Howard Ho. 2013. Discovering linkage points over web data. *Proceedings of the VLDB Endowment*, 6(6):445–456.

Yulan He and Deyu Zhou. 2011. Self-training from labeled features for sentiment analysis. *Information Processing & Management*, 47(4):606–616.

Robert Isele and Christian Bizer. 2012. Learning expressive linkage rules using genetic programming. *Proceedings of the VLDB Endowment*, 5(11):1638–1649.

Grzegorz Kondrak. 2005. N-gram similarity and distance. In *String Processing and Information Retrieval*, pages 115–126. Springer.

John R Koza, Forrest H Bennett III, and Oscar Stiffelman. 1999. *Genetic programming as a Darwinian invention machine*. Springer.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, volume 10, page 707.

Ming Li and Zhi-Hua Zhou. 2005. Setred: Self-training with editing. In *Advances in Knowledge Discovery and Data Mining*, pages 611–621. Springer.

Andrew McCallum, Kamal Nigam, and Lyle H Ungar. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178. ACM.

Un Yong Nahm, Mikhail Bilenko, and Raymond J Mooney. 2002. Two approaches to handling noisy variation in text mining. In *Proceedings of the ICML-2002 workshop on text learning (TextML2002)*, pages 18–27. Citeseer.

Saul B Needleman and Christian D Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453.

Mattis Neiling. 2006. Identification of real-world objects in multiple databases. In *From Data and Information Analysis to Knowledge Engineering*, pages 63–74. Springer.

Axel-Cyrille Ngonga Ngomo and Klaus Lyko. 2013. Unsupervised learning of link specifications: Deterministic vs. non-deterministic. *Ontology Matching*, page 25.

Byung-Won On, Nick Koudas, Dongwon Lee, and Divesh Srivastava. 2007. Group linkage. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 496–505. IEEE.

John Ross Quinlan. 1993. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann.

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 25–32. Association for Computational Linguistics.

Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. 2005. Semi-supervised self-training of object detection models.

Elke Rundensteiner. 1999. Special issue on data transformation. *IEEE Techn. Bull. Data Engineering*, 22(1).

Sheila Tejada, Craig A Knoblock, and Steven Minton. 2002. Learning domain-independent string transformation weights for high accuracy object identification. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 350–359. ACM.

William E Winkler. 1988. Using the em algorithm for weight computation in the fellegi-sunter model of record linkage. In *Proceedings of the Section on Survey Research Methods, American Statistical Association*, volume 667, page 671.

William E Winkler. 2006. Overview of record linkage and current research directions. In *Bureau of the Census*. Citeseer.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics.

Reducing Over-Weighting in Supervised Term Weighting for Sentiment Analysis

Haibing Wu
Department of Electronic Engineering
Fudan University
Shanghai, China
haibingwu13@fudan.edu.cn

Xiaodong Gu
Department of Electronic Engineering
Fudan University
Shanghai, China
xdgu@fudan.edu.cn

Abstract

Recently the research on supervised term weighting has attracted growing attention in the field of Traditional Text Categorization (TTC) and Sentiment Analysis (SA). Despite their impressive achievements, we show that existing methods more or less suffer from the problem of *over-weighting*. Overlooked by prior studies, over-weighting is a new concept proposed in this paper. To address this problem, two regularization techniques, *singular term cutting* and *bias term*, are integrated into our framework of supervised term weighting schemes. Using the concepts of over-weighting and regularization, we provide new insights into existing methods and present their regularized versions. Moreover, under the guidance of our framework, we develop a novel supervised term weighting scheme, regularized entropy (*re*). The proposed framework is evaluated on three datasets widely used in SA. The experimental results indicate that our *re* enjoys the best results in comparisons with existing methods, and regularization techniques can significantly improve the performances of existing supervised weighting methods.

1 Introduction

Sentiment Analysis (SA), also known as opinion mining, has enjoyed a burst of research interest with growing avenues (e.g., social networks and e-commerce websites) for people to express their sentiments on the Internet. A typical sentiment-analysis application mainly involves three key subtasks, namely holder detection, target extraction and sentiment classification (Liu, 2012; Hu and Liu, 2004). A simple and most extensively studied case of sentiment classification is sentiment polarity classification, which is the binary classification task of labelling the polarity of a sentiment-oriented document as positive or negative. Sentiment classification can be performed at the document, sentence, phase or word level. In this paper, we focus on sentiment polarity classification at document level.

Just like Information Retrieval (IR) and TTC, in sentiment classification, the content of an opinion-orientated document can be represented as a vector of terms in light of Vector Space Model (VSM). In VSM, each dimension of the vector corresponds to a term and different terms have different weights, thus the term weight represents the contribution of the term to the sentiment of a document in sentiment classification. Term weighting is the task of assigning appropriate weights to terms according to their correlations with the category concept. Term weighting schemes fall into two categories (Lan et al., 2009; Debole and Sebastiani, 2003). The first one, known as *unsupervised* term weighting method, does not take category information into account. The second one referred to as *supervised* term weighting method embraces the category label information of training documents in the categorization tasks. Although most term weighting approaches to text categorization, including sentiment classification, are borrowed from IR, recently several new supervised term weighting schemes have been studied and achieved significant successes in TTC and SA (Lan et al., 2009; Martineau and Finin, 2009; Paltoglou and Thelwall, 2010).

Despite the impressive achievements in the current field of supervised term weighting for TTC and SA, we indentify that existing supervised methods, more or less, suffer from over-weighting problem and thus develop a robust framework to address this problem. Over-weighting, overlooked by prior studies, is a new concept introduced in this paper. It would occur due to the presence of many noisy

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1322–1330, Dublin, Ireland, August 23-29 2014.

words and the unreasonably too large ratios between weights of different terms. Thus, it could result in poor representations of sentiments containing in documents. In order to reduce over-weighting problem for supervised term weighting, two regularization techniques called *singular term cutting* and *bias term* are proposed and integrated into our framework of supervised term weighting schemes. Singular term cutting is introduced to cut down the weights of noisy or unusual terms, and bias term is added to shrink the ratios between weights of different terms.

Using the concepts of over-weighting and regularization, we provide new insights into existing supervised weighting methods and then present their regularized versions. We also propose a novel term weighting scheme called *regularized entropy* (*re*) under the guidance of our framework. The formulation of *re* bases on entropy, which is used to measure the distribution of terms over different categories, and the terms with smaller entropy value have larger weights.

After presenting our framework, the regularized versions of existing methods and *re* in detail, experiments are conducted on three publicly available datasets widely used in SA. In our experiments, *re* is compared against many existing methods appearing in IR, TTC and SA. We also compare the performances of existing supervised weighting methods against their regularized versions. The results of comparative experiments indicate that *re* clearly outperform existing methods, the introduction of regularization techniques significantly improves the performances of existing supervised weighting methods.

2 Review of Term Weighting Schemes in IR, TTC and SA

In IR, TTC and SA, one of the main issues is the representation of documents. VSM provides a simplifying representation by representing documents as vector of terms. Term weighting aims to evaluate the relative importance of different terms in VSM. There are three components in a term weighting scheme, namely local weight, global weight and normalization factor (Salton and Buckley, 1988; Lan et al., 2009). Final term weight is the product of the three components:

$$t_{ij} = l_{ij} \times g_i \times n_j \,, \tag{1}$$

where t_{ij} is the final weight of i_{th} term in the j_{th} document, l_{ij} is the local weight of i_{th} term in the j_{th} document, g_i is the global weight of the i_{th} term, and n_j is the normalization factor for the j_{th} document.

2.1 Local Term Weighting Schemes

Local weight component is derived only from frequencies within the document. Table 1 lists three common local weighting methods, namely raw term frequency (*tf*), term presence (*tp*) and augmented term frequency (*atf*). In IR and TTC, the most widely used local weight is *tf* , but pioneering research

Local weight	Notation	Description
tf	tf	Raw term frequency.
$\begin{cases} 1, & \text{if } tf > 0 \\ 0, & \text{otherwise} \end{cases}$	tp	Term presence, 1 for presence and 0 for absence.
$k + (1-k)\dfrac{tf}{\max_t(tf)}$	atf	Augmented term frequency, $\max_t(tf)$ is the maximum frequency of any term in the document, k is set to 0.5 for short documents (Salton and Buckley, 1988).

Table 1: Local term weighting schemes.

Notation	Description
a	Positive document frequency, i.e., number of documents in positive category containing term t_i.
b	Number of documents in positive category which do not contain term t_i.
c	Negative document frequency, i.e., number of documents in negative category containing term t_i.
d	Number of documents in negative category which do not contain term t_i.
N	Total number of documents in document collection, $N = a + b + c + d$.
N^+, N^-	N^+ is number of documents in positive category, and N^- is number of documents in negative category. $N^+ = a+b$, $N^- = c+d$.

Table 2: Notations used to formulate global term weighting schemes.

Global weight	Notation	Description
$\log_2 \frac{N}{a+c}$	*idf*	Inverse document frequency (Jones, 1972)
$\log_2\left(\frac{N}{a+c}-1\right)$	*pidf*	Probabilistic *idf* (Wu and Salton, 1981)
$\log_2 \frac{b+d+0.5}{a+c+0.5}$	*bidf*	BM 25 *idf* (Jones et al., 2000)
$\frac{a}{N}\log_2 \frac{aN}{(a+b)(a+c)}+\frac{b}{N}\log_2 \frac{bN}{(a+b)(b+d)}+$ $\frac{c}{N}\log_2 \frac{cN}{(a+c)(c+d)}+\frac{d}{N}\log_2 \frac{dN}{(b+d)(c+d)}$	*ig*	Information gain
$\log_2\left(\max(\frac{aN}{(a+c)N^+},\frac{cN}{(a+c)N^-})\right)$	*mi*	Mutual information
$\log_2 \frac{N^-a}{N^+c}$	*didf*	Delta *idf* (Martineau and Finin, 2009)
$\log_2 \frac{N^-a+0.5}{N^+c+0.5}$	*dsidf*	Delta smoothed *idf* (Paltoglou and Thelwall, 2010)
$\log_2 \frac{N^-(a+0.5)}{N^+(c+0.5)}$	*dsidf'*	Another version of *dsidf*
$\log_2 \frac{(N^- -c+0.5)a+0.5}{(N^+ -a+0.5)c+0.5}$	*dbidf*	Delta BM25 *idf* (Paltoglou and Thelwall, 2010)
$\log_2 \frac{(N^- -c+0.5)(a+0.5)}{(N^+ -a+0.5)(c+0.5)}$	*dbidf'*	Another version of *dbidf*
$\log_2\left(2+\frac{a}{\max(1,c)}\right)$	*rf*	Relevance frequency (Lan et al., 2009)

Table 3: Global term weighting schemes.

on SA by Pang et al. (2002) showed that much better performance was achieved by using *tp*, not *tf*. This conclusion for SA was opposite to TTC, so *tp* was preferred in subsequent SA research.

2.2 Global Term Weighting Schemes

In contrast to local weight, global weight depends on the whole document collection. To formulate different global weighting schemes, some notations are first introduced in table 2. By using these notations, table 3 presents several representative global weighting schemes in IR, TTC and SA, including inverse document frequency (*idf*), probabilistic *idf* (*pidf*), BM25 *idf* (*bidf*), information gain (*ig*), delta *idf* (*didf*), *dsidf'*, delta BM25 *idf* (*dbidf*), *dbidf'* and relevance frequency (*rf*). Among these global weighting methods, *idf*, *pidf* and *bidf* are unsupervised methods because they do not utilize the category label information of document collection. The common idea behind them is that a term that occurs rarely is good at discriminating between documents.

Other global weighting schemes in table 3 are supervised term weighting methods. Among these supervised factors, feature selection methods, *ig* and *mi* are studied earliest. In TTC field, Debole and Sebastiani (2003) replaced *idf* with *ig* and other feature selection methods, *gr* and *chi*, for global term weighting. They concluded that these feature selection methods did not give a consistent superiority over the standard *idf*. In SA field, Deng et al. (2013) also employed several feature selection methods, including *ig* and *mi*, to learn the global weight of each term from training documents with category labels. The experimental results showed that compared with *bidf*, *mi* produced better accuracy on two of three datasets but *ig* provided very poor results.

For the rest of supervised term weighting schemes in table 3, *rf* is published in TTC literature, *didf* and *dbidf* are published in SA literature. The intuitive consideration of *rf* is that the more concentrated a high frequency term is in the positive category than in the negative category, the more contributions

it makes in selecting the positive samples from the negative samples. Driven by this intuition, *rf* was proposed to capture this basic idea. The experimental results showed that when combined with the local component *tf*, *rf* consistently and significantly outperformed other term weighting methods, including *idf* and *ig*. Due to the asymmetry of *rf*, it only boosts the weights of terms that appear more frequently in the positive category. In other words, *rf* discriminates against terms appearing more frequently in negative category. The asymmetry of *rf* is reasonable for TTC because it only cares whether a document belongs to a topic or not and a single document can concentrate on different topics. However, it is not the case for binary sentiment classification since terms appear in positive or negative reviews are of the same importance.

In SA field, The first published supervised term weighing scheme, introduced by Martineau and Finin (2009), is called delta *idf*. Instead of only using *tf* as term weights, the authors assigned term weights for a document by calculating the difference of that term's *idf* values in the positive and negative training documents. Obviously, *didf* boosts the importance of terms that are unevenly distributed between the positive and negative categories and discounts evenly distributed words. It is known that the distribution of sentimental words is more uneven than stop words, as a result, *didf* assign much greater weights to sentimental words than stop words. The produced results showed that *didf* provided higher classification accuracy than the simple *tf* or the binary weighting scheme *tp*. Nonetheless, *didf* is susceptible to the errors caused by the case that $a = 0$ or $c = 0$, and the authors did not provide any detail that how they deal with this problem. Following the idea of *didf* and to rectify the problem of *didf*, Paltoglou and Thelwall (2010) presented a smoothed version of *didf*, delta smoothed *idf* (*dsidf*), and explored other more sophisticated global term weighting methods originated from IR including BM25 *idf* (*bidf*) and delta BM25 *idf* (*dbidf*). The formulas of these schemes are also presented in table 3. They showed that these variants of the classic *tf-idf* scheme provided significant increases over the best term weighting methods for SA in terms of accuracy. The idea of introducing smoothness technique is wonderful and can indeed avoid the computational errors in *didf*, but due to the unsuitable implementation, the smoothed version of *didf* provided by Paltoglou and Thelwall (2010) severely encounters the problem of over-weighting. We provide another version of *dsidf*, namely *dsidf'*. Besides *dsidf*, over-weighting is also severely encountered by *dbidf*, and our versions of it is denoted as *dbidf'*.

3 Research Design

Based on our review of term weighting schemes above, we believe that supervised term weighting can, but not always, boost the performances of text categorization. Actually, the somewhat successful ones, such as *rf*, *didf* and *dsidf*, follow the same intuition that the more imbalanced a term's distribution is across different categories, the more contribution it makes in discriminating between the positive and negative documents. The only difference between them lies in the quantification of the imbalance of a term's distribution. However, existing methods more or less suffer from the problem of over-weighting. We argue that a successful supervised weighting method should satisfy the following two criteria and develop a robust framework of supervised term weighting schemes.

Criterion 1: Assign large weights to terms that unevenly distribute across different categories.
Criterion 2: Avoid the over-weighting problem.

3.1 Our Framework

Over-weighting is somewhat like over-fitting in statistical machine learning, so we name it over-weighting. It is known that over-fitting occurs when a statistical model describes random error or noise instead of the underlying relationship. Similarly, over-weighting could occur in supervised term weighting. In practice we indentify that over-weighting is caused by the presence of noisy terms and the unsuitable quantification of the degree of the imbalance of a term's distribution.

The presence of noisy terms would lead to the problem of over-weighting. To illustrate this phenomenon, suppose that the training document collection contains 10,000 documents and evenly distributes over the positive and negative category, the number of documents containing the strange term "leoni" belonging to positive category is 5, i.e., $a = 5$, and no document belonging to negative category contains "leoni", i.e., $c = 0$, according to the formulation of most existing supervised methods such as dsidf, the weight of "leoni" should be large since "leoni" unevenly distributes over different categories. However, since the total number of documents containing "leoni" is so trivial compared to the size of

training collection, "leoni" could be an unusual word. We call the terms like "leoni" singular terms. Statistically, singular terms account for a great part of the whole terms in the dictionary constructed based on the training documents even if we filter out low frequency words. As singular terms do not embody any sentiment and the weights of them are supposed to be small, we formulate the global weight of term t_i as

$$g_i = \begin{cases} 0, \text{if } a = 0 \,(\text{or } c = 0) \text{ and } (a+c)/N < \alpha \\ r, \text{otherwise} \end{cases} \tag{2}$$

where r is a variable quantifying the imbalance of a term's distribution across different categories and its value ranges from 0 to 1, α is a very small number, here we set α to 0.005. As formula (2) cuts down the weights of singular terms, we name the first regularization technique *singular term cutting*.

Also, an unsuitable quantification of a term's distribution would lead to unreasonably too large ratios between different weights and thus results in over-weighting, although the term weight calculated by (2) is no more than 1. This finding leads us to introduce the second regularization technique, bias term, to the weight of term t_i, so our framework of supervised term weighting schemes is modified as

$$g_i = \begin{cases} b_0, \text{if } a = 0 \,(\text{or } c = 0) \text{ and } (a+c)/N < \alpha \\ b_0 + r, \text{otherwise} \end{cases} \tag{3}$$

where b_0 is the bias term, it shrinks the ratios between different weights of terms, the value of it controls the trade-off between weighting the terms freely and preventing over-weighting. If b_0 is too large, supervised term weighting would make no difference and under-weighting would occur. If b_0 is too small, over-weighting would occur. The optimal value of b_0 can be obtained via cross-validation, a model selection technique widely used in machine learning.

3.2 Regularized Versions of Existing Methods

As mentioned before, the somewhat successful ones of existing supervised weighting methods try to quantify the imbalance of a term's distribution. Recall that in our framework, r is just right a variable sharing this purpose, so we can make improvement on existing supervised weighting methods by replacing r with them. Ahead of the improvement of existing methods, we first provide new insights into existing methods using the concepts of over-weighting and regularization.

Because r quantifies the degree of the imbalance of a term's distribution across different categories, existing methods are required to satisfy Criterion 1. It has been clear that *didf*, *dsidf*, *dsidf'*, *dbidf*, *dbidf'*, *mi* and *rf* satisfy Criterion 1 via the review of existing methods in section 2. Another property shared by them is that the formulations of them base on logarithmic function. It is known that logarithmic function plays the role of shrinking the ratios between different term weights, so they implicitly satisfy Criterion 2 and in some degree reduce the over-weighting problem. In actuality, *dsidf*, *dsidf'* and *rf* can be treated as the further regularized versions of *didf* since the constant 2+ in *rf* and the smoothness in *dsidf* and *dsidf'* can be treated as regularization techniques. We have pointed out in section 2 that due to the unreasonable implementation of smoothness, *dsidf* and *dbidf* do not reduce, but aggravate over-weighting. As to *dsidf'* and *dbidf'*, they limit over-weighting in a very great degree via the introduction of smoothness technique and logarithmic function, but over-weighting is still not overcome completely, experimental results in section 4 will show that the performances of them can be further enhanced by cutting the weights of singular terms and adding a bias term.

Method	Regularized version
didf *dsidf* *dsidf'* *rf*	$\begin{cases} b_0, \text{if } a = 0 \,(\text{or } c = 0) \text{ and } (a+c)/N < \alpha \\ b_0 + \dfrac{\log_2 \max(a,c)/\min(a,c)}{\max_t\{\log_2 \max(a,c)/\min(a,c)\}}, \text{otherwise} \end{cases}$
dbidf *dbidf'*	$\begin{cases} b_0, \text{if } a = 0 \,(\text{or } c = 0) \text{ and } (a+c)/N < \alpha \\ b_0 + \dfrac{\log_2 (N^- - \min(a,c))/\max(a,c)}{\max_t\{\log_2 (N^- - \max(a,c))/\min(a,c)\}}, \text{otherwise} \end{cases}$
mi	$\begin{cases} b_0, \text{if } a = 0 \,(\text{or } c = 0) \text{ and } (a+c)/N < \alpha \\ b_0 + \log_2 \dfrac{mi}{\max_t\{mi\}}, \text{otherwise} \end{cases}$

Table 4: Regularized versions of existing supervised term weighting schemes.

Up to present we have known that existing supervised methods encounter over-weighting in different degree. In order to make improvements on existing methods and under the guidance of our framework, we present the regularized versions of *didf*, *dsidf*, *dsidf'*, *dbidf*, *dbidf'* and *mi* in table 4. These methods are selected to improve due to their typical representations and diversities.
Note that the regularized versions of *didf*, *dsidf*, *dsidf'* and *rf* and are the same one due to the fact that *dsidf*, *dsidf'* and *rf* are same as *didf* if there is no smoothness or constant in them. For the same reason, *dbidf* and *dbidf'* are grouped together.

3.3 Regularized Entropy

Inspired by the derivation of our framework for supervised term weighting, we propose a novel supervised term weighting scheme called regularized entropy (*re*). For *re*, entropy is exploited to measure the degree of the imbalance of a term's distribution across different categories. According to information theory (Shannon, 1948), for a random variable X with m outcomes $\{x_1,\ldots, x_m\}$, the entropy, a measure of uncertainty and denoted by $H(X)$, is defined as

$$H(X) = -\sum_{i=1}^{m} p(x_i)\log_2 p(x_i), \tag{4}$$

where $p(x_i)$ is the probability that X equals to x_i. Let p^+ and p^- denote the probability of documents where term t_i occurs and belonging to positive and negative category respectively, then p^+ and p^- can be estimated as

$$p^+ \approx \frac{a}{a+c}, p^- \approx \frac{c}{a+c}. \tag{5}$$

According to formula (4), if term t_i occurs in a document, the degree of uncertainty of this document belonging to a category is

$$h = -p^+\log_2 p^+ - p^-\log_2 p^- = -\frac{a}{a+c}\log_2\frac{a}{a+c} - \frac{c}{a+c}\log_2\frac{c}{a+c}. \tag{6}$$

Obviously, if the documents containing term t_i distribute evenly over different categories, the entropy h will be large. In contrast, if the documents containing term t_i distribute unevenly over different categories, the entropy h will be relatively small. However, we hope that the more uneven the distribution of documents where term t_i occurs, the larger the weight of t_i is. And that the entropy h is between 0 and 1, so the original formula of the weight of term t_i is

$$g_i = 1 - h. \tag{7}$$

We call the scheme formulated by the (7) *nature entropy* (*ne*). It seems that *ne* can be used as the weights of terms directly and will perform well. Unfortunately, *ne* suffers from the same problem with existing methods. Under the guidance of our framework, the regularized version of *ne* is formulated as

$$g_i = \begin{cases} b_0, \text{if } a = 0\ (\text{or } c = 0) \text{ and } (a+c)/N < \alpha \\ b_0 + (1-h), \text{otherwise} \end{cases}. \tag{8}$$

We name the proposed method formulated by (8) *regularized entropy* (*re*), which literally indicates the idea behind the scheme.

4 Experimental Results

We conduct sentiment classification experiments on three document-level datasets. The first one is Cornell movie review dataset introduced by Pang and Lee (2004). This sentiment polarity dataset consists of 1,000 positive and 1,000 negative movie reviews. The second dataset is taken from Multi-Domain Sentiment Dataset (MDSD) of product reviews (Blitzer et al., 2007). MDSD is initially released for the research on sentiment domain adaption but can also be used for sentiment polarity classification. It contains Amazon product reviews for different product types, we select camera reviews and thus refer the second corpus as Amazon camera review dataset. Also, it consists of 1,000 positive and 1,000 negative camera reviews.

For the above two datasets, the results are based on the standard 10-fold cross validation. Term weighting is performed on the 1,800 training reviews for each fold and the remaining 200 are used to evaluate the predicting accuracy. The overall classification accuracy is the average accuracy across 10 folds.

We also use the Stanford large movie review dataset developed by Mass et al. (2011). It contains 50,000 movie reviews, split equally into 25,000 training and 25,000 testing set. For this dataset, due to the original

Cornell movie review				Amazon camera review				Stanford movie review			
	tf	*tp*	*atf*		*tf*	*tp*	*atf*		*tf*	*tp*	*atf*
no	85.20	88.05	88.15	*no*	86.80	87.25	87.50	*no*	88.38	88.72	88.71
idf	84.15	84.90	85.10	*idf*	85.70	85.75	86.10	*idf*	88.30	88.24	88.26
ig	86.40	87.65	87.90	*ig*	87.25	87.85	87.65	*ig*	88.71	88.40	88.45
mi	86.90	88.85	88.85	*mi*	88.95	89.05	89.15	*mi*	89.23	89.45	89.52
dsidf	80.25	80.20	80.10	*dsidf*	83.15	82.80	83.30	*dsidf*	86.72	86.89	86.77
dsidf'	86.65	88.20	88.15	*dsidf'*	88.20	88.95	89.10	*dsidf'*	89.23	89.25	89.32
dbidf	81.20	81.10	81.10	*dbidf*	86.60	87.00	86.90	*dbidf*	86.80	86.73	86.78
dbidf'	87.30	88.30	88.40	*dbidf'*	88.85	89.10	89.00	*dbidf'*	89.41	89.39	89.52
rf	85.10	88.00	87.75	*rf*	86.95	87.35	87.85	*rf*	87.84	88.36	88.46
re	87.85	89.60	89.65	*re*	89.15	89.45	89.50	*re*	89.53	89.81	89.80

Table 5: Classification accuracy of local and global weighting methods.

split, no cross validation is used. Term weighting is only implemented on the training set, and the classification accuracy is reported based on the testing set.

We only use unigrams as the features. Support Vector Machine (SVM) is used as the classifier. Specially, we adopt the L2-regularized L2-loss linear SVM and the implementation software is LIBLINEAR (Fan et al., 2008). In all our experiments, cross-validation is performed on training document collection to obtain optimal value of b_0. On Cornell and Stanford movie review dataset, b_0 is set to 0.1 for *re*, 0.05 for the improved versions of *didf*, *dsidf*, *dsidf'* and *rf*, 0.02 for that of *mi*, and 0.01 for those of *dbidf* and *dbidf'*. On Amazon camera review dataset, b_0 is set to 0.05 for *re* 0.03 for the improved versions of *didf*, *dsidf*, *dsidf'* and *rf*, 0.02 for that of *mi*, and 0.01 for those of *dbidf* and *dbidf'*.

4.1 Experiment 1: Comparisons of *re* Against Existing Methods

Table 5 reports the classification accuracies of *re* and other term weighting schemes. On the Cornell movie review dataset, the local weighting method *tp* outperforms *tf* significantly in general except the case that *dbidf* and *dsidf* are used as the global weighting methods. There is no distinct difference between *tp* and *atf*, neither of them consistently performs better than each other when combined with various global weighting methods.

Compared to the change of local weighting methods, global weighting methods lead to more significant difference on classification accuracy. Combined with different local weighting schemes, the proposed global weighting method, *re*, has always been shown to clearly perform better than other global weighting methods. Specially, the highest classification accuracy, 89.65%, is achieved by the combination of *re* and *atf*, i.e., *atf-re*. Compared to *no*, *re* shows apparent superiorities, the increases of accuracy are +1.55% (from 88.05% to 89.60%) and +1.50% (from 88.15% to 89.65%) respectively when the local methods are *tp* and *atf*. The most popular *idf* in IR field is not a good choice for sentiment classification. For the methods originated from TTC field, the feature selection approaches, *mi* performs well and the classification accuracies produced by it is higher than the others except *re* in apparent advantages. Unlike *mi*, *ig* is instead a disappointing performer, the accuracy 87.65%, provided by *ig* when combined with *tp*, is far lower than that of *mi*, this observation is entirely predictable due to the fact that *ig* does not follow Criterion 1 and suffers over-weighting. As for *rf*, it do not perform well, the highest accuracy provided by them is only 88.00% respectively. It is not surprising that *rf* does not even outperform no since its discrimination against the terms that appear more frequently in the negative reviews. When it comes to the approaches that recently appeared in SA literature, both *dsidf* and *dbidf* performs very poorly because of over-weighting problem caused by the unreasonable implementation. But both *dsidf'* and *dbidf'* are shown to give slightly better results than *no*.

On the Amazon camera review dataset, the performances of local weighting methods agree with those on Cornell movie review dataset. Again, *tp* and *atf* yield comparable classification accuracy and both of them outperform *tf*. The performances on this dataset produced by global weighting methods are, generally, in accordance to those on the previous dataset, but some differences deserve our attention. First, *re* outperforms *no* with greater superiorities compared to the previous dataset, the increase of accuracy is +2.20% (from 87.25% to 89.45%) and +2.00% (from 87.50% to 89.50%) respectively when the local methods are *tp* and *atf* . Another one is that *dsidf'* provides more apparent advantages over *no* compared to the previous dataset but differences between *re* and *dsidf'* become smaller.

Cornell movie review				Amazon camera review				Stanford movie review			
Method	Original version	Regularized version	Difference to original version	Method	Original version	Regularized version	Difference to original version	Method	Original version	Regularized version	Difference to original version
didf	N/A	89.50	N/A	*didf*	N/A	89.60	N/A	*didf*	N/A	89.71	N/A
dsidf	80.20	89.50	+9.30	*dsidf*	82.80	89.60	+6.80	*dsidf*	86.89	89.71	+2.82
dsidf'	88.20	89.50	+1.30	*dsidf'*	88.95	89.60	+0.65	*dsidf'*	89.25	89.71	+0.46
rf	88.00	89.50	+1.50	*rf*	87.35	89.60	+2.25	*rf*	88.36	89.71	+1.35
dbidf	81.10	89.25	+8.15	*dbidf*	87.00	89.65	+2.65	*dbidf*	86.83	89.49	+2.66
dbidf'	88.30	89.25	+0.95	*dbidf'*	89.10	89.65	+0.55	*dbidf'*	89.39	89.49	+0.10
mi	88.85	89.10	+0.25	*mi*	89.05	89.55	+0.50	*mi*	89.45	89.59	+0.14
ne	83.45	89.60	+6.15	*ne*	87.85	89.45	+1.60	*ne*	87.32	89.81	+2.49

Table 6: Classification accuracies of original versions of *ne* and some existing supervised term weighting schemes and their regularized versions under our framework.

On the Stanford large movie review dataset, differences in accuracy are smaller than those on the previous ones, but the testing set contains 25,000 documents, the variance of the performance estimate is quite low (Maas et al., 2011). Interestingly, unlike the conclusion on the Cornell movie review dataset, *tp* does not show significant advantages over *tf* and even slightly underperforms *tf* when the global methods are *idf*, *ig*, *dbidf*, and *dbidf'*. The performances of *tp* and *atf* are still comparable but *atf* reveals a slight superiority over *tp*. In spite of the smaller differences, among the global weighting methods, *re* still embraces the highest classification accuracy, 89.81%, when combined with *tp*. In accordance to the observations on the previous two datasets, *mi*, *dsidf'* and *dbidf'* yield higher classification accuracies than *no*. Again, the other global methods, *idf*, *ig*, *rf*, *dsidf* and *dbidf* still produce comparable or lower accuracies in comparison with *no*.

4.2 Experiment 2: Comparisons Existing Methods Against Their Regularized Versions

We also compare the performances of some representative supervised methods, i.e., *didf*, *dsidf*, *dsidf'*, *dbidf*, *dbidf'*, *rf*, and *mi* against their regularized versions. In this experiment, we only use *tp* as the local weighting method. Table 6 records the classification accuracies of original versions of these methods and their improved versions. We can observe that the regularized versions of existing methods consistently have much better accuracy. Regularized version of *dsidf* yields the most significant improvements, the accuracy difference to original version is +9.30%, +6.80% and +2.82% on three datasets respectively. The accuracy difference between *dbidf* and its regularized version is also remarkable and significant. These observations validate our analysis in section 2 that *dsidf* and *dbidf* severely encounters over-weighting problem. Note that the improvements of the regularized versions of *dsidf'*, *dbidf'* and *mi* over their originals are trivial as they are much less subjected to over-weighting. Significance test will be included for these methods to test if the improvements are statistically reliable.

5 Conclusion and Future Work

In this study we have proposed a robust framework of supervised term weighting schemes. This framework is developed based on the techniques introduced to reduce over-weighting problem commonly suffered by existing supervised weighting methods.

Over-weighting is a new concept proposed in this paper, which is caused by the presence of many noisy words and the unreasonably too large ratios between weights of different terms. To reduce over-weighting, we have introduced two regularization techniques, singular term cutting and bias term. Singular term cutting cuts down the weights of noisy or strange words, and bias term shrinks the ratios between weights of different terms. Comparative experiments have shown that regularization techniques significantly enhance the performances of existing supervised methods.

More over, a novel supervised term weighting scheme, *re*, is proposed under our framework. The formulation of *re* bases on entropy, which is used to measure a term's distribution across different categories. The experimental results have shown that *re* not only outperforms its original version, *ne*, with great advantage but also consistently outperforms existing methods appearing in IR, TTC and SA. In the future, we would like to extend our work to other tasks such as multi-class classification and traditional text categorization.

Acknowledgements

This work was supported in part by National Natural Science Foundation of China under grant 61371148 and Shanghai National Natural Science Foundation under grant 12ZR1402500.

References

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of ACL*, Pages 142-150.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan and Claypool Publishers.

Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts, In *Proceedings of ACL*, pages 271-278.

Bo Pang, Lillian Lee and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification Using Machine Learning Techniques, In *Proceedings of EMNLP*, pages 79-86.

Claude E. Shannon. 1948. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27 (3):379-423.

Franca Debole and Fabrizio Sebastiani. 2003. Supervised Term Weighting for Automated Text Categorization. In *Proceedings of ACM Symposium on Applied Computing*, pages 784-788.

Georgios Paltoglou and Mike Thelwall. 2010. A Study of Information Retrieval Weighting Schemes for Sentiment Analysis. In *Proceedings of ACL*, pages 1386-1395.

Gerard Salton and Christopher Buckley. 1988. Term Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, 24(5):513-523.

Gerard Salton and Michael J. McGill. 1983. Introduction to Modern Information Retrieval. *McGraw Hill Book Inc.*, New York.

Harry Wu and Gerard Salton. 1981. A Comparison of Search Term Weighting: Term Relevance vs. Inverse Document Frequency. In *Proceeding of ACM SIGIR*, pages 30-39.

John Blitzer, Mark Dredze and Fernando Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of ACL*, pages 440-447.

Justin Martineau and Tim Finin. 2009. Delta TFIDF: An Improved Feature Space for Sentiment Analysis. In *Proceedings of Third AAAI International Conference on Weblogs and Social Media*, pages 258-261.

Karen S. Jones, Stephen Walker and Stephen E. Robertson. 2000. A Probabilistic Model of Information Retrieval: Development and Comparative Experiments. *Information Processing and Management*, 36(6):779-808.

Man Lan, Chew L. Tan, Jian Su and Yue Lu. 2009. Supervised and Traditional Term Weighting Methods for Automatic Text Categorization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(4):721-735.

Mingqing Hu and Bing Liu. 2004. Mining and Summarizing Customer Reviews. In *Proceedings of ACM SIGKDD*, pages 168-177.

Rong E. Fan, Kai W. Chang, Cho J. Hsieh, Xiang R. Wang, and Chih J. Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871-1874.

Sparck K. Jones. 1972. A Statistical Interpretation of Term Specificity and Its Application in Retrieval. *Journal of Documentation*, 28:11-21.

William B. Croft. 1983. Experiments with Representation in A Document Retrieval System. *Information Technology: Research and Development*, 2:1-21.

Zhi H. Deng, Kun H. Luo and Hong L. Yu. 2013. A Study of Supervised Term Weighting Scheme for Sentiment Analysis. *Expert Systems with Applications*, 41(7):3506-3513.

Sentiment Classification with Graph Co-Regularization

Guangyou Zhou, Jun Zhao, and Daojian Zeng
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing 100190, China
{gyzhou,jzhao,djzeng}@nlpr.ia.ac.cn

Abstract

Sentiment classification aims to automatically predict sentiment polarity (e.g., positive or negative) of user-generated sentiment data (e.g., reviews, blogs). To obtain sentiment classification with high accuracy, supervised techniques require a large amount of manually labeled data. The labeling work can be time-consuming and expensive, which makes unsupervised (or semi-supervised) sentiment analysis essential for this application. In this paper, we propose a novel algorithm, called graph co-regularized non-negative matrix tri-factorization (GNMTF), from the geometric perspective. GNMTF assumes that if two words (or documents) are sufficiently close to each other, they tend to share the same sentiment polarity. To achieve this, we encode the geometric information by constructing the nearest neighbor graphs, in conjunction with a non-negative matrix tri-factorization framework. We derive an efficient algorithm for learning the factorization, analyze its complexity, and provide proof of convergence. Our empirical study on two open data sets validates that GNMTF can consistently improve the sentiment classification accuracy in comparison to the state-of-the-art methods.

1 Introduction

Recently, sentiment classification has gained a wide interest in natural language processing (NLP) community. Methods for automatically classifying sentiments expressed in products and movie reviews can roughly be divided into supervised and unsupervised (or semi-supervised) sentiment analysis. Supervised techniques have been proved promising and widely used in sentiment classification (Pang et al., 2002; Pang and Lee, 2008; Liu, 2012). However, the performance of these methods relies on manually labeled training data. In some cases, the labeling work may be time-consuming and expensive. This motivates the problem of learning robust sentiment classification via unsupervised (or semi-supervised) paradigm.

A traditional way to perform unsupervised sentiment analysis is the lexicon-based method (Turney, 2002; Taboada et al., 2011). Lexicon-based methods employ a sentiment lexicon to determine overall sentiment orientation of a document. However, it is difficult to define a universally optimal sentiment lexicon to cover all words from different domains (Lu et al., 2011a). Besides, most semi-automated lexicon-based methods yield unsatisfactory lexicons, with either high coverage and low precision or vice versa (Ng et al., 2006). Thus it is challenging for lexicon-based methods to accurately identify the overall sentiment polarity of users generated sentiment data. Recently, Li et al. (2009) proposed a constrained non-negative matrix tri-factorization (CNMTF) approach to sentiment classification, with a domain-independent sentiment lexicon as prior knowledge. Experimental results show that CNMTF achieves state-of-the-art performance.

From the geometric perspective, the data points (words or documents) may be sampled from a distribution supported by a low-dimensional manifold embedded in a high-dimensional space (Cai et al., 2011). This geometric structure, meaning that two words (or documents) sufficiently close to each other tend to share the same sentiment polarity, should be preserved during the matrix factorization. Research studies

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1331–1340, Dublin, Ireland, August 23-29 2014.

have shown that learning performance can be significantly enhanced in many real applications (e.g., text mining, computer vision, etc.) if the geometric structure is exploited (Roweis and Saul, 2000; Tenenbaum et al., 2000). However, CNMTF fails to exploit the geometric structure, it is not clear whether this geometric information is useful for sentiment classification, which remains an under-explored area. This paper is thus designed to fill the gap.

In this paper, we propose a novel algorithm, called graph co-regularized non-negative matrix tri-factorization (GNMTF). We construct two affinity graphs to encode the geometric information underlying the word space and the document space, respectively. Intuitively, if two words or documents are sufficiently close to each other, they tend to share the same sentiment polarity. Taking these two graphs as co-regularization for the non-negative matrix tri-factorization, leading to the better sentiment polarity prediction which respects to the geometric structures of the word space and document space. We also derive an efficient algorithm for learning the tri-factorization, analyze its complexity, and provide proof of convergence. Empirical study on two open data sets shows encouraging results of the proposed method in comparison to state-of-the-art methods.

The remainder of this paper is organized as follows. Section 2 introduces the basic concept of matrix tri-factorization. Section 3 describes our graph co-regularized non-negative matrix tri-factorization (GNMTF) for sentiment classification. Section 4 presents the experimental results. Section 5 introduces the related work. In section 6, we conclude the paper and discuss future research directions.

2 Preliminaries

2.1 Non-negative Matrix Tri-factorization

Li et al. (2009) proposed a matrix factorization based framework for unsupervised (or semi-supervised) sentiment analysis. The proposed framework is built on the orthogonal non-negative matrix tri-factorization (NMTF) (Ding et al., 2006). In these models, a term-document matrix $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_n] \in \mathbb{R}^{m \times n}$ is approximated by three factor matrices that specify cluster labels for words and documents by solving the following optimization problem:

$$\min_{\mathbf{U},\mathbf{H},\mathbf{V} \geq 0} \mathcal{O} = \left\|\mathbf{X} - \mathbf{U}\mathbf{H}\mathbf{V}^T\right\|_F^2 + \sigma_1 \left\|\mathbf{U}^T\mathbf{U} - \mathbf{I}\right\|_F^2 + \sigma_2 \left\|\mathbf{V}^T\mathbf{V} - \mathbf{I}\right\|_F^2 \tag{1}$$

where σ_1 and σ_2 are the shrinkage regularization parameters, $\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_k] \in \mathbb{R}_+^{m \times k}$ is the word-sentiment matrix, $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_n] \in \mathbb{R}_+^{n \times k}$ is the document-sentiment matrix, and k is the number of sentiment classes for documents. Our task is polarity sentiment classification (positive or negative), i.e., $k = 2$. For example, $\mathbf{V}_{i1} = 1$ (or $\mathbf{U}_{i1} = 1$) represents that the sentiment polarity of document i (or word i) is positive, and $\mathbf{V}_{i2} = 1$ (or $\mathbf{U}_{i2} = 1$) represents that the sentiment polarity of document i (or word i) is negative. $\mathbf{V}_{i*} = 0$ (or $\mathbf{U}_{i*} = 0$) represents unknown, i.e., the document i (or word i) is neither positive or negative. $\mathbf{H} \in \mathbb{R}_+^{k \times k}$ provides a condensed view of $\mathbf{X}$; $\|\cdot\|_F$ is the Frobenius norm and $\mathbf{I}$ is a $k \times k$ identity matrix with all entries equal to 1. Based on the shrinkage methodology, we can approximately satisfy the orthogonality constraints for $\mathbf{U}$ and $\mathbf{V}$ by preventing the second and third terms from getting too large.

2.2 Constrained NMTF

Lexical knowledge in the form of the polarity of words in the lexicon can be introduced in matrix tri-factorization. By partially specifying word polarity via $\mathbf{U}$, the lexicon influences the sentiment prediction $\mathbf{V}$ over documents. Following the literature (Li et al., 2009), let $\mathbf{U}_0$ represent lexical prior knowledge about sentiment words in the lexicon, e.g., if word i is positive $(\mathbf{U}_0)_{i1} = 1$ while if it is negative $(\mathbf{U}_0)_{i2} = 1$, and if it does not exist in the lexicon $(\mathbf{U}_0)_{i*} = 0$. Li et al. (2009) also investigated that we had a few documents manually labeled for the purpose of capturing some domain-specific connotations. Let $\mathbf{V}_0$ denote the manually labeled documents, if the document expresses positive sentiment $(\mathbf{V}_0)_{ii} = 1$, and $(\mathbf{V}_0)_{i2} = 1$ for negative sentiment. Therefore, the semi-supervised learning with lexical knowledge can be written as:

$$\min_{\mathbf{U},\mathbf{H},\mathbf{V} \geq 0} \mathcal{O} + \alpha \mathrm{Tr}\big[(\mathbf{U} - \mathbf{U}_0)^T \mathbf{C}^u (\mathbf{U} - \mathbf{U}_0)\big] + \beta \mathrm{Tr}\big[(\mathbf{V} - \mathbf{V}_0)^T \mathbf{C}^v (\mathbf{V} - \mathbf{V}_0)\big] \tag{2}$$

where $\mathrm{Tr}(\cdot)$ denotes the trace of a matrix, $\alpha > 0$ and $\beta > 0$ are the parameters which control the contribution of lexical prior knowledge and manually labeled documents. $\mathbf{C}^u \in \{0,1\}^{m\times m}$ is a diagonal matrix whose entry $\mathbf{C}^u_{ii} = 1$ if the category of the i-th word is known and $\mathbf{C}^u_{ii} = 0$ otherwise. $\mathbf{C}^v \in \{0,1\}^{n\times n}$ is a diagonal matrix whose entry $\mathbf{C}^v_{ii} = 1$ if the category of the i-th document is labeled and $\mathbf{C}^v_{ii} = 0$ otherwise.

3 Graph Co-regularized Non-negative Matrix Tri-factorization

In this section, we introduce our proposed graph co-regularized non-negative matrix tri-factorization (GNMTF) algorithm which avoids this limitation by incorporating the geometrically based co-regularization.

3.1 Model Formulation

Based on the manifold assumption (Belkin and Niyogi, 2001), if two documents $\mathbf{x}_i$ and $\mathbf{x}_j$ are sufficiently close to each other in the intrinsic geometric of the documents distribution, then their sentiment polarity $\mathbf{v}_i$ and $\mathbf{v}_j$ should be close. In order to model the geometric structure, we construct a document-document graph G^v. In the graph, nodes represent documents in the corpus and edges represent the affinity between the documents. The affinity matrix $\mathbf{W}^v \in \mathbb{R}^{n\times n}$ of the graph G^v is defined as

$$\mathbf{W}^v_{ij} = \begin{cases} \cos(\mathbf{x}_i, \mathbf{x}_j) & \text{if } \mathbf{x}_i \in \mathcal{N}_p(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \mathcal{N}_p(\mathbf{x}_i) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $\mathcal{N}_p(\mathbf{x}_i)$ represents the p-nearest neighbors of document $\mathbf{x}_i$. Many matrices, e.g., 0-1 weighting, textual similarity and heat kernel weighting (Belkin and Niyogi, 2001), can be used to obtain nearest neighbors of a document, and further define the affinity matrix. Since $\mathbf{W}^v_{ij}$ in our paper is only for measuring the closeness, we only use the simple textual similarity and do not treat the different weighting schemes separately due to the limited space. For further information, please refer to (Cai et al., 2011).

Preserving the geometric structure in the document space is reduced to minimizing the following loss function:

$$\begin{aligned} \mathcal{R}^v &= \frac{1}{2}\sum_{i,j=1}^{n} \left\|\mathbf{v}_i - \mathbf{v}_j\right\|_2^2 \mathbf{W}^v_{ij} = \sum_{i=1}^{n} \mathbf{v}_i^T \mathbf{v}_i \mathbf{D}^v_{ii} - \sum_{i,j=1}^{n} \mathbf{v}_i^T \mathbf{v}_j \mathbf{W}^v_{ij} \\ &= \mathrm{Tr}(\mathbf{V}^T \mathbf{D}^v \mathbf{V}) - \mathrm{Tr}(\mathbf{V}^T \mathbf{W}^v \mathbf{V}) = \mathrm{Tr}(\mathbf{V}^T \mathbf{L}^v \mathbf{V}) \end{aligned} \tag{4}$$

where $\mathbf{D}^v \in \mathbb{R}^{n\times n}$ is a diagonal matrix whose entries are column (or row, since $\mathbf{D}^v$ is symmetric) sums of $\mathbf{W}^v$, $\mathbf{D}^v_{ii} = \sum_{j=1}^{n} \mathbf{W}^v_{ij}$, and $\mathbf{L}^v = \mathbf{D}^v - \mathbf{W}^v$ is the Laplacian matrix (Chung, 1997) of the constructed graph G^v.

Similarly to document-document geometric structure, if two words $\mathbf{w}_i = [\mathbf{x}_{i1}, \cdots, \mathbf{x}_{in}]$ and $\mathbf{w}_j = [\mathbf{x}_{j1}, \cdots, \mathbf{x}_{jn}]$ are sufficiently close to each other in the intrinsic geometric of the words distribution, then their sentiment polarity $\mathbf{u}_i$ and $\mathbf{u}_j$ should be close. In order to model the geometric structure in the word space, we construct a word-word graph G^u. In the graph, nodes represent distinct words and edges represent the affinity between words. The affinity matrix $\mathbf{W}^u \in \mathbb{R}^{m\times m}$ of the graph G^u is defined as

$$\mathbf{W}^u_{ij} = \begin{cases} \cos(\mathbf{w}_i, \mathbf{w}_j) & \text{if } \mathbf{w}_i \in \mathcal{N}_p(\mathbf{w}_j) \text{ or } \mathbf{w}_j \in \mathcal{N}_p(\mathbf{w}_i) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $\mathcal{N}_p(\mathbf{w}_j)$ represents the p-nearest neighbor of word $\mathbf{w}_j$. Here, we represent a term $\mathbf{w}_j$ as a document vector $[\mathbf{x}_{j1}, \cdots, \mathbf{x}_{jn}]$. To measure the closeness of two words, a common way is to calculate the similarity of their vector representations. Although there are several ways (e.g., co-occurrence information, semantic similarity computed by WordNet, Wikipedia, or search engine have been empirically studied in NLP literature (Hu et al., 2009)) to define the affinity matrix $\mathbf{W}^u$, we do not treat the different ways separately and leave this investigation for future work.

Preserving the geometric structure in the word space is reduced to minimizing the following loss function:

$$\mathcal{R}^u = \frac{1}{2}\sum_{i,j=1}^{m} \left\|\mathbf{u}_i - \mathbf{u}_j\right\|_2^2 \mathbf{W}^u_{ij} = \mathrm{Tr}(\mathbf{U}^T \mathbf{L}^u \mathbf{U}) \tag{6}$$

where $\mathbf{L}^u = \mathbf{D}^u - \mathbf{W}^u$ is the Laplacian matrix of the constructed graph G^u, and $\mathbf{D}^u \in \mathbb{R}^{m\times m}$ is a diagonal matrix whose entries are $\mathbf{D}^u_{ii} = \sum_{j=1}^{m} \mathbf{W}^u_{ij}$.

Finally, we treat unsupervised (or semi-supervised) sentiment classification as a clustering problem, employing lexical prior knowledge and partial manually labeled data to guide the learning process. Moreover, we introduce the geometric structures from both document and word sides as co-regularization. Therefore, our proposed unsupervised (or semi-supervised) sentiment classification framework can be mathematically formulated as solving the following optimization problem:

$$\begin{aligned}\min_{\mathbf{U},\mathbf{H},\mathbf{V}\geq 0} \mathcal{L} = &\left\|\mathbf{X} - \mathbf{U}\mathbf{H}\mathbf{V}^T\right\|_F^2 + \sigma_1\left\|\mathbf{U}^T\mathbf{U} - \mathbf{I}\right\|_F^2 + \sigma_2\left\|\mathbf{V}^T\mathbf{V} - \mathbf{I}\right\|_F^2 \\ &+ \alpha\mathrm{Tr}\big[(\mathbf{U}-\mathbf{U}_0)^T\mathbf{C}^u(\mathbf{U}-\mathbf{U}_0)\big] + \gamma\mathrm{Tr}(\mathbf{U}^T\mathbf{L}^u\mathbf{U}) \\ &+ \beta\mathrm{Tr}\big[(\mathbf{V}-\mathbf{V}_0)^T\mathbf{C}^v(\mathbf{V}-\mathbf{V}_0)\big] + \delta\mathrm{Tr}(\mathbf{V}^T\mathbf{L}^v\mathbf{V})\end{aligned} \tag{7}$$

where $\delta > 0$ and $\gamma > 0$ are parameters which control the contributions of document space and word space geometric information, respectively. With the optimization results, the sentiment polarity of a new document $\mathbf{x}_i$ can be easily inferred by $f(\mathbf{x}_i) = \arg\max_{j\in\{p,\, n\}} \mathbf{V}_{ij}$.

3.2 Learning Algorithm

We present the solution to the GNMTF optimization problem in equation (7) as the following theorem. The theoretical aspects of the optimization are presented in the next subsection.

Theorem 3.1. *Updating* $\mathbf{U}$*,* $\mathbf{H}$ *and* $\mathbf{V}$ *using equations (8)∼(10) will monotonically decrease the objective function in equation (7) until convergence.*

$$\mathbf{U} \leftarrow \mathbf{U} \circ \frac{\left[\mathbf{X}\mathbf{V}\mathbf{H}^T + \sigma_1\mathbf{U} + \alpha\mathbf{C}^u\mathbf{U}_0 + \gamma\mathbf{W}^u\mathbf{U}\right]}{\left[\mathbf{U}\mathbf{H}\mathbf{V}^T\mathbf{V}\mathbf{H}^T + \sigma_1\mathbf{U}\mathbf{U}^T\mathbf{U} + \alpha\mathbf{C}^u\mathbf{U} + \gamma\mathbf{D}^u\mathbf{U}\right]} \tag{8}$$

$$\mathbf{H} \leftarrow \mathbf{H} \circ \frac{\left[\mathbf{U}^T\mathbf{X}\mathbf{V}\right]}{\left[\mathbf{U}^T\mathbf{U}\mathbf{H}\mathbf{V}^T\mathbf{V}\right]} \tag{9}$$

$$\mathbf{V} \leftarrow \mathbf{V} \circ \frac{\left[\mathbf{X}^T\mathbf{U}\mathbf{H} + \sigma_2\mathbf{V} + \beta\mathbf{C}^v\mathbf{V}_0 + \delta\mathbf{W}^v\mathbf{V}\right]}{\left[\mathbf{V}\mathbf{H}^T\mathbf{U}^T\mathbf{U}\mathbf{H} + \sigma_2\mathbf{V}\mathbf{V}^T\mathbf{V} + \beta\mathbf{C}^v\mathbf{V} + \delta\mathbf{D}^v\mathbf{V}\right]} \tag{10}$$

where operator $\circ$ *is element-wise product and* $\frac{[\cdot]}{[\cdot]}$ *is element-wise division.*

Based on Theorem 3.1, we note that the multiplicative update rules given by equations (8)∼(10) are obtained by extending the updates of standard NMTF (Ding et al., 2006). A number of techniques can be used here to optimize the objective function in equation (7), such as alternating least squares (Kim and Park, 2008), the active set method (Kim and Park, 2008), and the projected gradients approach (Lin, 2007). Nonetheless, the multiplicative updates derived in this paper has reasonably fast convergence behavior as shown empirically in the experiments.

3.3 Theoretical Analysis

In this subsection, we give the theoretical analysis of the optimization, convergence and computational complexity. Without loss of generality, we only show the optimization of $\mathbf{U}$ and formulate the Lagrange function with constraints as follows:

$$\mathcal{L}(\mathbf{U}) = \left\|\mathbf{X} - \mathbf{U}\mathbf{H}\mathbf{V}^T\right\|_F^2 + \sigma_1\left\|\mathbf{U}^T\mathbf{U} - \mathbf{I}\right\|_F^2 + \alpha\mathrm{Tr}\big[(\mathbf{U}-\mathbf{U}_0)^T\mathbf{C}^u(\mathbf{U}-\mathbf{U}_0)\big] + \mathrm{Tr}(\Psi\mathbf{U}^T) \tag{11}$$

where Ψ is the Lagrange multiplier for the nonnegative constraint $\mathbf{U} \geq \mathbf{0}$.

The partial derivative of $\mathcal{L}(\mathbf{U})$ w.r.t. $\mathbf{U}$ is

$$\begin{aligned}\bigtriangledown_{\mathbf{U}}\mathcal{L}(\mathbf{U}) = &-2\mathbf{X}\mathbf{V}\mathbf{H}^T + 2\mathbf{U}\mathbf{H}\mathbf{V}^T\mathbf{V}\mathbf{H}^T + 2\sigma_1\mathbf{U}\mathbf{U}^T\mathbf{U} - 2\sigma_1\mathbf{U} \\ &+ 2\alpha\mathbf{C}^u\mathbf{U} - 2\alpha\mathbf{C}^u\mathbf{U}_0 + 2\gamma\mathbf{D}^u\mathbf{U} - 2\gamma\mathbf{W}^u\mathbf{U} + \Psi\end{aligned}$$

Using the Karush-Kuhn-Tucker (KKT) (Boyd and Vandenberghe, 2004) condition $\Psi \circ \mathbf{U} = \mathbf{0}$, we can obtain

$$\begin{aligned}\nabla_{\mathbf{U}}\mathcal{L}(\mathbf{U}) \circ \mathbf{U} = &\left[\mathbf{U}\mathbf{H}\mathbf{V}^T\mathbf{V}\mathbf{H}^T + \sigma_1\mathbf{U}\mathbf{U}^T\mathbf{U} + \alpha\mathbf{C}^u\mathbf{U} + \gamma\mathbf{D}^u\mathbf{U}\right] \circ \mathbf{U} \\ &- \left[\mathbf{X}\mathbf{V}\mathbf{H}^T + \sigma_1\mathbf{U} + \alpha\mathbf{C}^u\mathbf{U}_0 + \gamma\mathbf{W}^u\mathbf{U}\right] \circ \mathbf{U} = \mathbf{0}\end{aligned}$$

This leads to the update rule in equation (8). Following the similar derivations as shown above, we can obtain the updating rules for all the other variables $\mathbf{H}$ and $\mathbf{V}$ in GNMTF optimization, as shown in equations (9) and (10).

3.3.1 Convergence Analysis

In this subsection, we prove the convergence of multiplicative updates given by equations (8)$\sim$(10). We first introduce the definition of auxiliary function as follows.

Definition 3.1. *$\mathcal{F}(\mathbf{Y}, \mathbf{Y}')$ is an auxiliary function for $\mathcal{L}(\mathbf{Y})$ if $\mathcal{L}(\mathbf{Y}) \leq \mathcal{F}(\mathbf{Y}, \mathbf{Y}')$ and equality holds if and only if $\mathcal{L}(\mathbf{Y}) = \mathcal{F}(\mathbf{Y}, \mathbf{Y})$.*

Lemma 3.1. *(Lee and Seung, 2001) If $\mathcal{F}$ is an auxiliary function for $\mathcal{L}$, $\mathcal{L}$ is non-increasing under the update $\mathbf{Y}^{(t+1)} = \arg\min_{\mathbf{Y}} \mathcal{F}(\mathbf{Y}, \mathbf{Y}^{(t)})$*

Proof. By Definition 3.1, $\mathcal{L}(\mathbf{Y}^{(t+1)}) \leq \mathcal{F}(\mathbf{Y}^{(t+1)}, \mathbf{Y}^{(t)}) \leq \mathcal{F}(\mathbf{Y}^{(t)}, \mathbf{Y}^{(t)}) = \mathcal{L}(\mathbf{Y}^{(t)})$ □

Theorem 3.2. *Let function*

$$\begin{aligned}\mathcal{F}(\mathbf{U}_{ij}, \mathbf{U}_{ij}^{(t)}) = &\mathcal{L}(\mathbf{U}_{ij}^{(t)}) + \mathcal{L}'(\mathbf{U}_{ij}^{(t)})(\mathbf{U}_{ij} - \mathbf{U}_{ij}^{(t)}) \\ &+ \frac{\left[\mathbf{U}\mathbf{H}\mathbf{V}^T\mathbf{V}\mathbf{H}^T + \sigma_1\mathbf{U}\mathbf{U}^T\mathbf{U} + \alpha\mathbf{C}^u\mathbf{U} + \gamma\mathbf{D}^u\mathbf{U}\right]_{ij}}{\mathbf{U}_{ij}}(\mathbf{U}_{ij} - \mathbf{U}_{ij}^{(t)})\end{aligned} \tag{12}$$

be a proper auxiliary function for $\mathcal{L}(\mathbf{U}_{ij})$, where $\mathcal{L}'(\mathbf{U}_{ij}) = [\nabla_{\mathbf{U}}\mathcal{L}(\mathbf{U})]_{ij}$ is the first-order derivatives of $\mathcal{L}(\mathbf{U}_{ij})$ with respect to $\mathbf{U}_{ij}$.

Theorem 3.2 can be proved similarly to (Ding et al., 2006). Due to limited space, we omit the details of the validation. Based on Lemmas 3.1 and Theorem 3.2, the update rule for $\mathbf{U}$ can be obtained by minimizing $\mathcal{F}(\mathbf{U}_{ij}^{(t+1)}, \mathbf{U}_{ij}^{(t)})$. When setting $\nabla_{\mathbf{U}_{ij}^{(t+1)}}\mathcal{F}(\mathbf{U}_{ij}^{(t+1)}, \mathbf{U}_{ij}^{(t)})$, we can obtain

$$\mathbf{U}_{ij}^{(t+1)} = \mathbf{U}_{ij}^{(t)} \frac{\left[\mathbf{X}\mathbf{V}\mathbf{H}^T + \sigma_1\mathbf{U} + \alpha\mathbf{C}^u\mathbf{U}_0 + \gamma\mathbf{W}^u\mathbf{U}\right]_{ij}}{\left[\mathbf{U}\mathbf{H}\mathbf{V}^T\mathbf{V}\mathbf{H}^T + \sigma_1\mathbf{U}\mathbf{U}^T\mathbf{U} + \alpha\mathbf{C}^u\mathbf{U} + \gamma\mathbf{D}^u\mathbf{U}\right]_{ij}}$$

By Lemma 3.1 and Theorem 3.2, we have $\mathcal{L}(\mathbf{U}^{(0)}) = \mathcal{F}(\mathbf{U}^{(0)}, \mathbf{U}^{(0)}) \geq \mathcal{F}(\mathbf{U}^{(1)}, \mathbf{U}^{(0)}) \geq \mathcal{F}(\mathbf{U}^{(1)}, \mathbf{U}^{(1)}) = \mathcal{L}(\mathbf{U}^{(1)}) \geq \cdots \geq \mathcal{L}(\mathbf{U}^{(Iter)})$, where $Iter$ denotes the number of iteration number. Therefore, $\mathbf{U}$ is monotonically decreasing. Since the objective function $\mathcal{L}$ is lower bounded by 0, the correctness and convergence of Theorem 3.1 is validated.

3.3.2 Time Complexity Analysis

In this subsection, we discuss the time computational complexity of the proposed algorithm GNMTF. Besides expressing the complexity of the algorithm using big O notation, we also count the number of arithmetic operations to provide more details about running time. We show the results in Table 1, where $m \gg k$ and $n \gg k$.

Based on the updating rules summarized in Theorem 3.1, it it not hard to count the arithmetic operators of each iteration in GNMTF. It is important to note that $\mathbf{C}^u$ is a diagonal matrix, the nonzero elements on each row of $\mathbf{C}^u$ is 1. Thus, we only need zero addition and mk multiplications to compute $\mathbf{C}^u\mathbf{U}$. Similarly, for $\mathbf{C}^u\mathbf{U}_0$, $\mathbf{C}^v\mathbf{V}$, $\mathbf{C}^v\mathbf{V}_0$, $\mathbf{D}^u\mathbf{U}$ and $\mathbf{D}^v\mathbf{V}$, we also only need zero addition and mk multiplications for each of them. Besides, we also note that $\mathbf{W}^u$ is a sparse matrix, if we use a p-nearest neighbor graph, the average nonzero elements on each row of $\mathbf{W}^u$ is p. Thus, we only need mpk additions and mpk multiplications to compute $\mathbf{W}^u\mathbf{U}$. Similarly, for $\mathbf{W}^v\mathbf{V}$, we need the same operation counts as $\mathbf{W}^u\mathbf{U}$. Suppose the multiplicative updates stop after $Iter$ iterations, the time cost of multiplicative updates then becomes $O(Iter \times mnk)$. Therefore, the overall running time of GNMTF is similar to the standard NMTF and CNMTF.

	addition	multiplication	division	overall
GNMTF: $\mathbf{U}$	$2k^3 + (2m+n)k^2 + m(n+p)k$	$2k^3 + (2m+n)k^2 + m(n+p+7)k$	mk	$O(mnk)$
GNMTF: $\mathbf{H}$	$2k^3 + (m+n+2)k^2 + mnk$	$2k^3 + (m+n+1)k^2 + mnk$	k^2	$O(mnk)$
GNMTF: $\mathbf{V}$	$2k^3 + (2n+m)k^2 + n(m+p)k$	$2k^3 + (2n+m)k^2 + n(m+p+7)k$	nk	$O(mnk)$

Table 1: Computational operation counts for each iteration in GNMTF.

4 Experiments

4.1 Data Sets

Sentiment classification has been extensively studied in the literature. Among these, a large majority proposed experiments performed on the benchmarks made of Movies Reviews (Pang et al., 2002) and Amazon products (Blitzer et al., 2007).

Movies data This data set has been widely used for sentiment analysis in the literature (Pang et al., 2002), which consists of 1000 positive and 1000 negative reviews drawn from the IMDB archive of rec.arts.movies.reviews.newsgroups.

Amazon data This data set is heterogeneous, heavily unbalanced and large-scale, a smaller version has been released. The reduced data set contains 4 product types: Kitchen, Books, DVDs, and Electronics (Blitzer et al., 2007). There are 4000 positive and 4000 negative reviews.[1]

For these two data sets, we select 8000 words with highest document-frequency to generate the vocabulary. Stopwords[2] are removed and a normalized term-frequency representation is used. In order to construct the lexical prior knowledge matrix $\mathbf{U}_0$, we use the sentiment lexicon generated by (Hu and Liu, 2004). It contains 2,006 positive words (e.g., "beautiful") and 4,783 negative words (e.g., "upset").

4.2 Unsupervised Sentiment Classification

Our first experiment is to explore the benefits of incorporating the geometric information in the unsupervised paradigm (that is $\mathbf{C}^v = \mathbf{0}$). Therefore, the third part in equation (7) will be ignored. For this unsupervised paradigm of GNMTF, we empirically set $\alpha = \delta = \gamma = 1$, $\sigma_1 = \sigma_2 = 1$, $Iter = 100$ and run GNMTF 10 repeated times to remove any randomness caused by the random initialization. Due to limited space, we do not present the impacts of the parameters on the learning model. Now we compare our proposed GNMTF with the following four categories of methods:

(1) Lexicon-Based Methods (LBM in short): Taboada et al. (2011) proposed to incorporate intensification and negation to refine the sentiment score for each document. This is the state-of-the-art lexicon-based method for unsupervised sentiment classification.

(2) Document Clustering Methods: We choose the most representative cluster methods, K-means, NMTF, Information-Theoretic Co-clustering (ITCC) (Dhillon et al., 2003), and Euclidean Co-clustering method (ECC) (Cho et al., 2004). We set the number of clusters as two in these methods. Note that all these methods do not make use of the sentiment lexicon.

(3) Constrained NMTF (CNMTF in short): Li et al. (2009) incorporated the sentiment lexicon into NMTF as a domain-independent prior constraint.

(4) Graph co-regularized Non-negative Matrix Tri-factorization (GNMTF in short): It is a new algorithm proposed in this paper. We use cosine similarity for constructing the p-nearest neighbor graph for its simplicity. The number of nearest neighbor p is set to 10 empirically both on document and word spaces.

4.2.1 Sentiment Classification Results

The experimental results are reported in Table 2. We perform a significant test, i.e., a t-test with a default significant level of 0.05. From Table 2, we can see that (1) Both CNMTF and GNMTF consider the lexical prior knowledge from off-the-shelf sentiment lexicon and achieve better performance than NMTF. This suggests the importance of the lexical prior knowledge in learning the sentiment classification (row

[1]The data set can be freely downloaded from http://www.cs.jhu.edu/ mdredze/datasets/sentiment/.

[2]http://truereader.com/manuals/onix/stopwords1.html

#	Methods	Movies	Amazon
1	LBM	0.632	0.580
2	K-means	0.543 (-8.9%)	0.535 (-4.5%)
3	NMTF	0.561 (-7.1%)	0.547 (-3.3%)
4	ECC	0.678 (+4.6%)	0.642 (+6.2%)
5	ITCC	0.714 (+8.2%)	0.655 (+7.5%)
6	CNMTF	0.695 (+6.3%)	0.658 (+7.8%)
7	**GNMTF**	**0.736 (+10.4%)**	**0.705 (+12.5%)**

Table 2: Sentiment classification accuracy of unsupervised paradigm on the data sets. Improvements of K-means, NMTF, ITCC, ECC, CNMTF and GNMTF over baseline LBM are shown in parentheses.

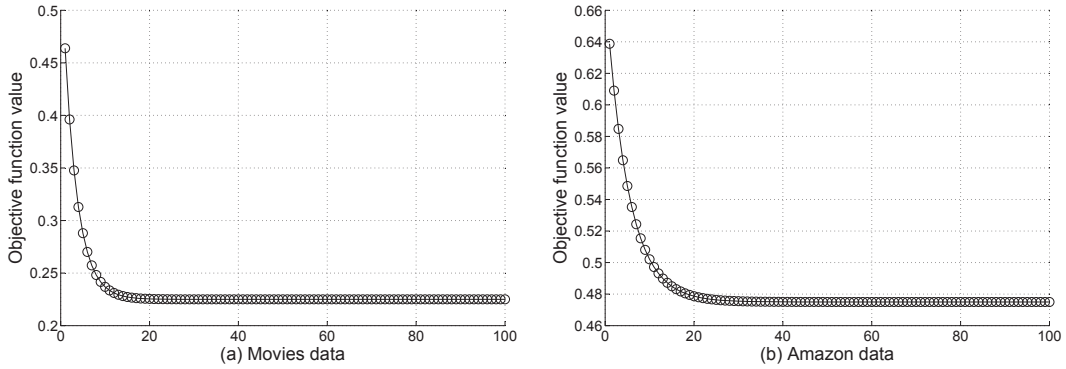


Figure 1: Convergence curves of GNMTF on both data sets.

3 vs. row 6 and row 7); (2) Regardless of the data sets, our GNMTF significantly outperforms state-of-the-art CNMTF and achieves the best performance. This shows the superiority of geometric information and graph co-regularization framework (row 4 vs. row 5, the improvements are statistically significant at $p < 0.05$).

4.2.2 Convergence Behavior

In subsection 3.3.1, we have shown that the multiplicative updates given by equations (8)$\sim$(10) are convergent. Here, we empirically show the convergence behavior of GNMTF.

Figure 1 shows the convergence curves of GNMTF on Movies and Amazon data sets. From the figure, y-axis is the value of objective function and x-axis denotes the iteration number. We can see that the multiplicative updates for GNMTF converge very fast, usually within 50 iterations.

4.3 Semi-supervised Sentiment Classification

In this subsection, we describe our proposed GNMTF with a few labeled documents. For this semi-supervised paradigm of GNMTF, we empirically set $Iter = 100$, $\sigma_1 = \sigma_2 = 2$, $\alpha = \beta = \delta = \gamma = 1$ and $p = 10$ on document and word spaces and also run 10 repeated times to remove any randomness caused by the random initialization. Due to limited space, we do not give an in-depth parameter analysis. For CNMTF, we set $\alpha = \beta = 1$ for fair comparison. We also compare our proposed GNMTF with some representative semi-supervised approaches described in (Li et al., 2009): (1) Semi-supervised learning with local and global consistency (Consistency Method in short) (Zhou et al., 2004); (2) Semi-supervised learning using gaussian fields and harmonic functions (GFHF in short) (Zhu et al., 2003). Besides, we also compare the results of our proposed GNMTF with the representative supervised classification method: support vector machine (SVM), which has been widely used in sentiment classification (Pang et al., 2002).

The results are presented in Figure 2. From the figure, we can see that GNMTF outperforms other methods over the entire range of number of labeled documents on both data sets. By this observation, we can conclude that taking the geometric information can still improve the sentiment classification accuracy in semi-supervised paradigm.

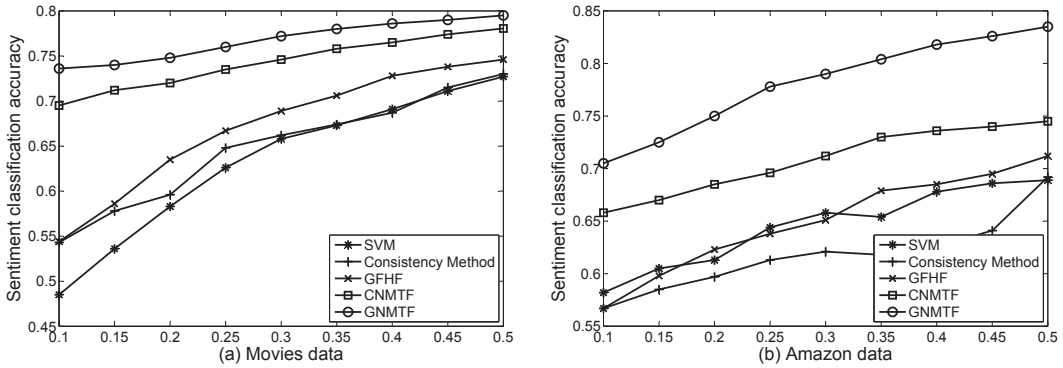


Figure 2: Sentiment classification accuracy vs. different percentage of labeled documents, where x-axis denotes the number of documents labeled as a fraction of the original labeled documents.

5 Related Work

Sentiment classification has gained widely interest in NLP community, we point the readers to recent books (Pang and Lee, 2008; Liu, 2012) for an in-depth survey of literature on sentiment analysis.

Methods for automatically classifying sentiments expressed in products and movie reviews can roughly be divided into supervised and unsupervised (or semi-supervised) sentiment analysis. Supervised techniques have been proved promising and widely used in sentiment classification (Pang et al., 2002; Pang and Lee, 2008; Liu, 2012). However, the performance of these methods relies on manually labeled training data. In some cases, the labeling work may be time-consuming and expensive. This motivates the problem of learning robust sentiment classification via unsupervised (or semi-supervised) paradigm.

The most representative way to perform semi-supervised paradigm is to employ partial labeled data to guide the sentiment classification (Goldberg and Zhu, 2006; Sindhwani and Melville, 2008; Wan, 2009; Li et al., 2011). However, we do not have any labeled data at hand in many situations, which makes the unsupervised paradigm possible. The most representative way to perform unsupervised paradigm is to use a sentiment lexicon to guide the sentiment classification (Turney, 2002; Taboada et al., 2011) or learn sentiment orientation via a matrix factorization clustering framework (Li et al., 2009; **?**; Hu et al., 2013). In contrast, we perform sentiment classification with the different model formulation and learning algorithm, which considers both word-level and document-level sentiment-related contextual information (e.g., the neighboring words or documents tend to share the same sentiment polarity) into a unified framework. The proposed framework makes use of the valuable geometric information to compensate the problem of lack of labeled data for sentiment classification. In addition, some researchers also explored the matrix factorization techniques for other NLP tasks, such as relation extraction (Peng and Park, 2013) and question answering (Zhou et al., 2013)

Besides, many studies address some other aspects of sentiment analysis, such as cross-domain sentiment classification (Blitzer et al., 2007; Pan et al., 2010; Hu et al., 2011; Bollegala et al., 2011; Glorot et al., 2011), cross-lingual sentiment classification (Wan, 2009; Lu et al., 2011b; Meng et al., 2012) and imbalanced sentiment classification (Li et al., 2011), which are out of scope of this paper.

6 Conclusion and Future Work

In this paper, we propose a novel algorithm, called graph co-regularized non-negative matrix tri-factorization (GNMTF), from a geometric perspective. GNMTF assumes that if two words (or documents) are sufficiently close to each other, they tend to share the same sentiment polarity. To achieve this, we encode the geometric information by constructing the nearest neighbor graphs, in conjunction with a non-negative matrix tri-factorization framework. We derive an efficient algorithm for learning the factorization, analyze its complexity, and provide proof of convergence. Our empirical study on two open data sets validates that GNMTF can consistently improve the sentiment classification accuracy in comparison to state-of-the-art methods.

There are some ways in which this research could be continued. First, some other ways should be considered to construct the graphs (e.g., hyperlinks between documents, synonyms or co-occurrences between words). Second, we will try to extend the proposed framework for other aspects of sentiment analysis, such as cross-domain or cross-lingual settings.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61303180 and No. 61272332), the Beijing Natural Science Foundation (No. 4144087), CCF Opening Project of Chinese Information Processing, and also Sponsored by CCF-Tencent Open Research Fund. We thank the anonymous reviewers for their insightful comments.

References

M. Belkin and P. Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Proceedings of NIPS*, pages 585-591.

J. Blitzer, M. Dredze and F. Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In *Proceedings of ACL*, pages 440-447.

D. Bollegala, D. Weir, and J. Carroll. 2011. Using multiples sources to construct a sentiment sensitive thesaurus. In *Proceedings of ACL*, pages 132-141.

S. Boyd and L. Vandenberghe. 2004. *Convex Optimization*. Cambridge university press.

D. Cai, X. He, J. Han, and T. Huang. 2011. Graph regularized non-negative matrix factorization for data representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8): 1548-1560.

H. Cho, I. Dhillon, Y. Guan, and S. Sra. 2004. Minimum sum squared residue co-clutering of gene expression data. In *Proceedings of SDM*, pages 22-24.

F. Chung. 1997. Spectral graph theory. *Regional Conference Series in Mathematics*, Volume 92.

I. Dhillon, S. Mallela, and D. Modha. 2003. Information-theoretic Co-clustering. In *Proceedings of KDD*, pages 89-98.

C. Ding, T. Li, W. Peng, and H. Park. 2006. Orthogonal non-negative matrix tri-factorization for clustering. In *Proceedings of KDD*, pages 126-135.

X. Glorot, A. Bordes, and Y. Bengio. 2011. Domain adaptation for larage-scale sentiment classification: a deep learning approach. In *Proceedings of ICML*.

A. Goldberg and X. Zhu. 2006. Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization. In *Proceedings of NAACL Workshop*.

Y. He, C. Lin and H. Alani. 2011. Automatically extracting polarity-bearing topics for cross-domain sentiment classification. In *Proceedings of ACL*, pages 123-131.

M. Hu and B. Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of KDD*.

X. Hu, J. Tang, H. Gao, and H. Liu. 2013. Unsupervised sentiment analysis with emotional signals. In *Proceedings of WSDM*.

X. Hu, N. Sun, C. Zhang, and T. Chua. 2009. Exploiting internal and external semantics for the clustering of short texts using world knowldge. In *Proceedings of CIKM*, pages 919-928.

H. Kim and H. Park. 2008. Non-negative matrix factorization based on alternating non-negativity constrained least squares and active set method. *SIAM J Matrix Anal Appl*, 30(2):713-730.

D. Lee and H. Seung. 2001. Algorithms for non-negative matrix factorization. In *Proceedings of NIPS*.

S. Li, Z. Wang, G. Zhou, and S. Lee. 2011. Semi-supervised learning for imbalanced sentiment classification. In *Proceedings of IJCAI*, pages 1826-1831.

T. Li, Y. Zhang, and V. Singhwani. 2009. A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In *Proceedings of ACL*, pages 244-252.

C. Lin. 2007. Projected gradient methods for nonnegative matrix factorization. *Neural Comput*, 19(10):2756-2779.

B. Liu. 2012. Sentiment analysis and opinion mining. *Morgan & Claypool Publishers*.

B. Lu, C. Tan, C. Cardie, and B. Tsou. 2011. Joint bilingual sentiment classification with unlabeled parallel corpora. In *Proceedings of ACL*, pages 320-330.

Y. Lu, M. Castellanos, U. Dayal, and C. Zhai. 2011. Automatic construction of a context-aware sentiment lexicon: an optimization approach. In *Proceedings of WWW*, pages 347-356.

X. Meng, F. Wei, X. Liu, M. Zhou, G. Xu, and H. Wang. 2012. Cross-lingual mixture model for sentiment classification. In *Proceedings of ACL*, pages 572-581.

V. Ng, S. Dasgupta, and S. Arifin. 2006. Examing the role of linguistic knowlege sources in the automatic identification and classificaton of reviews. In *Proceedings of ACL*.

S. Pan, X. Ni, J. Sun, Q. Yang, and Z. Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of WWW*.

B. Pang and L. Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Informaiton Retrieval*, pages 1-135.

B. Pang, L. Lee, S. Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79-86.

S. Riedel, L. Yao, A. McCallum, and B. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of NAACL*.

W. Peng and D. Park. 2011. Generative adjective sentiment dictionary for social media sentiment analysis using constrained nonnegative matrix factorization. In *Proceedings of ICWSM*.

S. Roweis and L. Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323-2326.

V. Sindhwani and P. Melville. 2008. Document-word co-regulariztion for semi-supervised sentiment analysis. In *Proceedings of ICDM*, pages 1025-1030.

J. Tenenbaum, V. Silva, and J. Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319-2323.

M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics*.

P. Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL*, pages 417-424.

X. Wan. 2009. Co-training for cross-lingual sentiment classification. In *Proceedings of ACL*, pages 235-243.

D. Zhou, Q. Bousquet, T. Lal, J. Weston, and B. Scholkopf. 2004. Learning with local and global consistency. In *Proceedings of NIPS*.

G. Zhou, F. Liu, Y. Liu, S. He, and J. Zhao. 2013. Statistical machine translation improves question retrieval in community question answering via matrix factorization. In *Proceedings of ACL*, pages 852-861.

X. Zhu, Z. Ghahramani, and J. Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of ICML*.

Hybrid Deep Belief Networks for Semi-supervised Sentiment Classification

Shusen Zhou[⋆] **Qingcai Chen**[†] **Xiaolong Wang**[†] **Xiaoling Li**[⋆]
[⋆] School of Information and Electrical Engineering, Ludong University, Yantai 264025, China.
[†] Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China.
zhoushusen@gmail.com, qingcai.chen@hitsz.edu.cn
wangxl@insun.hit.edu.cn, appleling@live.cn

Abstract

In this paper, we develop a novel semi-supervised learning algorithm called hybrid deep belief networks (HDBN), to address the semi-supervised sentiment classification problem with deep learning. First, we construct the previous several hidden layers using restricted Boltzmann machines (RBM), which can reduce the dimension and abstract the information of the reviews quickly. Second, we construct the following hidden layers using convolutional restricted Boltzmann machines (CRBM), which can abstract the information of reviews effectively. Third, the constructed deep architecture is fine-tuned by gradient-descent based supervised learning with an exponential loss function. We did several experiments on five sentiment classification datasets, and show that HDBN is competitive with previous semi-supervised learning algorithm. Experiments are also conducted to verify the effectiveness of our proposed method with different number of unlabeled reviews.

1 Introduction

Recently, more and more people write reviews and share opinions on the World Wide Web, which present a wealth of information on products and services (Liu et al., 2010). These reviews will not only help other users make better judgements but they are also useful resources for manufacturers of products to keep track and manage customer opinions (Wei and Gulla, 2010). However, there are large amount of reviews for every topic, it is difficult for a user to manually learn the opinions of an interesting topic. Sentiment classification, which aims to classify a text according to the expressed sentimental polarities of opinions such as '*positive*' or '*negtive*', '*thumb up*' or '*thumb down*', '*favorable*' or '*unfavorable*' (Li et al., 2010), can facilitate the investigation of corresponding products or services.

In order to learn a good text classifier, a large number of labeled reviews are often needed for training (Zhen and Yeung, 2010). However, labeling reviews is often difficult, expensive or time consuming (Chapelle et al., 2006). On the other hand, it is much easier to obtain a large number of unlabeled reviews, such as the growing availability and popularity of online review sites and personal blogs (Pang and Lee, 2008). In recent years, a new approach called semi-supervised learning, which uses large amount of unlabeled data together with labeled data to build better learners (Zhu, 2007), has been developed in the machine learning community.

There are several works have been done in semi-supervised learning for sentiment classification, and get competitive performance (Li et al., 2010; Dasgupta and Ng, 2009; Zhou et al., 2010). However, most of the existing semi-supervised learning methods are still far from satisfactory. As shown by several researchers (Salakhutdinov and Hinton, 2007; Hinton et al., 2006), deep architecture, which composed of multiple levels of non-linear operations, is expected to perform well in semi-supervised learning because of its capability of modeling hard artificial intelligent tasks. Deep belief networks (DBN) is a representative deep learning algorithm achieving notable success for text classification, which is a directed belief nets with many hidden layers constructed by restricted Boltzmann machines (RBM), and refined by a gradient-descent based supervised learning (Hinton et al., 2006). Ranzato and Szummer (Ranzato and

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1341–1349, Dublin, Ireland, August 23-29 2014.

Szummer, 2008) propose an algorithm to learn text document representations based on semi-supervised auto-encoders that are combined to form a deep network. Zhou et al. (Zhou et al., 2010) propose a novel semi-supervised learning algorithm to address the semi-supervised sentiment classification problem with active learning. The key issue of traditional DBN is the efficiency of RBM training. Convolutional neural networks (CNN), which are specifically designed to deal with the variability of two dimensional shapes, have had great success in machine learning tasks and represent one of the early successes of deep learning (LeCun et al., 1998). Desjardins and Bengio (Desjardins and Bengio, 2008) adapt RBM to operate in a convolutional manner, and show that the convolutional RBM (CRBM) are more efficient than standard RBM.

CRBM has been applied successfully to a wide range of visual and audio recognition tasks (Lee et al., 2009a; Lee et al., 2009b). Though the success of CRBM in addressing two dimensional issues, there is still no published research on the using of CRBM in textual information processing. In this paper, we propose a novel semi-supervised learning algorithm called hybrid deep belief networks (HDBN), to address the semi-supervised sentiment classification problem with deep learning. HDBN is a hybrid of RBM and CRBM deep architecture, the bottom layers are constructed by RBM, and the upper layers are constructed by CRBM, then the whole constructed deep architecture is fine tuned by a gradient-descent based supervised learning based on an exponential loss function.

The remainder of this paper is organized as follows. In Section 2, we introduce our semi-supervised learning method HDBN in details. Extensive empirical studies conducted on five real-world sentiment datasets are presented in Section 3. Section 4 concludes our paper.

2 Hybrid deep belief networks

2.1 Problem formulation

The sentiment classification dataset composed of many review documents, each review document composed of a bag of words. To classify these review documents using corpus-based approaches, we need to preprocess them in advance. The preprocess method for these reviews is similar with (Zhou et al., 2010). We tokenize and downcase each review and represent it as a vector of unigrams, using binary weight equal to 1 for terms present in a vector. Moreover, the punctuations, numbers, and words of length one are removed from the vector. Finally, we combine all the words in the dataset, sort the vocabulary by document frequency and remove the top 1.5%, because many of these high document frequency words are stopwords or domain specific general-purpose words.

After preprocess, each review can be represented as a vector of binary weight $\mathbf{x}^i$. If the j^{th} word of the vocabulary is in the i^{th} review, $\mathbf{x}_j^i = 1$; otherwise, $\mathbf{x}_j^i = 0$. Then the dataset can be represented as a matrix:

$$\mathbf{X} = \left[\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{R+T}\right] = \begin{bmatrix} x_1^1, x_1^2, \ldots, x_1^{R+T} \\ x_2^1, x_2^2, \ldots, x_2^{R+T} \\ \vdots, \vdots, \ldots, \vdots \\ x_D^1, x_D^2, \ldots, x_D^{R+T} \end{bmatrix} \tag{1}$$

where R is the number of training reviews, T is the number of test reviews, D is the number of feature words in the dataset. Every column of $\mathbf{X}$ corresponds to a sample $\mathbf{x}$, which is a representation of a review. A sample that has all features is viewed as a vector in $\mathbb{R}^D$, where the i^{th} coordinate corresponds to the i^{th} feature.

The L labeled reviews are chosen randomly from R training reviews, or chosen actively by active learning, which can be seen as:

$$\mathbf{X}^L = \mathbf{X}^R(\mathbf{S}),\ \mathbf{S} = [s_1,\ ...,\ s_L],\ 1 \le s_i \le R \tag{2}$$

where $\mathbf{S}$ is the index of selected training reviews to be labeled manually.

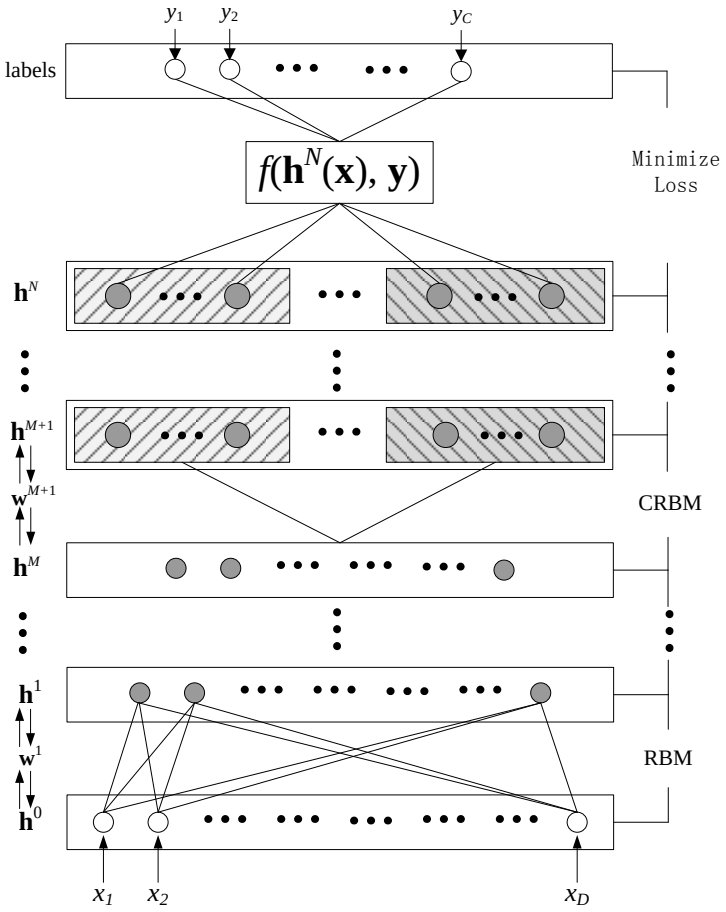


Figure 1: Architecture of HDBN.

The L labels correspond to L labeled training reviews is denoted as:

$$\mathbf{Y}^L = \left[\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^L\right] = \begin{bmatrix} y_1^1, y_1^2, \ldots, y_1^L \\ y_2^1, y_2^2, \ldots, y_2^L \\ \vdots, \vdots, \ldots, \vdots \\ y_C^1, y_C^2, \ldots, y_C^L \end{bmatrix} \tag{3}$$

where C is the number of classes. Every column of $\mathbf{Y}$ is a vector in $\mathbb{R}^C$, where the j^{th} coordinate corresponds to the j^{th} class.

$$y_j^i = \begin{cases} 1 & \text{if } \mathbf{x}^i \in j^{\text{th}} \text{ class} \\ -1 & \text{if } \mathbf{x}^i \notin j^{\text{th}} \text{ class} \end{cases} \tag{4}$$

For example, if a review $\mathbf{x}^i$ is positive, $\mathbf{y}^i = [1, -1]'$; otherwise, $\mathbf{y}^i = [-1, 1]'$.

We intend to seek the mapping function $\mathbf{X} \rightarrow \mathbf{Y}$ using the L labeled data and all unlabeled data. After training, we can determine $\mathbf{y}$ using the mapping function when a new sample $\mathbf{x}$ comes.

2.2 Architecture of HDBN

In this part, we propose a novel semi-supervised learning method HDBN to address the problem formulated in Section 2.1. The sentiment datasets have high dimension (about 10,000), and computation complexity of convolutional calculation is relatively high, so we use RBM to reduce the dimension of review with normal calculation firstly. Fig. 1 shows the deep architecture of HDBN, a fully interconnected directed belief nets with one input layer $\mathbf{h}^0$, N hidden layers $\mathbf{h}^1, \mathbf{h}^2, ..., \mathbf{h}^N$, and one label layer at the top. The input layer $\mathbf{h}^0$ has D units, equal to the number of features of sample review $\mathbf{x}$. The hidden

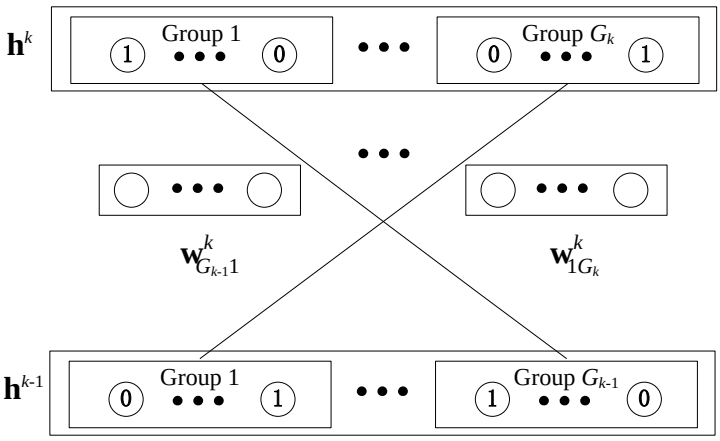


Figure 2: Architecture of CRBM.

layer has M layers constructed by RBM and $N-M$ layers constructed by CRBM. The label layer has C units, equal to the number of classes of label vector $\mathbf{y}$. The numbers of hidden layers and the number of units for hidden layers, currently, are pre-defined according to the experience or intuition. The seeking of the mapping function $\mathbf{X} \rightarrow \mathbf{Y}$, here, is transformed to the problem of finding the parameter space $\mathbf{W} = \{\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^N\}$ for the deep architecture.

The training of the HDBN can be divided into two stages:

1. HDBN is constructed by greedy layer-wise unsupervised learning using RBMs and CRBMs as building blocks. L labeled data and all unlabeled data are utilized to find the parameter space $\mathbf{W}$ with N layers.

2. HDBN is trained according to the exponential loss function using gradient descent based supervised learning. The parameter space $\mathbf{W}$ is refined using L labeled data.

2.3 Unsupervised learning

As show in Fig. 1, we construct HDBN layer by layer using RBMs and CRBMs, the details of RBM can be seen in (Hinton et al., 2006), and CRBM is introduced below.

The architecture of CRBM can be seen in Fig. 2, which is similar to RBM, a two-layer recurrent neural network in which stochastic binary input groups are connected to stochastic binary output groups using symmetrically weighted connections. The top layer represents a vector of stochastic binary hidden feature $\mathbf{h}^k$ and the bottom layer represents a vector of binary visible data $\mathbf{h}^{k-1}$, $k = M+1, ..., N$. The k^{th} layer consists of G_k groups, where each group consists of D_k units, resulting in $G_k \times D_k$ hidden units. The layer $\mathbf{h}^M$ is consist of 1 group and D_M units. $\mathbf{w}^k$ is the symmetric interaction term connecting corresponding groups between data $\mathbf{h}^{k-1}$ and feature $\mathbf{h}^k$. However, comparing with RBM, the weights of CRBM between the hidden and visible groups are shared among all locations (Lee et al., 2009a), and the calculation is operated in a convolutional manner (Desjardins and Bengio, 2008).

We define the energy of the state $(\mathbf{h}^{k-1}, \mathbf{h}^k)$ as:

$$E\left(\mathbf{h}^{k-1}, \mathbf{h}^k; \theta\right) = -\sum_{s=1}^{G_{k-1}} \sum_{t=1}^{G_k} (\tilde{w}^k_{st} * h^{k-1}_s) \bullet h^k_t - \sum_{s=1}^{G_{k-1}} b^{k-1}_s \sum_{u=1}^{D_{k-1}} h^{k-1}_s - \sum_{t=1}^{G_k} c^k_t \sum_{v=1}^{D_k} h^k_t \qquad (5)$$

where $\theta = (\mathbf{w}, \mathbf{b}, \mathbf{c})$ are the model parameters: w^k_{st} is a filter between unit s in the layer $\mathbf{h}^{k-1}$ and unit t in the layer $\mathbf{h}^k$, $k = M+1, ..., N$. The dimension of the filter w^k_{st} is equal to $D_{k-1} - D_k + 1$. b^{k-1}_s is the s^{th} bias of layer $\mathbf{h}^{k-1}$ and c^k_t is the t^{th} bias of layer $\mathbf{h}^k$. A tilde above an array ($\tilde{w}$) denote flipping the array, $*$ denote valid convolution, and $\bullet$ denote element-wise product followed by summation, i.e., $A \bullet B = trA^T B$ (Lee et al., 2009a).

Similar to RBM, Gibbs sampler can be performed based on the following conditional distribution.

The probability of turning on unit v in group t is a logistic function of the states of $\mathbf{h}^{k-1}$ and w_{st}^k:

$$p\left(h_{t,v}^k = 1|\mathbf{h}^{k-1}\right) = \text{sigm}\left(c_t^k + (\sum_s \tilde{w}_{st}^k * h_s^{k-1})_v\right) \tag{6}$$

The probability of turning on unit u in group s is a logistic function of the states of $\mathbf{h}^k$ and w_{st}^k:

$$p\left(h_{s,u}^{k-1} = 1|\mathbf{h}^k\right) = \text{sigm}\left(b_s^{k-1} + (\sum_t w_{st}^k \star h_t^k)_u\right) \tag{7}$$

A star $\star$ denote full convolution.

2.4 Supervised learning

In HDBN, we construct the deep architecture using all labeled reviews with unlabeled reviews by inputting them one by one from layer $\mathbf{h}^0$. The deep architecture is constructed layer by layer from bottom to top, and each time, the parameter space $\mathbf{w}^k$ is trained by the calculated data in the $k-1^{th}$ layer.

Algorithm 1: Algorithm of HDBN

Input: data $\mathbf{X}$, $\mathbf{Y}^L$
number of training data R; number of test data T;
number of layers N; number of epochs Q;
number of units in every hidden layer $D_1 ... D_N$;
number of groups in every convolutional hidden layer $G_M ... G_N$;
hidden layer $\mathbf{h}^1, \ldots, \mathbf{h}^M$;
convolutional hidden layer $\mathbf{h}^{M+1}, \ldots, \mathbf{h}^{N-1}$;
parameter space $\mathbf{W} = \{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$;
biases $\mathbf{b}$, $\mathbf{c}$; momentum ϑ and learning rate η;
Output: deep architecture with parameter space $\mathbf{W}$
1. Greedy layer-wise unsupervised learning
for $k = 1; k \leq N-1$ **do**
  for $q = 1; q \leq Q$ **do**
    for $r = 1; r \leq R+T$ **do**
      Calculate the non-linear positive and negative phase:
      if $k \leq M$ **then**
        Normal calculation.
      else
        Convolutional calculation according to Eq. 6 and Eq. 7.
      end
      Update the weights and biases:
      $w_{st}^k = \vartheta w_{st}^k + \eta\left(\left\langle h_{s,r}^{k-1} h_{t,r}^k \right\rangle_{P_0} - \left\langle h_{s,r}^{k-1} h_{t,r}^k \right\rangle_{P_1}\right)$
    end
  end
end
2. Supervised learning based on gradient descent
$\arg\min_W \sum_{i=1}^{L}\sum_{j=1}^{C} \exp(-h^N(x_j^i) y_j^i)$

According to the $\mathbf{w}^k$ calculated by RBM and CRBM, the layer $\mathbf{h}^k, k = 1, \ldots, M$ can be computed as following when a sample $\mathbf{x}$ inputs from layer $\mathbf{h}^0$:

$$h_t^k(\mathbf{x}) = \text{sigm}\left(c_t^k + \sum_{s=1}^{D_{k-1}} w_{st}^k h_s^{k-1}(\mathbf{x})\right), t = 1, \ldots, D_k \tag{8}$$

When $k = M+1, \ldots, N-1$, the layer $\mathbf{h}^k$ can be represented as:

$$h_t^k(\mathbf{x}) = \mathrm{sigm}\left(c_t^k + \sum_{s=1}^{G_{k-1}} \tilde{w}_{st}^k * h_s^{k-1}(\mathbf{x})\right), t = 1, \ldots, G_k \tag{9}$$

The parameter space $\mathbf{w}^N$ is initialized randomly, just as backpropagation algorithm.

$$h_t^N(\mathbf{x}) = c_t^N + \sum_{s=1}^{G_{N-1} \times D_{N-1}} w_{st}^N h_s^{N-1}(\mathbf{x}),\ t = 1, \ldots, D_N \tag{10}$$

After greedy layer-wise unsupervised learning, $\mathbf{h}^N(\mathbf{x})$ is the representation of $\mathbf{x}$. Then we use L labeled reviews to refine the parameter space $\mathbf{W}$ for better discriminative ability. This task can be formulated as an optimization problem:

$$\arg\min_{\mathbf{W}} \quad \mathrm{f}\left(h^N\left(\mathbf{X}^L\right), \mathbf{Y}^L\right) \tag{11}$$

where

$$\mathrm{f}\left(h^N\left(\mathbf{X}^L\right), \mathbf{Y}^L\right) = \sum_{i=1}^{L}\sum_{j=1}^{C} \mathrm{T}\left(h_j^N\left(\mathbf{x}^i\right) y_j^i\right) \tag{12}$$

and the loss function is defined as

$$T(r) = \exp(-r) \tag{13}$$

We use gradient-descent through the whole HDBN to refine the weight space. In the supervised learning stage, the stochastic activities are replaced by deterministic, real valued probabilities.

2.5 Classification using HDBN

The training procedure of HDBN is given in Algorithm 1. For the training of HDBN architecture, the parameters are random initialized with normal distribution. All the reviews in the dataset are used to train the HDBN with unsupervised learning. After training, we can determine the label of the new data through:

$$\arg\max_j h^N(\mathbf{x}) \tag{14}$$

3 Experiments

3.1 Experimental setup

We evaluate the performance of the proposed HDBN method using five sentiment classification datasets. The first dataset is MOV (Pang et al., 2002), which is a classical movie review dataset. The other four datasets contain products reviews come from the multi-domain sentiment classification corpus, including books (BOO), DVDs (DVD), electronics (ELE), and kitchen appliances (KIT) (Blitzer et al., 2007). Each dataset contains 1,000 positive and 1,000 negative reviews.

The experimental setup is same as (Zhou et al., 2010). We divide the 2,000 reviews into ten equal-sized folds randomly, maintaining balanced class distributions in each fold. Half of the reviews in each fold are random selected as training data and the remaining reviews are used for test. Only the reviews in the training data set are used for the selection of labeled reviews by active learning. All the algorithms are tested with cross-validation.

We compare the classification performance of HDBN with four representative semi-supervised learning methods, i.e., semi-supervised spectral learning (Spectral) (Kamvar et al., 2003), transductive SVM (TSVM) (Collobert et al., 2006), deep belief networks (DBN) (Hinton et al., 2006), and personal/impersonal views (PIV) (Li et al., 2010). Spectral learning, TSVM methods are two baseline methods for sentiment classification. DBN (Hinton et al., 2006) is the classical deep learning method proposed recently. PIV (Li et al., 2010) is a new sentiment classification method proposed recently.

Table 1: HDBN structure used in experiment.

Dataset	Structure
MOV	100-100-4-2
KIT	50-50-3-2
ELE	50-50-3-2
BOO	50-50-5-2
DVD	50-50-5-2

Table 2: Test accuracy with 100 labeled reviews for semi-supervised learning.

Type	MOV	KIT	ELE	BOO	DVD
Spectral	67.3	63.7	57.7	55.8	56.2
TSVM	68.7	65.5	62.9	58.7	57.3
DBN	71.3	72.6	73.6	64.3	66.7
PIV	-	**78.6**	70.0	60.1	49.5
HDBN	**72.2**	74.8	**73.8**	**66.0**	**70.3**

3.2 Performance of HDBN

The HDBN architecture used in all our experiments have 2 normal hidden layer and 1 convolutional hidden layer, every hidden layer has different number of units for different sentiment datasets. The deep structure used in our experiments for different datasets can be seen in Table 1. For example, the HDBN structure used in MOV dataset experiment is 100-100-4-2, which represents the number of units in 2 normal hidden layers are 100, 100 respectively, and in output layer is 2, the number of groups in 1 convolutional hidden layer is 4. The number of unit in input layer is the same as the dimensions of each datasets. For greedy layer-wise unsupervised learning, we train the weights of each layer independently with the fixed number of epochs equal to 30 and the learning rate is set to 0.1. The initial momentum is 0.5 and after 5 epochs, the momentum is set to 0.9. For supervised learning, we run 30 epochs, three times of linear searches are performed in each epoch.

The test accuracies in cross validation for five datasets and five methods with semi-supervised learning are shown in Table 2. The results of previous two methods are reported by (Dasgupta and Ng, 2009). The results of DBN method are reported by (Zhou et al., 2010). Li et al. (Li et al., 2010) reported the results of PIV method. The result of PIV on MOV dataset is empty, because (Li et al., 2010) did not report it. HDBN is the proposed method.

Through Table 2, we can see that HDBN gets most of the best results except on KIT dataset, which is just slight worse than PIV method. However, the preprocess of PIV method is much more complicated than HDBN, and the PIV results on other datasets are much worse than HDBN method. HDBN method is adjusted by DBN, all the experiment results on five datasets for HDBN are better than DBN. This could be contributed by the convolutional computation in HDBN structure, and proves the effectiveness of our proposed method.

3.3 Performance with variance of unlabeled data

To verify the contribution of unlabeled reviews for our proposed method, we did several experiments with fewer unlabeled reviews and 100 labeled reviews.

The test accuracies of HDBN with different number of unlabeled reviews and 100 labeled reviews on five datasets are shown in Fig. 3. The architectures for HDBN used in this experiment are same as Section 3.2 too, which can be seen in Table 1. We can see that the performance of HDBN is much worse when just using 400 unlabeled reviews. However, when using more than 1200 unlabeled reviews, the performance of HDBN is improved obviously. For most of review datasets, the accuracy of HDBN with 1200 unlabeled reviews is close to the accuracy with 1600 and 2000 unlabeled reviews. This proves that HDBN can get competitive performance with just few labeled reviews and appropriate number of

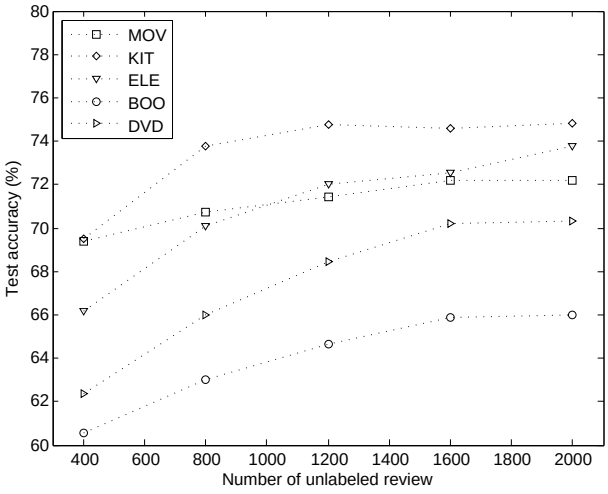


Figure 3: Test accuracy of HDBN with different number of unlabeled reviews on five datasets.

unlabeled reviews. Considering the much time needed for training with more unlabeled reviews and less accuracy improved for HDBN method, we suggest using appropriate number of unlabeled reviews in real application.

4 Conclusions

In this paper, we propose a novel semi-supervised learning method, HDBN, to address the sentiment classification problem with a small number of labeled reviews. HDBN seamlessly incorporate convolutional computation into the DBN architecture, and use CRBM to abstract the review information effectively. To the best of our knowledge, HDBN is the first work that uses convolutional neural network to improve sentiment classification performance. One promising property of HDBN is that it can effectively use the distribution of large amount of unlabeled data, together with few label information in a unified framework. In particular, HDBN can greatly reduce the dimension of reviews through RBM and abstract the information of reviews through the cooperate of RBM and CRBM. Experiments conducted on five sentiment datasets demonstrate that HDBN outperforms state-of-the-art semi-supervised learning algorithms, such as SVM and DBN based methods, using just few labeled reviews, which demonstrate the effective of deep architecture for sentiment classification.

Acknowledgements

This work is supported in part by National Natural Science Foundation of China (No. 61300155, No. 61100115 and No. 61173075), Natural Science Foundation of Shandong Province (No. ZR2012FM008), Science and Technology Development Plan of Shandong Province (No. 2013GNC11012), Science and Technology Research and Development Funds of Shenzhen City (No. JC201005260118A and No. JC201005260175A), and Scientific Research Fund of Ludong University (LY2013004).

References

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2006. *Semi-supervised learning*. MIT Press, USA.

Ronan Collobert, Fabian Sinz, Jason Weston, and Leon Bottou. 2006. Large scale transductive svms. *Journal of Machine Learning Research*, 7:1687–1712.

Sajib Dasgupta and Vincent Ng. 2009. Mine the easy, classify the hard: A semi-supervised approach to automatic sentiment classfication. In *Joint Conference of the 47th Annual Meeting of the Association for Computational*

Linguistics and 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, pages 701–709, Stroudsburg, PA, USA. Association for Computational Linguistics.

Guillaume Desjardins and Yoshua Bengio. 2008. Empirical evaluation of convolutional rbms for vision. Technical report.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554.

Sepandar Kamvar, Dan Klein, and Christopher Manning. 2003. Spectral learning. In *International Joint Conferences on Artificial Intelligence*, pages 561–566, Catalonia, Spain. AAAI Press.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. 2009a. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning*, pages 609–616, Montreal, Canada. ACM.

Honglak Lee, Yan Largman, Peter Pham, and Andrew Y. Ng. 2009b. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems*, pages 1096–1103, Vancouver, B.C., Canada. NIPS Foundation.

Shoushan Li, Chu-Ren Huang, Guodong Zhou, and Sophia Yat Mei Lee. 2010. Employing personal/impersonal views in supervised and semi-supervised sentiment classification. In *Annual Meeting of the Association for Computational Linguistics*, pages 414–423, Uppsala, Sweden. Association for Computational Linguistics.

Yang Liu, Xiaohui Yu, Xiangji Huang, and Aijun An. 2010. S-plasa+: Adaptive sentiment analysis with application to sales performance prediction. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 873–874, New York, NY, USA. ACM.

Bo Pang and Lillian Lee. 2008. *Opinion Mining and Sentiment Analysis*, volume 2 of *Foundations and Trends in Information Retrieval*.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Conference on Empirical Methods in Natural Language Processing*, pages 79–86, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marc'Aurelio Ranzato and Martin Szummer. 2008. Semi-supervised learning of compact document representations with deep networks. In *International Conference on Machine Learning*, pages 792–799, Helsinki, Finland. ACM.

Ruslan Salakhutdinov and Geoffrey E. Hinton. 2007. Learning a nonlinear embedding by preserving class neighbourhood structure. *Journal of Machine Learning Research*, 2:412–419.

Wei Wei and Jon Atle Gulla. 2010. Sentiment learning on product reviews via sentiment ontology tree. In *Annual Meeting of the Association for Computational Linguistics*, pages 404–413, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yi Zhen and Dit-Yan Yeung. 2010. Sed: Supervised experimental design and its application to text classification. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299–306, Geneva, Switzerland. ACM.

Shusen Zhou, Qingcai Chen, and Xiaolong Wang. 2010. Active deep networks for semi-supervised sentiment classification. In *International Conference on Computational Linguistics*, pages 1515–1523, Beijing, China. Coling 2010 Organizing Committee.

Xiaojin Zhu. 2007. *Semi-supervised learning literature survey*. Ph.D. thesis.

Latent Dynamic Model with Category Transition Constraint for Opinion Classification

Takeshi S. Kobayakawa
NHK / 1-10-11 Kinuta, Setagaya-ku, Tokyo 157-8510, JAPAN
kobayakawa.t-ko@nhk.or.jp

Abstract

Latent models for opinion classification are studied. Training a probabilistic model with a number of latent variables is found unstable in some cases; thus this paper presents how to construct a stable model for opinion classification by constraining classification transitions. The baseline model is a CRF classification model with plural latent variables, dynamically constructed from the dependency parsed tree. The aim of the baseline model is to have each latent variable convey a partial sentiment of the input sentence which is not explicitly given in the training data, and the complete sentiment of the sentence is computed by summing up such partial sentiment where those latent variables hold. Since such a conventional model has many degeneracies in principle, a model with a category transition constraint is proposed, which is expressed by a novel penalty term in the objective function for training the model. The constraint is such that the sentiment of a partial sentence more likely propagates to the same sentiment of the complete sentence, rather than to another sentiment. The effectiveness and the robustness of the proposed model are confirmed by the experiments on binary as well as multi-class opinion classification task.

1 Introduction

Opinion classification is a task to classify sentences into given categories, according to sentiment, evaluation, or some opinion-related points of view. A practical implementation of opinion classification would be very useful for managing customer relationships at contact centers, etc. The classification problem may be binary or sometimes multi-class.

One of the simplest modeling process is to use explicit bag features, such as word surfaces, polarity information from the sentiment dictionary, etc. Thanks to the good behavior of the Maximum Entropy or the Conditional Random Field (CRF) model, the maximum likelihood training is straight-forward, because the local optimum is always the global optimum.

A challenge is to introduce into the model latent variables, which are not explicitly observable. The implicit modeling here is supposed to express ambiguities of natural language; the partial sentiment of the sentence is not determined until the end of the sentence. This paper presents in detail a probabilistic model with latent variables. The baseline model is a CRF model which is constructed dynamically according to the dependency parsed tree and which contains latent variables on the nodes that correspond to the chunked expressions (Nakagawa et al., 2010). The latent variables in the model are expected to convey a partial sentiment of the sentence, such as the sentiment of the dependency-parsed-subtree itself, which is not explicitly observable.

Although this idea is attractive, it actually suffers from numerical instability. Our aim here is to find a way to deal with this problem. We tried using a global optimizer and investigated the behavior of the model, to ensure that this lack of stability comes from the degeneracy of the model.

Our contribution to remedy this problem is as follows: We propose a model with a penalty on category transitions and compare several optimizers to train the model. We also confirm the stability of the model

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1350–1359, Dublin, Ireland, August 23-29 2014.

by applying it to the multi-class classification problem. We also investigate the origin of the degeneracy and how regularizer works for this type of classification model, to see the robustness of the model.

2 Related Work

Studies on opinion classification have more than a decade of history, including the pioneer work (Turney, 2002; Pang et al., 2002), followed by (Takamura et al., 2005; Brun, 2012). Our baseline model treats the sentiment of a partial sentence (Nakagawa et al., 2010); CRF with latent variables are constructed dynamically by the dependency structure tree of the input sentence.

CRF model was first used in sequence labeling tasks (Lafferty, 2001). The model does not suffer from the label-bias problem as does the Maximum Entropy model, and its parameter estimation is well behaved with the help of a convex loss function. However, the convexity of the loss function of the CRF model does not hold anymore when there are unobserved data or latent variables (Sutton and McCallum, 2007).

Latent variables were first used with CRF for the purpose of noun coreference (McCallum and Wellner, 2005), and object recognition (Quattoni et al., 2005). Latent variables have been used to construct meaning representations in a process called grounded language acquisition (Liang et al., 2009). Another approach with hidden variables has been used an recursive auto-encoder to reduce the reliance on sentiment dictionaries (Socher et al., 2011).

Machine learning, used for training such models, is largely based on the concept of numerical optimization. A general discussion of convex optimization can be found in (Boyd and Vandenberghe, 2004); it can be proven that the `log-sum-exp` type of a convex function is still a convex function. This is the situation with the CRF model without latent variables. Since convexity holds, the best solution to the problem would be a numerical local optimization. A general discussion of local optimization can be found in the textbook (Nocedal and Wright, 2006), who is one of the authors of the Limited memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) optimizer program. Many NLP application programs use probabilistic models, including this optimizer or its derived work. As the long name of the BFGS algorithm shows, the state-of-the-art local optimizer has a long history. Another textbook (Press et al., 2007) covers a global optimization algorithm, Simulated Annealing, originating from (Kirkpatrick et al., 1983). The main idea comes from the statistical physics of equilibrium; when a material is warm, its energy is distributed in excited states, whereas when it is cooled, it is very probable that the system will end in the ground state, which corresponds to the global minimum point of the energy.

Previous studies on CRF with latent variables were trained either by setting the initial parameters randomly (Nakagawa et al., 2010), or by online training (McCallum and Wellner, 2005). As far as the author knows, this is the first study to impose a penalty between latent variables in CRF model, and to compare several optimization algorithms.

3 The Model

The model studied is a CRF model, which has set of conditional probabilities whose log is a linear combination of model parameters associated with features given by (Lafferty, 2001):

$$\log p_\Lambda(\mathbf{y}|\mathbf{x}) \propto \sum_{v\in V,k} \mu_k g_k(v, \mathbf{y}|_v, \mathbf{x}) + \sum_{e\in E,k} \lambda_k f_k(e, \mathbf{y}|_e, \mathbf{x}), \tag{1}$$

where $\mathbf{x}$ is an input vector and $\mathbf{y}$ is an output label sequence, and $\mathbf{y}|_v$ and $\mathbf{y}|_e$ are the vertex v and the edge e related to the component of $\mathbf{y}$, respectively. The model parameter vector Λ is estimated from the training data, whose components are the sum of the two sets: a vertex feature set $(\mu_1, \mu_2, \dots)$ and an edge feature set $(\lambda_1, \lambda_2, \dots)$. The complete set of features are supposed to be enumerated and fixed, so that each feature can be indicated by an index k in a rather relaxed way. g_k and f_k are so-called feature functions, to indicate whether the feature in the argument appears in the input, or not.

Nakagawa et al. (2010) proposed a CRF model with latent variables that uses a dependency parsed structure, where the latent variables are expected to convey the sentiment classifying the part underneath the parsed tree. We choose this model as a baseline and continue the same kind of treatment of the latent

variables. Section 3.1 and 3.2 briefly review this baseline model, then the proposed model follows in Section 3.3.

3.1 Sentence Classification Model with Latent Variables

To classify sentence $\mathbf{x}$ into a given set of a class, say C, the classification problem is formulated as:

$$\arg\max_{s_0 \in C} p_\Lambda(s_0|\mathbf{x}), \tag{2}$$

where s_0 is a class label of the complete sentence and $\mathbf{x}$ is a given sentence such that

$$p_\Lambda(s_0|\mathbf{x}) = \sum_{\mathbf{s}} p_\Lambda(s_0, \mathbf{s}|\mathbf{x}), \tag{3}$$

where $\mathbf{s}$ is the latent variables to be summed up, and Λ is the set of all parameters in the model. Each element of $\mathbf{s}$ takes one of these class labels, corresponding to the partial sentiment of the sentence, and is to be summed up to construct the complete sentiment of the sentence. A partial sentiment of a sentence is sometimes ambiguous, so it is treated as such an unobserved variable. The model parameters Λ are estimated from the training data; the sentiment label is only available for a whole sentence, not for a part of the sentence.

3.2 Dependency Structure as a Graphical Model

The given sentence $\mathbf{x}$ is parsed into a dependency structure of phrase chunks:

$$\mathbf{x} \xrightarrow{\text{Dependency Parsing}} G(\mathbf{x}) = \{V(\mathbf{x}), E(\mathbf{x})\}, \tag{4}$$

where $V(\mathbf{x})$ are the set of chunks and $E(\mathbf{x})$ is the set of dependency arcs. The dependency structure is regarded as a graphical model, an example of which is shown in Figure 1. The words are chunked up into phrases, and the dependencies between those chunks are determined by dependency parsing. Each chunk corresponds to a variable that is supposed to convey a sentiment.

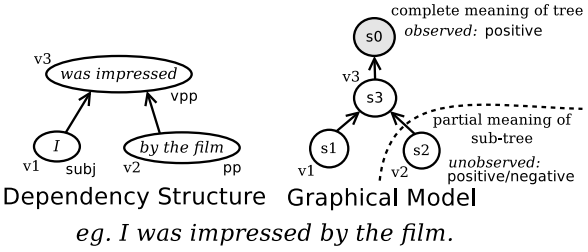


Figure 1: Correspondence between dependency structure and graphical representation of the model

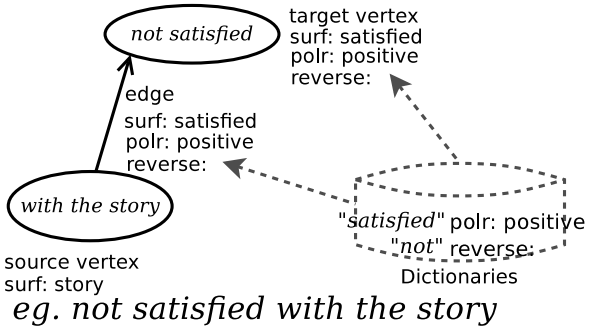


Figure 2: Features attached to vertices and edges

Vertex features	Edge features
word surface unigrams	word surface unigrams of the parent vertex
succeeding word surface bigrams	word surface unigrams of the child vertex
sentiment information from a dictionary	sentiment information from a dictionary
negation expression from a dictionary	negation expression from a dictionary
meaning label of functional expression	meaning label of functional expression of the parent vertex
	meaning label of functional expression of the child vertex

Table 1: Summary of features adopted in the model

Every feature belongs to one of two types: vertex features or edge features. Vertex features are those locally related to a vertex, and edge features are those that affect both sides of vertices of the dependency. The specific features adopted in the model are summarized in Table 1.

A symbol μ as a notation for model parameters related to vertex features. It has two indices since they are related to a vertex feature and a classification category. As for the vertex feature f_v related part, the log probability that the vertex v has the category s is:

$$\log p_{f_v}(s) = \mu_{f_v,s}. \tag{5}$$

A symbol λ as a notation for model parameters related to edge features. It has three indices since they are related to an edge feature and classification categories of both sides of vertices of the edge.

As for the edge feature f_e related part, the log conditional probability that the target vertex takes the category s_2 is:

$$\log p_{f_e}(s_2|s_1) = \lambda_{f_e,s_2,s_1}, \tag{6}$$

given the category of the source vertex is s_1.

Multiple features can be attached either on a vertex or on a edge. So, the whole vertex or edge probability is constructed as follows:

$$\log p_v(s) = \sum_{f_v \in F^{(\text{vertex})}(v)} \log p_{f_v}(s) = \sum_{f_v \in F^{(\text{vertex})}(v)} \mu_{f_v,s}, \tag{7}$$

$$\log p_e(s_2|s_1) = \sum_{f_e \in F^{(\text{edge})}(e)} \log p_{f_e}(s_2|s_1) = \sum_{f_e \in F^{(\text{edge})}(e)} \lambda_{f_e,s_2,s_1}, \tag{8}$$

where $F^{(\text{vertex})}(v)$ is a set of features attached to a vertex v, and where $F^{(\text{edge})}(e)$ is those attached to an edge e.

Finally, the probability of a given sentence $\mathbf{x}$ is constructed as a log-linear model:

$$\log p(s_0, \mathbf{s}|\mathbf{x}) = \sum_{v \in V(\mathbf{x})} \log p_v(s^{(\text{v})}) + \sum_{e \in E(\mathbf{x})} \log p_e(s^{(\text{target}(e))}|s^{(\text{source}(e))}), \tag{9}$$

where the set of vertices and edges are dynamically constructed from the dependency parsed tree of the sentence, *i.e.* eq. (4), and the notation source(e) and target(e) are the source and target vertex of the edge e, respectively (Figure 2.)

Care is necessary when assigning values to the latent variables in eq. (9). Because each latent variable which is assigned on a vertex and the connecting edges, share the same values, the latent variables must be summed up in such way; The summations can be done efficiently by using dynamic programming (*a.k.a.* the factor graph in graphical model terminology.) The tables of probabilities are constructed for each vertex, and the tree is constructed in a bottom up manner.

The sets[1] of all the vertex and edge features appearing in the training data $\mathcal{D}$ are denoted as $\mathcal{V}^{(\mathcal{D})}$ and $\mathcal{E}^{(\mathcal{D})}$ respectively:

$$\mathcal{V}^{(\mathcal{D})} = \sum_{\mathbf{x} \in \mathcal{D}} \sum_{v \in V(\mathbf{x})} F^{(\text{vertex})}(v), \qquad \mathcal{E}^{(\mathcal{D})} = \sum_{\mathbf{x} \in \mathcal{D}} \sum_{e \in E(\mathbf{x})} F^{(\text{edge})}(e), \tag{10}$$

so that the complete set of parameters is:

$$\Lambda = \{\mu_{v,c} | v \in \mathcal{V}^{(\mathcal{D})}, c \in C\} + \{\lambda_{e,c_1,c_2} | e \in \mathcal{E}^{(\mathcal{D})}, \quad c_1, c_2 \in C\},$$

where the number of parameters is $\mathcal{V}^{(\mathcal{D})} \times C + \mathcal{E}^{(\mathcal{D})} \times C \times C$.

[1] Note that the following equations up to eq. (11) are in the terminology of set theory; the addition is done with the elimination of duplicated elements, and a product means a direct product, and a n-th power is an abbreviation of n direct products of the set.

The log likelihood of the training data is given by

$$\mathbb{L}(\Lambda;\mathcal{D}) = \sum_{\mathbf{x}\in\mathcal{D}} \log\left(\sum_{\mathbf{s}\in C^{|V(\mathbf{x})|}} p_\Lambda(s_0,\mathbf{s}|\mathbf{x})\right), \tag{11}$$

where $|V(\mathbf{x})|$ is the number of vertices in a given sentence $\mathbf{x}$, and where s_0 is the correct classification label for $\mathbf{x}$, and where $\mathbf{s}$ is the set of latent variables whose number is as many as $|V(\mathbf{x})|$.

3.3 Category Transition Penalty

We found that the classification accuracy of the trained model remained low, because little number of parameters for edge features moved away from initial non-contributing values during the training. The following form of regularizer[2] is used for the training:

$$\mathbb{R}(\Lambda) = C_{\text{regularizer}} \left|\Lambda - \frac{1}{n}\cdot\mathbb{1}\right|^2. \tag{12}$$

The constant $C_{\text{regularizer}}$ is the strength of the regularizer, and no matter how strong, the classification accuracy did not improve in our preliminary experiments. This phenomenon seems to be because of the degeneracies the model has in principle.

Degeneracy is a notion to explain the same probabilities of different configurations. If the two different configurations are preferably distinguished, an asymmetric treatment of them is required.

In order to avoid such extra degeneracies, we introduce a novel constraint between latent variables expressed by the following penalty term:

$$\mathbb{P}(\Lambda) = C_{\text{penalty}} \sum_{f_e\in\mathcal{E}^{(\mathcal{D})}} \left(\sum_{s_1=s_2} (\log p_{f_e}(s_2|s_1) - \log C_{\text{same}})^2 + \sum_{s_1\neq s_2} (\log p_{f_e}(s_2|s_1) - \log C_{\text{different}})^2\right), \tag{13}$$

to satisfy

$$C_{\text{same}} + (n-1)C_{\text{different}} = 1, \tag{14}$$

where C_{penalty} is the weight of this penalty, and where C_{same} and $C_{\text{different}}$ are constant probabilities for the following two cases; that is the categories connected to the other side of the edge should be the same or different, respectively. This term is incorporated into the objective function for training the model, to form a soft constraint that diminishes the change in the classification category.

3.4 Model Training

The maximum likelihood training of a probabilistic model is a constrained optimization problem. A probabilistic interpretation is possible if and only if 1) all probabilities are non-negative, and 2) the sum of the probabilities are one (or renormalizable to one).

Using log probabilities almost automatically satisfies the first condition: Real number in log space corresponds to positive number in anti-log space, so that the only consideration needed is zero probability (which corresponds to negative infinity in the log space.) In the experiments, overflow is checked that none occurred in the final results. The model parameter to express zero probability could be finite but reasonably small, instead of zero.

As for the second condition, instead of the strict constraint, we adopted a quadratic penalty in log space:

$$\mathbb{C}(\Lambda) = C_{\text{prob}} \left(\sum_{f_v\in\mathcal{V}^{(\mathcal{D})}} \left(\log\sum_{s\in C} p_{f_v}(s)\right)^2 + \sum_{f_e\in\mathcal{E}^{(\mathcal{D})}} \sum_{s_1\in C} \left(\log\sum_{s_2\in C} p_{f_e}(s_2|s_1)\right)^2\right), \tag{15}$$

[2]The offset $\frac{1}{n}\cdot\mathbb{1}$ in the regularizer is so as to avoid singularity around zero probabilities in real space, which causes negative infinity in log space calculation.

where the strictly normalized probabilities lead to zero penalty; otherwise, a quadratic penalty is given according to the amount away from strictly normalized probabilities. The weight of the penalty C_{prob} is chosen to be heavy enough for the sum of the model probability to be adequately normalized.

Finally, we adopted the probability calculation in the log space, with the quadratic penalty for normalization during the training. The objective function for training the model is:

$$\mathbb{O}(\Lambda; \mathcal{D}) = \mathbb{L}(\Lambda; \mathcal{D}) - \mathbb{R}(\Lambda) - \mathbb{P}(\Lambda) - \mathbb{C}(\Lambda). \tag{16}$$

4 Experiments

Experiments are conducted to evaluate the effect of the proposed penalty term expressed by eq. (13).

4.1 Test Sets

Two kinds of test set in Japanese were used: Opinions in the Kyoto University and NTT Blog (KNB) Corpus[3]and Comments on an TV cultural program. Both sets are balanced in the numbers sentence categories. The characteristics of each test set are shown in Table 2.

The first test set, the KNB Corpus, is a collection of opinion sentences about Kyoto sightseeing spots, cellular phones, gourmet food, and sports. The sentences are categorized in terms of many aspects, and we used the sentences labeled with Evaluation+ or Evaluation–. "Evaluation" is a category of subjective but non-emotional opinions.

The second test set is used for non-binary classification. To make this set, viewers were asked to comment (in Japanese natural language) on a certain TV program. The comments are classified into categories, *i.e.* evaluations, impressions, requirements and questions. The following four categories are used: positive and negative evaluations, and impressions of what the viewers learned from the program, and what they thought after watching the program.

Name of Test Set	# of Sentences	Categories
Opinions in KNB Corpus	328	2(Evaluation +/–)
Comments on TV cultural program	432	4(positive/negative evaluations, what viewers learned/think)

Table 2: Characteristics of Test Set

4.2 NLP Resources

The input sentence was processed by a morphological analyzer to split it into words, since Japanese is an agglutinative language. The words were then chunked and the dependencies between those chunks were determined. Functional multi-word expressions were also detected by the analyzer we developed. We used a dictionary of sentiment expressions, and one of negation expressions, both of which were distributed with the KNB Corpus.

We did not prepare any special sentiment dictionary for the second test set because preparing such a dictionary is too costly. Furthermore, robustness can be estimated without a domain-adapted dictionary. The parameter values for training the models were tuned for the first test set, and the tuned values were used without any extra tuning for the second test set. In this situation, the first test set can be regarded as the development test set, and the second as the evaluation test set.

4.3 Latent Dynamic Model with Category Transition Constraint

Experiments on classifying opinions using the KNB Corpus are shown in Table 3. The rightmost column is a trivial baseline, where the classification category is decided by the majority occurrence of sentiment

[3]The corpus is publicly available from `http://nlp.ist.i.kyoto-u.ac.jp/kuntt/\#ga739fe2`, and the details of corpus are explained at `http://alaginrc.nict.go.jp/opinion/index_e.html`. We excluded short sentences, made up of a few words, that were exclamations rather than natural complete sentences. They do not form tree structures, which are not aimed to this study. Accuracy of experiments conducted below are different from that by (Nakagawa et al., 2010) in that only the subset of the test set that satisfy the condition is used.

words in the sentence. This method needs no training, so only one figure is indicated. The other columns are figures for trained CRF; closed tests are the case where all the training data is used for the evaluation as well, shown in the upper row, while 10-folds open test are the case is 1:9 split of data for evaluation and training and the evaluation is done cyclically 10 times, shown in the lower row. The leftmost column is CRF without latent variables. The 2nd left is a model with latent variables but without penalty, which is the model in (Nakagawa et al., 2010). The 3rd left is a model with latent variables that has a category transition penalty, which is the proposed model. The proposed model performs the best accuracy among all the models.

Experiments on classifying opinions using the Comments on the TV cultural Program are shown in Table 4. Majority voting is not possible because a suitable dictionary that has the same class polarity as this classification problem does not exist.

	Trained CRF			(no training)
	without latent variables	with latent variables		Majority
		non-penalized	penalized	Voting
closed test	95.12	95.73	95.73	
10-fold open test	61.59	63.72	**65.55**	64.79

Table 3: Effect of Latent Variables and Penalized Model (Opinions in KNB Corpus)

	Trained CRF			(no training)
	without latent variables	with latent variables		Majority
		non-penalized	penalized	Voting
closed test	99.54	99.77	99.31	
10-fold open test	60.42	60.42	**64.81**	N/A

Table 4: Effect of Penalized Latent Dynamic Model (Comments on TV cultural program)

4.4 Comparison of Optimizers

Three optimization algorithms for model training were compared, two of which are local optimizers (BFGS and Steepest Descent), and one of which is a global optimizer (Simulated Annealing).

BFGS was used as batch training where all of the training data were used during the training iteration. Two types of initial parameter configurations were tried for BFGS; initial parameters have the same fixed values, or were chosen randomly. Steepest Descent (SD) was used as online training where some portion (*i.e.* chunk) of the training data were used during an iteration. Two types of chunk selection scheme were tried for Steepest Descent; chunks were fixed during the training, or chunks were randomly shuffled after every complete loop of the whole training data.

Simulated Annealing was adopted as a global optimizer. We implemented feature level granularity for acceptance or rejection: Every parameter corresponding to a feature was randomly moved, and decided probabilistically whether or not to accept according to the Boltzmann distribution under scheduled cooling down. Although all of these methods utilize random variables, they were used in different ways. The accuracy ranges of ten trials are shown in Table 5 and 6.

The results show that the proposed model trained by a global optimizer outperforms models trained by the other local optimizers (Steepest Descent and BFGS); The penalty in the proposed model works well because the degeneracies in the penalized model seem to decrease, and the computation is noteworthy stable.

5 Discussion

The degeneracy in the model is illustrated in Figure 3 and 4.

Firstly, convergence for penalized model is quicker, as shown in Figure 3; The horizontal axis is the number of iterations, and the left vertical axis is the acceptance ratio, where the lower is well converged.

	Batch Training (BFGS) parameter initialization		Online Training (SD) chunked data selection		Simulated Annealing
	random	*fixed*	*shuffled*	*fixed*	
closed test	89.33-82.32	87.80	76.83-72.56	76.83	95.73-95.73
10-fold open test	62.80-58.54	59.15	66.46-65.55	65.55	65.55-64.63

Table 5: Comparison of Optimizers (Opinions in KNB Corpus)

	Batch Training (BFGS) parameter initialization		Online Training (SD) chunked data selection		Simulated Annealing
	random	*fixed*	*shuffled*	*fixed*	
closed test	85.65-35.65	88.19	47.69-45.37	45.60	99.31-99.31
10-fold open test	58.56-34.72	59.95	38.89-36.81	37.50	**64.81-64.58**

Table 6: Comparison of Optimizers (Comments on TV cultural program)

The acceptance ratio remains high for the non-penalized model, while the penalized model quickly descends. The dash line indicates the log likelihood of the training data, for penalized and non-penalized models, which are almost identical.

Secondly, in order to split degeneracy, only a small penalty is adequate, as illustrated in Figure 4; how less the penalty is, as long as it exists, the improvement remains. The horizontal axis is the strength of the proposed penalty, C_{penalty} in eq.(13). The vertical axis is the classification accuracy. The dash line is a line fit for the accuracy by non-zero penalty. In general, such an extra constraint term for the original model may change the model itself, so, the less it is the better. That is the reason for decreasing accuracy when large penalty is used. The significant jump in the accuracy between zero and non-zero penalty strongly suggests the existence of degeneracies in the original model: Infinitesimally small penalty can lead to break those degeneracies.

Local optima usually do not matter when regularizers are used in training. However, according to our experiments in this type of models, the conventional regularizers are not able to avoid such local optima no matter how strong they are. The reason the introduced penalty works well for this model is considered that the term works for excessive latent variables, which are not controlled by the ordinary regularizers. The ordinary regularizers only works for excessive number of explicitly observed parameters (*i.e.* features). If only a few latent variables are used, such a penalty is not necessary, just as a regularizer is not necessary for a small number of features. When the model is constructed dynamically and the number of latent variables grows, there appear a number of latent variables having excessive freedom, which need to be controlled.

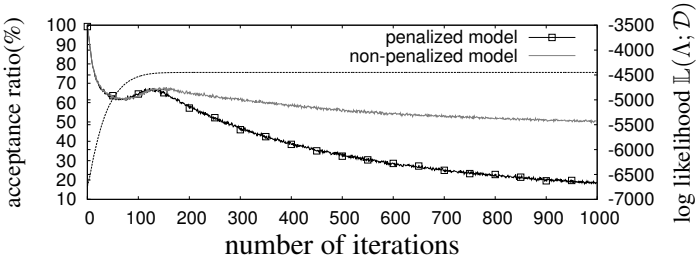


Figure 3: Transition of the Acceptance Ratio

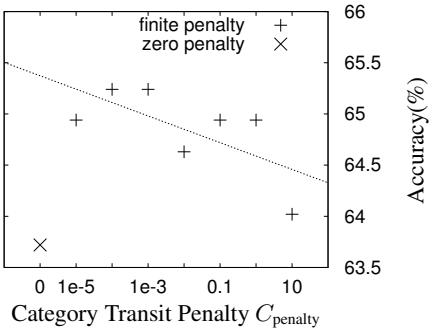


Figure 4: Effect of Weak Penalty

6 Conclusion

A latent dynamic model with a category transition constraint is proposed for opinion classification task. The constraint is such that the sentiment of a partial sentence tends to propagate toward the complete sentiment of the whole sentence, which is realized, in the objective function, by our novel term that penalizes, diminishing the change in the classification category.

According to our experiments, the penalized latent dynamic model outperforms the conventional model, not only in binary but also in multi-class opinion classification.

The comparison of optimizers strongly suggests that the degeneracies in the conventional model deteriorate the performance, and the proposed model solves such a defect. The numerical stability of training the proposed model is also confirmed.

Acknowledgments

The author would like to thank to Jun'ichi Tsujii, Akiko Aizawa, Yusuke Miyao, Takuya Matsuzaki, Hideki Tanaka and Tadashi Kumano. The author would also like to express the greatest gratitude for the discussions with Akio Kobayashi, Hiroyuki Segi, Tsuneo Hirano, Clara K. Hirano, and Mariko Hirano. Mariko Hirano also helped some of the experiments.

References

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

Caroline Brun. 2012. Learning opinionated patterns for contextual opinion detection. In *Proceedings of COLING 2012: Posters*, pages 165–174, Mumbai, India, December. The COLING 2012 Organizing Committee.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. 1983. Optimization by simulated annealing. *Science*, 220(4598):671–680.

John Lafferty. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pages 282–289. Morgan Kaufmann.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, Suntec, Singapore, August. Association for Computational Linguistics.

Andrew McCallum and Ben Wellner. 2005. Conditional models of identity uncertainty with application to noun coreference. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 905–912. MIT Press, Cambridge, MA.

Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using crfs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794, Los Angeles, California, June. Association for Computational Linguistics.

Jorge Nocedal and Stephen J. Wright. 2006. *Numerical Optimization*. Springer Verlag, 2nd edition.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 79–86. Association for Computational Linguistics, July.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 2007. *Numerical Recipes*. Cambridge University Press, 3rd edition.

Ariadna Quattoni, Michael Collins, and Trevor Darrell. 2005. Conditional random fields for object recognition. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1097–1104. MIT Press, Cambridge, MA.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 151–161, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Charles Sutton and Andrew McCallum, 2007. *Introduction to Statistical Relational Learning*, chapter 4 An Introduction to Conditional Random Fields. The MIT Press. also available on e-Print archive: arXiv:1011.4088v1.

Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting semantic orientations of words using spin model. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 133–140, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Peter Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Sentence Compression for Target-Polarity Word Collocation Extraction

Yanyan Zhao[1], **Wanxiang Che**[2], **Honglei Guo**[3], **Bing Qin**[2], **Zhong Su**[3] **and Ting Liu**[2*]
1: Department of Media Technology and Art, Harbin Institute of Technology
2: Department of Computer Science and Technology, Harbin Institute of Technology
3: IBM Research-China
{yyzhao, bqin, tliu}@ir.hit.edu.cn, {guohl, suzhong}@cn.ibm.com

Abstract

Target-polarity word (T-P) collocation extraction, a basic sentiment analysis task, relies primarily on syntactic features to identify the relationships between targets and polarity words. A major problem of current research is that this task focuses on customer reviews, which are natural or spontaneous, thus posing a challenge to syntactic parsers. We address this problem by proposing a framework of adding a sentiment sentence compression (*Sent_Comp*) step before performing T-P collocation extraction. *Sent_Comp* seeks to remove the unnecessary information for sentiment analysis, thereby compressing a complicated sentence into one that is shorter and easier to parse. We apply a discriminative conditional random field model, with some special sentiment-related features, in order to automatically compress sentiment sentences. Experiments show that *Sent_Comp* significantly improves the performance of T-P collocation extraction.

1 Introduction

Sentiment analysis deals with the computational treatment of opinion, sentiment and subjectivity in text (Pang and Lee, 2008), and has received considerable attention in recent years (Liu, 2012). Target-Polarity word (T-P) collocation extraction, which aims to extract the collocation of a target and its corresponding polarity word in a sentiment sentence, is a basic task in sentiment analysis. For example, in a sentiment sentence "这款相机拥有新颖的外形" (*The camera has a novel appearance*), "外形" (*appearance*) is the target, and "新颖" (*novel*) is the polarity word that modifies "外形" (*appearance*). According, ⟨外形, 新颖⟩ (⟨*appearance, novel*⟩) is the T-P collocation. Generally, T-P collocation is a basic and complete sentiment unit, thus is very useful for many sentiment analysis applications.

Features derived from syntactic parse trees are particularly useful for T-P collocation extraction (Abbasi et al., 2008; Duric and Song, 2012). For example, the syntactic relation "Adj $\overset{\text{ATT}}{\curvearrowleft}$ Noun", where the ATT denotes an attributive syntactic relation, can be used as an important evidence to extract the T-P collocation ⟨外形, 新颖⟩ (⟨*appearance, novel*⟩) in the above sentiment sentence (Bloom et al., 2007; Qiu et al., 2011; Xu et al., 2013).

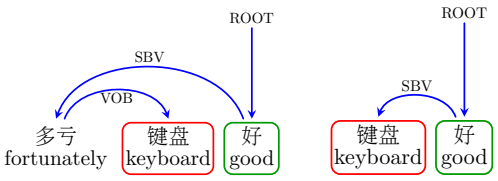
However, one major problem of these approaches is the "naturalness" of sentiment sentences, that is, such sentences are more natural or spontaneous compared with normal sentences, thus posing a challenge to syntactic parsers. Accordingly, many wrong syntactic features have been produced and these can further result in the poor performance of T-P collocation extraction. Taking the sentence in Figure 1(a) as an example, because the word "多亏" (*fortunately*) is so chatty,[1] the parsing result is wrong. Thus, are unable to extract the T-P collocation ⟨键盘, 好⟩ (⟨*keyboard, good*⟩).

To solve the "naturalness" problem, we can train a parser on sentiment sentences. Unfortunately, annotating such data will cost us a lot of time and effort. Instead, in this paper we produce a sentence compression model, *Sent_Comp*, which is designed especially to compress complicated sentiment sentences into formal and easier to parse ones, further improving T-P collocation extraction.

*Correspondence author: tliu@ir.hit.edu.cn

[1]Note that, in Figure 1, the Chinese word "多亏" is chatty, although its translated English word "fortunately" is not. In this paper, we focus on processing the Chinese data.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1360–1369, Dublin, Ireland, August 23-29 2014.



(a) before compression (b) after compression

Figure 1: Parse trees before and after compression.

This idea is motivated by the observation that, current syntactic parsers usually perform accurately for short, simple and formal sentences, whereas error rates increase for longer, more complex or more natural and spontaneous sentences (Finkel et al., 2008). Hence, the improvement in syntactic parsing performance would have a ripple effect over T-P collocation extraction. For example, we can compress the sentence in Figure 1(a) into a shortened sentence in Figure 1(b) by removing the chatty part "多亏" (*fortunately*). We can see that the shortened sentence is now well-formed (in Chinese) and its parse tree is correct, making it easier to accurately extract T-P collocation.

Traditional sentence compression aims to obtain a shorter grammatical sentence by retaining important information (usually important grammar structure) (Jing, 2000). For example, the sentence "Overall, this is a great camera." can be compressed into "This is a camera." by removing the adverbial "overall" and the modifier "great". However, the modifier "great" is a polarity word and very important for sentiment analysis. Therefore, *Sent_Comp* model for sentiment sentences is different from the traditional compression models, because it needs to retain the important sentiment information, such as the polarity word. Hence, using *Sent_Comp*, the above sentence can be compressed into "This is a great camera."

We regard *Sent_Comp* as a sequence labeling task, which can be solved by a conditional random fields (CRF) model. Instead of seeking the manual rules on parse trees for compression, as in other studies (Vickrey and Koller, 2008), this method is an automatic procedure. In this work, we introduce some sentiment-related features to retain the sentiment information for *Sent_Comp*.

We apply *Sent_Comp* as the first step in the T-P collocation extraction task. First, we compress the sentiment sentences into easier to parse ones using *Sent_Comp*, after which we employ the state-of-the-art T-P collocation extraction approach on the compressed sentences. Experimental results on a Chinese corpus of four product domains show the effectiveness of our approach.

The main contributions of this paper are as follows:

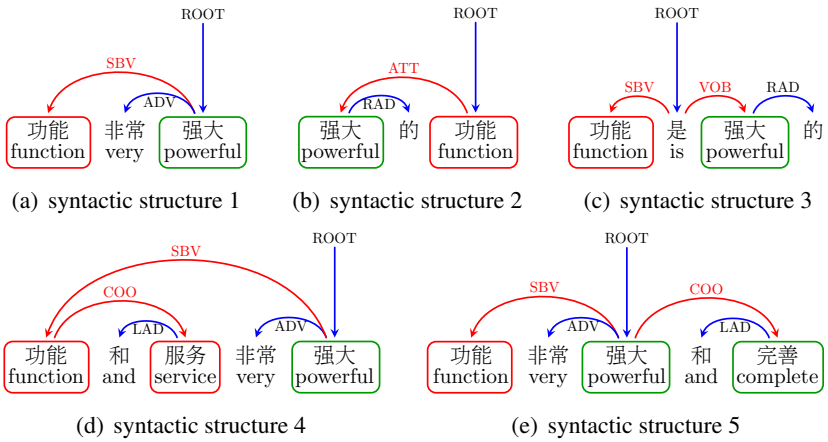
- We present a framework of using sentiment sentence compression preprocessing step to improve T-P collocation extraction. This framework can better solve the "over-natural" problem of sentiment sentences, which poses a challenge to syntactic parsers. More importantly, the idea of this framework can be applied to some other sentiment analysis tasks that rely heavily on syntactic results.

- We develop a simple yet effective compression model *Sent_Comp* for sentiment sentences. To the best of our knowledge, this is the first sentiment sentence compression model.

2 Background

For our baseline system, we used the state-of-the-art method to extract T-P collocations introduced by Qiu et al. (2011), who proposed a double propagation method. This idea is based on the observation that there is a natural syntactic relationship between polarity words and targets owing to the fact that polarity words are used to modify targets. Furthermore, they also found that polarity words and targets themselves have relations in some sentiment sentences (Qiu et al., 2011).

Based on this idea, in the double propagation method, we first used an initial seed polarity word lexicon and the syntactic relations to extract the targets, which can fall into a new target lexicon. Then we used the target lexicon and the same syntactic relations to extract the polarity words and to subsequently expand the polarity word lexicon. This is an iterative procedure, because this method can iteratively produce the new polarity words and targets back and forth using the syntactic relations.



(a) syntactic structure 1 (b) syntactic structure 2 (c) syntactic structure 3

(d) syntactic structure 4 (e) syntactic structure 5

Figure 2: Example of syntactic structure rules for T-P collocation extraction. We showed five examples from a total of nine syntactic structures. For each kind of syntactic structure (a) to (e), the target is shown with a red box and the polarity word is shown with a green box. Syntactic structures (a) to (c) describe the relations between targets and polarity words. Syntactic structure (d), which is extended from (a), describes the relation between two targets. Syntactic structure (e), which is also extended from (a), describes the relation between two polarity words. Similarly, we can summarize the other four rules extended from (b) and (c) to describe the relations between two targets or two polarity words.

We can see that the syntactic relations are important for this method, and Qiu et al. (2011) proposed eight rules to describe these relations. However, their work only focused on English sentences, whereas the relations for Chinese sentences are different. Thus, in accordance with Chinese grammar, we proposed nine syntactic structure rules between target t and polarity word p in a Chinese T-P collocation $\langle t, p \rangle$.[2] The three main rules are shown below and some example rules are illustrated in Figure 2.

Rule 1: $t \overset{\text{SBV}}{\curvearrowleft} p$, the "subject-verb" structure between t and p, such as the example in Figure 2(a).

Rule 2: $p \overset{\text{ATT}}{\curvearrowleft} t$, that p is an attribute for t, such as the example in Figure 2(b).

Rule 3: $t \overset{\text{SBV}}{\curvearrowleft} \circ \overset{\text{VOB}}{\curvearrowright} p$, the "subject-verb-object" structure between t and p, such as the example in Figure 2(c). The $\circ$ denotes any word.

The other six rules can be extended from the three main rules by obtaining the coordination (COO) relation of t or p, such as $t \overset{\text{SBV}}{\curvearrowleft} \circ \overset{\text{COO}}{\curvearrowright} p$ in Figure 2(e). Note that the POS for t should be noun and for p should be adjective.

As described above, the T-P collocation extraction relies heavily on syntactic parsers. Hence, if we can use the *Sent_Comp* model to improve the performance of parsers, the performance of T-P collocation extraction can also be improved accordingly.

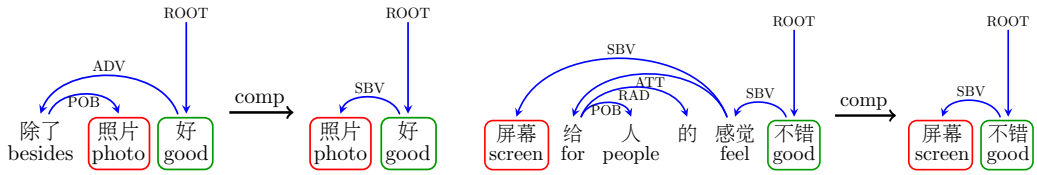
3 Sentiment Sentence Compression

3.1 Problem Analysis

First, we conducted an error analysis for the results of current T-P collocation extraction, from which we observed that the "naturalness" of sentiment sentences is one of the main problems. For examples:

- Chatty form: some sentiment sentences are so chatty, that they bring many difficulties to the parser. For example, in the sentence "多亏键盘好" (*fortunately the keyboard is good*) shown in Figure 1, the usage of the chatty word "多亏" (*fortunately*) affects the accuracy of the syntactic parser.

[2] A Chinese natural language processing toolkit, Language Technology Platform (LTP) (Che et al., 2010), was used as our dependency parser. More information about the syntactic relations can be found in their paper. The state-of-the-art graph-based dependency parsing model, in the toolkit, was trained on Chinese Dependency Treebank 1.0 (LDC2012T05).



(a) parse tree 1 before and after compression

(b) parse tree 2 before and after compression

Figure 3: "Naturalness" problem of sentiment sentences.

- Conjunction word usage: conjunction words are often used in sentiment sentences to show the discourse relations between two sentences. However, there are so many conjunction words in Chinese, some of which can cause errors among parsers. For example, in Figure 3(a), the parse tree of sentence "除了相片较好" (*besides the photo is good*) is wrong because of the usage of the conjunction word "除了" (*besides*).

- Feeling words/phrase usage: in sentiment sentences, people often use some feeling words/phrase, such as "给人的感觉" (*feel like*) in Figure 3(b) or "闻起来" (*smell like*). Given that the current syntactic parser cannot handle the feeling words/phrases very well, the T-P collocation ⟨屏幕, 不错⟩ (⟨$screen, good$⟩) in Figure 3(b) cannot be extracted correctly.

To address the "naturalness" problem, we compressed the sentiment sentences into one that are shorter and easier to parse. Similar to the examples in Figure 1 and 3, the compressed sentences can be easily and correctly parsed. The above analysis can be used as the criteria to guide us in compressing sentiment sentences when annotating, and can also help us exploit more useful features for automatic sentiment sentence compression.

3.2 Task Definition

We focus on studying the methods for extractive sentence compression.[3] Formally, extractive sentence compression aims to shorten a sentence $\mathbf{x} = x_1 \cdots x_n$ into a substring $\mathbf{y} = y_1 \cdots y_m$, where $y_i \in \{x_1, \cdots, x_n\}$, $m \leq n$.

In this paper, similar to Nomoto (2007), we also treated the sentence compression as a sequence labeling task which can be solved by a CRF model. We assigned a compression tag t_i to each word x_i in an original sentence $\mathbf{x}$, where $t_i = \mathsf{N}$ if $x_i \in \mathbf{y}$, else $t_i = \mathsf{Y}$.

A first-order linear-chain CRF is used which defines the following conditional probability:

$$P(\mathbf{t}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_i M_i(t_i, t_{i-1}|\mathbf{x}) \quad (1)$$

where $\mathbf{x}$ and $\mathbf{t}$ are the input and output sequences respectively, $Z(\mathbf{x})$ is the partition function, and M_i is the clique potential for edge clique i. Here, we used the CRFsuite toolkit to train the CRF model.[4]

3.3 Features

The features for *Sent_Comp* are listed in Table 1. Aside from the basic word (w), POS tag (t) and their combination context features (01 – 04), we introduced some sentiment-related features (05 – 06) and latent semantic features (07 – 08) to better handle sentiment analysis data and generalize word features. Then we added the syntactic parse features (09), which are commonly used in traditional sentence compression task.

One sentiment-related feature ($\mathsf{feeling}(\cdot)$) indicates whether a word is a feeling word, which is inspired by the naturalness problem in Figure 3(b). As discussed above, the current parser often produces wrong parse trees because of these feeling words. Therefore, the feeling words tend to be removed from a

[3]Generally, there are two kinds of sentence compression methods: extractive method and abstractive method. Because abstractive method needs more resource and is more complicated, in this paper, we only focus on extractive approach.

[4]www.chokkan.org/software/crfsuite/

Basic Features
01: $w_{i+k}, -1 \leq k \leq 1$
02: $w_{i+k-1} \circ w_{i+k}, 0 \leq k \leq 1$
03: $t_{i+k}, -2 \leq k \leq 2$
04: $t_{i+k-1} \circ t_{i+k}, -1 \leq k \leq 2$
Sentiment-related Features
05: feeling(w_i)
06: polarity(w_i)
Latent Semantic Features
07: suffix(w_i) if $t(w_i) == n$ else prefix(w_i)
08: cluster(w_i)
Syntactic Features
09: dependency(w_i)

Table 1: Features of sentiment sentence compression

sentiment sentence for *Sent_Comp*. We can obtain a feeling word lexicon from HowNet,[5] a popular Chinese sentiment thesaurus, where a feeling word is defined by DEF={perception|感知} tag. Finally, we collected 38 feeling words, such as 发觉 (*realize*), 发现 (*find*), and 认为 (*think*).

The other sentiment-related feature (polarity$(\cdot)$) indicates whether a word is a polarity word. One of the main differences between a sentiment sentence and a formal sentence is that the former often contains polarity words. In contrast to the features of feeling$(\cdot)$, polarity words (e.g., "great" in the sentence "Overall, this is a great camera") tend to be retained, because they are important and special to sentiment analysis. In this paper, we treat polarity words as important features, considering that they are often tagged as modifiers and are easily removed by common sentence compression methods. We can obtain the polarity feature (polarity$(\cdot)$) from a polarity lexicon, which can also be obtained from HowNet.

To generalize the words in sentiment sentences, we proposed two kinds of semantic features. The first one is a suffix or prefix character feature (prefix$(\cdot)$ or suffix$(\cdot)$). In contrast to English, the suffix (for noun) or prefix (for non noun) characters of a Chinese word often carry that word's core semantic information. For example, 自行**车** (*bicycle*), 汽**车** (*car*), and 火**车** (*train*) are all various kinds of 车 (*vehicle*), which is also the suffix of the three words. Given that all of them may become targets, they tend to be retained in compressed sentences. The verbs, **感**觉 and **感**到, can be denoted by their prefix *feel* (感), and can be removed from original sentences because they are feeling words.

We used word clustering features (cluster$(\cdot)$) as the other latent semantic feature to further improve the generalization over common words. Word clustering features contain some semantic information and have been successfully used in several natural language processing tasks, including NER (Miller et al., 2004; Che et al., 2013) and dependency parsing (Koo et al., 2008). For instance, the words 外观 and 样子 (*appearance*) belong to the same word cluster, although they have a different suffix or prefix. Both words are important for T-P collocation extraction and should be retained. We used the Brown word clustering algorithm (Brown et al., 1992) to obtain the word clusters (Liang, 2005). Raw texts were obtained from the fifth edition of Chinese Gigaword (LDC2011T13).

Finally, similar to McDonald (2006), we also added the dependency relation between a word and its parent as the syntactic features. Intuitively, the dependency relations are helpful in carrying out sentence compression. For example, the ROOT relation typically indicates that the word should not be removed because it is the main verb of a sentence.

4 Experiments

4.1 Experimental Setup

4.1.1 Corpus

We conducted the experiments on a Chinese corpus of four product domains, which came from the Task3 of the Chinese Opinion Analysis Evaluation (COAE) (Zhao et al., 2008).[6] Table 2 describes the corpus,

[5] www.keenage.com

[6] www.ir-china.org.cn/coae2008.html

Domain	# reviews	# sentences	# collocations
Camera	138	1,249	1,335
Car	161	1,172	1,312
Notebook	56	623	674
Phone	123	1,350	1,479
All	478	4,394	4,800

Table 2: Corpus statistics for the Chinese corpus of four product domains.

where 4,394 sentiment sentences containing 4,800 T-P collocations are manually found and annotated from 478 reviews.

We ask annotators to manually compress all the sentiment sentences. Specifically, the annotators removed some words from a sentiment sentence according to two criteria stated as follows: (1) removing the word should not change the essential content of the sentence, and (2) removing the word should not change the sentiment orientation of the sentence. In order to assess the quality of the annotation, we sampled 500 sentences from this corpus and asked two annotators to perform the annotation. The resulting word-based Cohen's kappa (Cohen, 1960) (i.e., a measure of inter-annotator agreement ranging from zero to one) of 0.7 indicated a good strength of agreement.

4.1.2 Evaluation

Generally, compressions are evaluated using three criteria (McDonald, 2006), namely, grammaticality, importance, and compression rate. Obviously, the former two are difficult to evaluate objectively. Previous works used human judgment, which entails a difficult and expensive process. In this paper, similar to a common sequence labeling task, we simply used the F-score metric of removed words to roughly evaluate the performance of sentiment sentence compression. Of course, the final effectiveness of sentence compression model can be reviewed by the derived T-P collocation extraction task.

For T-P collocation extraction, we applied the traditional *P*, *R* and *F-score* for the final evaluations. Specially, a fuzzy matching evaluation is adopted for the T-P collocation extraction. That is to say, given an extracted T-P collocation $\langle t, p \rangle$, whose standard result is $\langle t_s, p_s \rangle$, if t is the substring of t_s, and meanwhile p is the substring of p_s, we consider the extracted $\langle t, p \rangle$ is a correct T-P collocation.

4.2 Sentiment Sentence Compression Results

Features	P(%)	R(%)	F(%)
Basic (01 – 04)	76.4	57.4	65.5
+ feeling (05)	75.9	57.6	65.5
+ polarity (06)	76.6	57.6	65.7
+ suffix or prefix (07)	78.4	56.9	66.0
+ cluster (08)	74.9	58.9	65.9
+ dependency (09)	75.3	57.2	65.0
All (01 – 08)	**77.3**	**59.1**	**67.0**
All - feeling (05)	77.1	58.9	66.8

Table 3: The results of sentiment sentence compression with different features.

Results of *Sent_Comp* with different features are shown in Table 3. All results are reported using five-fold cross validation. We can see that the performance is improved when we added feeling[7] and polarity features (05 – 06) respectively, indicating that the sentiment-related features are useful for sentiment sentence compression. In addition, the latent semantic features (07 – 08) are also helpful, especially the suffix or prefix features, which show better performance than the four other kinds of features.

Nonetheless, the dependency features (09) have a negative on compression performance due to the specificity of compression for sentiment sentences. That is because the lower dependency parsing performance on sentiment sentences introduces many wrong dependency relations, which counteract the

[7]In Table 3, although the performance of adding feeling is comparative to the basic system (Basic (01-04)), the system without feeling (All - feeling (05), the last line) is worse than the system using all the features (All (01-08)). This can illustrate the effectiveness of the feeling feature.

Domain	Method	P(%)	R(%)	F(%)
Camera	no_Comp	74.7	58.4	65.6
	manual_Comp	83.4	62.7	71.6
	auto_Comp	80.4	62.1	70.1
Car	no_Comp	68.2	53.1	59.7
	manual_Comp	76.3	57.7	65.7
	auto_Comp	72.3	56.1	63.2
Notebook	no_Comp	74.1	56.8	64.3
	manual_Comp	82.7	64.5	72.5
	auto_Comp	79.7	62.8	70.2
Phone	no_Comp	77.3	60.9	68.1
	manual_Comp	82.7	65.7	73.2
	auto_Comp	80.3	63.3	70.8
All	no_Comp	73.7	57.5	**64.6**
	manual_Comp	81.2	62.5	**70.6**
	auto_Comp	78.1	60.9	**68.4**

Table 4: Results on T-P collocation extraction for four product domains.

contribution of the dependency relation features. This is also the reason why we need to compress sentiment sentences as the first step for T-P collocation extraction. Finally, when we combine all of useful features (01 – 08), the performance achieves the highest score.

It is worth noting that sentiment sentence compression is a new task proposed in this paper. For simplicity, this paper aims to attempt a simple yet effective sentiment sentence compression model. We will polish the *Sent_Comp* model in the future work.

4.3 *Sent_Comp* for T-P Collocation Extraction

We designed three comparative systems to demonstrate the effectiveness of *Sent_Comp* for T-P collocation extraction. Note that, *Sent_Comp* is the first step to process the corpus before T-P collocation extraction. The method for T-P collocation extraction was based on the state-of-the-art method proposed by Qiu et al. (2011) as described in Section 2.

no_Comp - This refers to the system that only uses the T-P collocation extraction method and does not perform sentence compression as the first step.

manual_Comp - This system **manually** compresses the corpus into a new one as the first step, and then applies the T-P collocation extraction method on the new compressed corpus.

auto_Comp - This system uses *Sent_Comp* as the first step to **automatically** compress the corpus into a new one, and then applies the T-P collocation extraction method on the new corpus.

From the descriptions above, we can draw a conclusion that the performance of **manual_Comp** can be considered as the upper bound for the sentiment sentence compression based T-P collocation extraction task.

Table 4 shows the experimental results of the three systems on T-P collocation extraction for four product domains. Here, **manual_Comp** can significantly ($p < 0.01$) improved the *F-score* by approximately 6%,[8] compared with **no_Comp**. This illustrates that the idea of sentiment sentence compression is useful for T-P collocation extraction. Specifically, the proposed method can transform some over-natural sentences into normal ones, further influencing their final syntactic parsers. Evidently, because the T-P collocation extraction relies heavily on syntactic features, the more correct syntactic parse trees derived from the compressed sentences can help to increase the performance of this task.

Compared with **no_Comp**, the **auto_Comp** system also yielded a significantly better results ($p < 0.01$) that indicated an improvement of 3.8% in the *F-score*, despite the fact that the automatic sentence compression model *Sent_Comp* may wrongly compress some sentences. This demonstrates the usefulness of sentiment sentence compression step in the T-P collocation extraction task and further proves the effectiveness of our proposed model.

[8]We use paired bootstrap resampling significance test (Efron and Tibshirani, 1993).

Moreover, we can observe that the idea of sentence compression and our *Sent_Comp* are useful for all the four product domains on T-P collocation extraction task, indicating that *Sent_Comp* is domain adaptive. However, we can find a small gap between **auto_Comp** and **manual_Comp**, which indicates that the *Sent_Comp* model can still be improved further. In the future, we will explore more effective sentence compression algorithms to bridge the gap between the two systems.

5 Related Works

5.1 Sentiment Analysis

T-P collocation extraction is a basic task in sentiment analysis. In order to solve this task, most methods focused on identifying relationships between targets and polarity words. In early studies, researchers recognized the target first, and then chose its polarity word within a window of size k (Hu and Liu, 2004). However, considering that this kind of method is too heuristic, the performance proved to be very limited. To tackle this problem, many researchers found syntactic patterns that can better describe the relationships between targets and polarity words. For example, Bloom et al. (2007) constructed a linkage specification lexicon containing 31 patterns, while Qiu et al. (2011) proposed a double propagation method that introduced eight heuristic syntactic patterns to extract the collocations. Xu et al. (2013) used the syntactic patterns to extract the collocation candidates in their two-stage framework.

Based on the above, we can conclude that syntactic features are very important for T-P collocation extraction. However, the "naturalness" problem can still seriously affect the performance of syntactic parser. Once our sentiment sentence compression method can improve the quality of parsing, the performance of T-P collocation extraction task can be improved as well. Note that, to date, there is no previous work using a sentence compression model to improve this task.

5.2 Sentence Compression

Sentence compression is a paraphrasing task aimed at generating sentences shorter than the given ones, while preserving the essential content (Jing, 2000). There are many applications that can benefit from a robust compression system, such as summarization systems (Li et al., 2013), semantic role labeling (Vickrey and Koller, 2008), relation extraction (Miwa et al., 2010) and so on.

Commonly used to compress sentences, tree-based approaches (Knight and Marcu, 2002; Turner and Charniak, 2005; Galley and McKeown, 2007; Cohn and Lapata, 2009; Galanis and Androutsopoulos, 2010; Woodsend and Lapata, 2011; Thadani and McKeown, 2013) compress a sentence by editing the syntactic tree of the original sentence. However, the automatic parsing results may not be correct; thus, the compressed tree (after removing constituents from a bad parse) may not produce a good compressed sentence. McDonald (2006), Nomoto (2007), and Clarke and Lapata (2008) tried to solve the problem by using discriminative models.

Aside from above *extractive* sentence compression approaches, there is another research line, namely, *abstractive* approach, which compresses an original sentence by reordering, substituting, and inserting, as well as removing (Cohn and Lapata, 2013). This method needs more resource and is more complicated. Therefore, in this paper, we only focus on *extractive* approach.

At present, the current sentence compression methods all focus on formal sentences, and few methods are being proposed to study sentiment sentences. As discussed in the above sections, the current compression models cannot be directly utilized to T-P collocation extraction owing to the specificity of sentiment sentences. Therefore, a new compression model for sentiment sentences should be established.

6 Conclusion and Future Work

In this work, we presented a framework that adopted a CRF based sentiment sentence compression model *Sent_Comp*, as a preprocessing step, to improve the T-P collocation extraction task. Different from the existing sentence compression models used for formal sentences, *Sent_Comp* incorporated some sentiment-related features to retain the sentiment information. Experimental results showed that the system with the sentence compression step performed better than that without this step, thus demonstrating the effectiveness of the framework and the compression model *Sent_Comp*.

Generally, the idea of this framework maybe useful for many sentiment analysis tasks that rely heavily on syntactic results. Thus in the future, we will try to apply the *Sent_Comp* model for these tasks. Besides, the simplicity and effectiveness of this framework motivates us to pursue the study further. For example, we will polish the *Sent_Comp* model by exploring more sentiment-related features and exploring other types of compression models.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. This work was supported by National Natural Science Foundation of China (NSFC) via grant 61300113, 61133012 and 61273321, the Ministry of Education Research of Social Sciences Youth funded projects via grant 12YJCZH304, the Fundamental Research Funds for the Central Universities via grant No.HIT.NSRIF.2013090 and IBM Research-China Joint Research Project.

References

Ahmed Abbasi, Hsinchun Chen, and Arab Salem. 2008. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Trans. Inf. Syst.*, 26(3):12:1–12:34, June.

Kenneth Bloom, Navendu Garg, and Shlomo Argamon. 2007. Extracting appraisal expressions. In *HLT-NAACL 2007*, pages 308–315.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.*, 18(4):467–479, December.

Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. Ltp: A chinese language technology platform. In *Coling 2010: Demonstrations*, pages 13–16, Beijing, China, August. Coling 2010 Organizing Committee.

Wanxiang Che, Mengqiu Wang, Christopher D. Manning, and Ting Liu. 2013. Named entity recognition with bilingual constraints. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 52–62, Atlanta, Georgia, June. Association for Computational Linguistics.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *J. Artif. Intell. Res. (JAIR)*, 31:399–429.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37 – 46.

Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Trevor Cohn and Mirella Lapata. 2013. An abstractive approach to sentence compression. *ACM Transactions on Intelligent Systems and Technology*, 4(3):1–35.

Adnan Duric and Fei Song. 2012. Feature selection for sentiment analysis based on content and syntax models. *Decis. Support Syst.*, 53(4):704–711, November.

B. Efron and R. J. Tibshirani. 1993. *An Introduction to the Bootstrap*. Chapman & Hall, New York.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of ACL-08: HLT*, pages 959–967, Columbus, Ohio, June. Association for Computational Linguistics.

Dimitrios Galanis and Ion Androutsopoulos. 2010. An extractive supervised two-stage method for sentence compression. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 885–893, Los Angeles, California, June. Association for Computational Linguistics.

Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 180–187, Rochester, New York, April. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of KDD-2004*, pages 168–177.

Hongyan Jing. 2000. Sentence reduction for automatic text summarization. In *IN PROCEEDINGS OF THE 6TH APPLIED NATURAL LANGUAGE PROCESSING CONFERENCE*, pages 310–315.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artif. Intell.*, 139(1):91–107, July.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio, June. Association for Computational Linguistics.

Chen Li, Fei Liu, Fuliang Weng, and Yang Liu. 2013. Document summarization via guided sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 490–500, Seattle, Washington, USA, October. Association for Computational Linguistics.

Percy Liang. 2005. Semi-supervised learning for natural language. Master's thesis, MIT.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Ryan McDonald. 2006. Discriminative sentence compression with soft syntactic evidence. In *In Proc. EACL*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 337–342, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Makoto Miwa, Rune Saetre, Yusuke Miyao, and Jun'ichi Tsujii. 2010. Entity-focused sentence simplification for relation extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 788–796.

Tadashi Nomoto. 2007. Discriminative sentence compression with conditional random fields. *Information Processing and Management*, 43(6):1571–1587, November.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37(1):9–27.

Kapil Thadani and Kathleen McKeown. 2013. Sentence compression with joint structural inference. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 65–74, Sofia, Bulgaria, August. Association for Computational Linguistics.

Jenine Turner and Eugene Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 290–297, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Vickrey and Daphne Koller. 2008. Sentence simplification for semantic role labeling. In *ACL*, pages 344–352. The Association for Computer Linguistics.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 409–420, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen, and Jun Zhao. 2013. Mining opinion words and opinion targets in a two-stage framework. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1764–1773, Sofia, Bulgaria, August. Association for Computational Linguistics.

Jun Zhao, Hongbo Xu, Xuanjing Huang, Songbo Tan, Kang Liu, and Qi Zhang. 2008. Overview of chinese pinion analysis evaluation 2008. In *The First Chinese Opinion Analysis Evaluation (COAE) 2008*.

Hybrid Grammars for Discontinuous Parsing

Mark-Jan Nederhof
School of Computer Science
University of St Andrews
KY16 9SX, UK

Heiko Vogler
Department of Computer Science
Technische Universität Dresden
D-01062 Dresden, Germany

Abstract

We introduce the concept of hybrid grammars, which are extensions of synchronous grammars, obtained by coupling of lexical elements. One part of a hybrid grammar generates linear structures, another generates hierarchical structures, and together they generate discontinuous structures. This formalizes and generalizes some existing mechanisms for dealing with discontinuous phrase structures and non-projective dependency structures. Moreover, it allows us to separate the degree of discontinuity from the time complexity of parsing.

1 Introduction

Discontinuous phrases occur frequently in languages with relatively free word order, and adequate description of their structure requires special care (Kathol and Pollard, 1995; Müller, 2004). Even for languages such as English, with a relatively rigid word order, there is a clear need for discontinuous structures (McCawley, 1982; Stucky, 1987).

Early treebanks for English (Marcus et al., 1993) have often represented discontinuity in a way that makes it tempting to ignore it altogether, certainly for the purposes of parsing, whereas recent approaches tend to represent discontinuity in a more overt form, sometimes after transformation of existing treebanks (Choi and Palmer, 2010; Evang and Kallmeyer, 2011). In many modern treebanks, discontinuous structures have been given a prominent status (Böhmová et al., 2000).

Classes of trees without discontinuity can be specified as the sets of parse trees of context-free grammars (CFGs). Somewhat larger classes can be specified by tree substitution grammars (Sima'an et al., 1994) and regular tree grammars (Brainerd, 1969; Gécseg and Steinby, 1997). Practical parsers for these three formalisms have running time $\mathcal{O}(n^3)$, where n is the length of the input sentence. Discontinuous structures go beyond their strong generative capacity however. Similarly, non-projective dependency structures cannot be obtained by traditional dependency grammars. See (Rambow, 2010) for discussion of the relation between constituent and dependency structures and see (Maier and Lichte, 2009) for a comparison of discontinuity and non-projectivity.

One way to solve the above problems has been referred to as *pseudo-projectivity*, i.e. a parser produces a projective structure, which in a second phase is transformed into a non-projective structure (Kahane et al., 1998; McDonald and Pereira, 2006; Nivre and Nilsson, 2005). In particular, this may involve *lifting*, whereby one end point of a dependency link moves across a path of nodes. A related idea for discontinuous phrase structure is the reversible splitting conversion of (Boyd, 2007). See also (Johnson, 2002; Campbell, 2004; Gabbard et al., 2006).

As shown by (Nivre, 2009), the second phase of pseudo-projective dependency parsing can be interleaved with the first, by replacing the usual one-way input tape by an additional stack, or *buffer*. Where

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1370–1381, Dublin, Ireland, August 23-29 2014.

non-topmost positions from the parsing stack are moved back to the buffer, input positions are effectively swapped and non-projective dependency structures arise.

Tree adjoining grammars (TAGs) can describe strictly larger classes of word order phenomena than CFGs (Rambow and Joshi, 1997). TAG parsers have a time complexity of $\mathcal{O}(n^6)$ (Vijay-Shankar and Joshi, 1985). However, the *derived* trees they generate are still continuous. Although their *derivation* trees may be argued to be discontinuous, these by themselves are not normally the desired syntactic structures. It was argued by (Becker et al., 1991) that further additions to TAGs are needed to obtain adequate descriptions of scrambling phenomena.

An alternative is proposed by (Kallmeyer and Kuhlmann, 2012): a transformation is added that turns a derivation tree of a (lexicalized) TAG into a non-projective dependency structure. A very similar mechanism is used to obtain non-projective dependency structures using linear context-free rewriting systems (LCFRSs) (Kuhlmann, 2013) that are lexicalized. In a LCFRS the synthesis of strings is normally specified by yield functions associated with rules. By an additional interpretation of the templates of these yield functions in the algebra of dependency trees (with the overt lexical items as roots), the LCFRS generates both strings and (possibly non-projective) dependency structures.

However, the running time of LCFRS parsers is generally very high, still polynomial in the sentence length, but with a degree determined by properties of the grammar; difficulties involved in running LCFRS parsers for natural languages are described by (Kallmeyer and Maier, 2013).

It follows from the above that there is considerable freedom in the design of parsers that produce discontinuous structures for given input sentences. One can distinguish between two main issues. The first is the formalism that guides the parsing of the input. This determines a class of input (string) languages, which can be that of the context-free languages, or tree adjoining languages, etc. We assume parsing with any of these formalisms results in derivations of some sort. The second main issue is the mechanism that translates such derivations into discontinuous structures.

This leads to a number of open questions that are all related. First, what is, or should be, the division of labor between the parser producing the derivations and the mechanism turning those derivations into discontinuous structures? If we want to achieve high degrees of discontinuity in the output structures, should the formalism for the input language be much more powerful than, say, context-free? Or can highly discontinuous structures be obtained equally well through ordinary CFGs in combination with an advanced mechanism producing discontinuous structures out of derivations?

Second, how should one approach the problem of finding the grammar (and grammar class) for the input language and the mapping from derivations to structures if the only thing that is given is a treebank? A third question is which formalisms are suitable to formally describe mappings from derivations to discontinuous structures. Lastly, can we characterize the classes of output (tree-)languages for various combinations of input grammars and derivation-to-structure mappings?

In this paper we provide one possible answer to these questions by a new type of formalism, which we call *hybrid grammars*. Such a grammar consists of a string grammar and a tree grammar. Derivations are coupled so as to achieve synchronous rewriting. The input string language and the output tree language are thereby straightforwardly defined. Different from synchronous grammars (Shieber and Schabes, 1990; Satta and Peserico, 2005) is that occurrences of terminal symbols are also coupled. Thereby the linear order of the symbols in a derived string imposes an order on the coupled symbols in the synchronously derived tree; this allows a straightforward specification of a discontinuous structure.

One can define a hybrid grammar consisting of a simple macro grammar (Fischer, 1968) and a simple context-free tree grammar (Rounds, 1970), but various other combinations of a string grammar and a tree grammar are possible as well. Due to lack of space we will here concentrate on only one kind of hybrid grammar, namely that consisting of a LCFRS as string grammar and a form of definite clause program as tree grammar. We will show that hybrid grammars that induce (finite) sets of hybrid trees can always be constructed, even if the allowable derivations are severely restricted, and we discuss experiments. Lastly, a negative result will be given, which shows that a certain linguistic phenomenon cannot be handled if the string grammar is too restricted.

We cast our definitions in terms of *hybrid trees*, of which discontinuous phrase structures and non-

projective dependency structures are special cases.[1] Thereby the generality of the framework is demonstrated.

2 Preliminaries

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ and $\mathbb{N}_+ = \mathbb{N} \setminus \{0\}$. For each $n \in \mathbb{N}_+$, we let $[n]$ stand for the set $\{1, \ldots, n\}$, and we let $[0]$ stand for $\emptyset$. We write $[n]_0$ to denote $[n] \cup \{0\}$. We fix an infinite list $x_1, x_2, \ldots$ of pairwise distinct *variables*. We let $X = \{x_1, x_2, x_3, \ldots\}$ and $X_k = \{x_1, \ldots, x_k\}$ for each $k \in \mathbb{N}$.

A *ranked set* Δ is a set of symbols, associated with a rank function assigning a number $\mathrm{rk}_\Delta(\delta) \in \mathbb{N}$ to each symbol $\delta \in \Delta$. A *ranked alphabet* is a ranked set with a finite number of symbols. We let $\Delta^{(k)}$ denote $\{\delta \in \Delta \mid \mathrm{rk}_\Delta(\delta) = k\}$.

The following definitions were inspired by (Seki and Kato, 2008). The sets of *terms* and *sequence-terms* (*s-terms*) over ranked set Δ, with variables in some set $Y \subseteq X$, are denoted by $T_\Delta(Y)$ and $T^*_\Delta(Y)$, respectively, and defined inductively as follows:
(i) $Y \subseteq T_\Delta(Y)$,
(ii) if $k \in \mathbb{N}$, $\delta \in \Delta^{(k)}$ and $s_i \in T^*_\Delta(Y)$ for each $i \in [k]$, then $\delta(s_1, \ldots, s_k) \in T_\Delta(Y)$, and
(iii) if $n \in \mathbb{N}$ and $t_i \in T_\Delta(Y)$ for each $i \in [n]$, then $\langle t_1, \ldots, t_n \rangle \in T^*_\Delta(Y)$.
We let T^*_Δ and T_Δ stand for $T^*_\Delta(\emptyset)$ and $T_\Delta(\emptyset)$ respectively. Throughout this paper, we use variables such as s and s_i for s-terms and variables such as t and t_i for terms. The justification for using s-terms as defined here is that they provide the required flexibility for dealing with both strings ($\Delta = \Delta^{(0)}$) and unranked trees ($\Delta = \Delta^{(1)}$), in combination with derivational nonterminals.

Concatenation of s-terms is given by $\langle t_1, \ldots, t_n \rangle \cdot \langle t_{n+1}, \ldots, t_{n+m} \rangle = \langle t_1, \ldots, t_{n+m} \rangle$. Sequences such as $s_1, \ldots, s_k$ or $x_1, \ldots, x_k$ will typically be abbreviated to $s_{1,k}$ or $x_{1,k}$, respectively. For $\delta \in \Delta^{(0)}$ we sometimes abbreviate $\delta()$ to δ.

In examples we also abbreviate $\langle t_1, \ldots, t_n \rangle$ to $t_1 \cdots t_n$, that is, omitting the angle brackets and commas. Moreover, we sometimes abbreviate $\delta(\langle\rangle)$ to δ. Whether δ then stands for $\delta(\langle\rangle)$ or for $\delta()$ depends on whether $\delta \in \Delta^{(1)}$ or $\delta \in \Delta^{(0)}$, which will be clear from the context.

Subterms in terms or s-terms are identified by *positions*; these can be formalized by a suitable refinement of the familiar notion of Gorn address. The set of all positions in term t or in s-term s is denoted by $\mathrm{pos}(t)$ or $\mathrm{pos}(s)$, respectively. The subset of $\mathrm{pos}(t)$ consisting of all positions where the label is in some set $\Gamma \subseteq \Delta$ is denoted by $\mathrm{pos}_\Gamma(t)$.

3 Hybrid trees

The purpose of this section is to unify existing notions of non-projective dependency structures and discontinuous phrase structures, formalized using s-terms.

We fix an alphabet $\Delta = \Delta^{(1)}$ and a subset $\Gamma \subseteq \Delta$. A *hybrid tree* over (Γ, Δ) is a pair $h = (s, \leq_s)$, where $s \in T^*_\Delta$ and $\leq_s$ is a total order on $\mathrm{pos}_\Gamma(s)$. In words, a hybrid tree combines hierarchical structure, in the form of an s-term over the full alphabet Δ, with a linear structure, which can be seen as a string over $\Gamma \subseteq \Delta$. This string will be denoted by $\mathrm{str}(h)$.

For discontinuous phrase structures, the elements of Γ would typically represent lexical items, and the elements of $\Delta \setminus \Gamma$ would typically represent syntactic categories. For non-projective dependency structures, Δ would be equal to Γ. Simple examples of discontinuous phrase structures are presented in Figures 1 and 2.

4 Basic grammatical formalisms

The concept of hybrid grammars is illustrated in Section 5, by coupling a class of string grammars and a class of tree grammars.

[1] Moreover, we need to avoid any confusion with the term "discontinuous tree" from (Bunt, 1996), which is characterized by the notion of "context daughter", which is absent from our framework. The term "hybrid tree" was used before by (Lu et al., 2008), also for a mixture of a tree structure and a linear structure, generated by a probabilistic model. However, the linear 'surface' structure was obtained by a simple left-to-right tree traversal, whereas a meaning representation was obtained by a slightly more flexible traversal of the same tree. The emphasis in the current paper is rather on separating the linear structure from the tree structure.

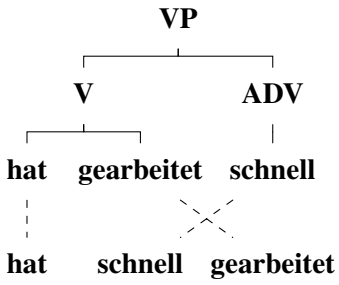


Figure 1: Hybrid tree for German "[...] hat schnell gearbeitet" ("[...] has worked quickly"), after (Seifert and Fischer, 2004). The bottom line indicates the word order in German. (Alternative analyses exist that do not require discontinuity; we make no claim the structure above is the most adequate.)

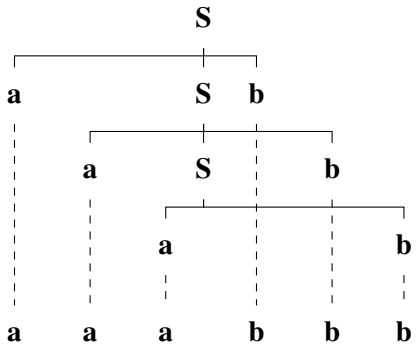


Figure 2: Abstract representation of cross-serial dependencies in Dutch (Bresnan et al., 1982).

4.1 Linear context-free rewriting systems

Much as in (Vijay-Shanker et al., 1987), we define a *linear context-free rewriting system* (LCFRS) as a tuple $G = (N, S, \Sigma, P)$, where N is a ranked alphabet of *nonterminals*, $S \in N^{(1)}$ is the *start symbol*, $\Sigma = \Sigma^{(0)}$ is a ranked alphabet of *terminals* ($\Sigma \cap N = \emptyset$), and P is a finite set of *rules*, each of the form:

$$A_0(s_{1,k_0}) \rightarrow \langle A_1(x_{1,m_1}), A_2(x_{m_1+1,m_2}), \ldots, A_n(x_{m_{n-1}+1,m_n})\rangle \tag{1}$$

where $n \in \mathbb{N}$, $A_i \in N^{(k_i)}$ for each $i \in [n]_0$, and $m_i = \sum_{j:1\leq j\leq i} k_j$ for each $i \in [n]$, and $s_j \in T^*_\Sigma(X_{m_n})$ for each $j \in [k_0]$. In words, the right-hand side is an s-term consisting of nonterminals A_i ($i \in [n]$), with distinct variables as arguments; there are m_n variables altogether, which is the sum of the ranks k_i of all A_i ($i \in [n]$). The left-hand side is an occurrence of A_0 with each argument being a string of variables and terminals. Furthermore, we demand that each x_j ($j \in [m_n]$) occurs exactly once in the left-hand side. The largest rank of any nonterminal is called the *fanout* of the grammar.

A *rule instance* is obtained by choosing a rule of the above form, and consistently substituting variables with s-terms in T^*_Σ (which are strings due to the terminals having rank 0). The language induced is the set of s-terms s such that $\langle S(s)\rangle \Rightarrow^*_G \langle\rangle$, where $\Rightarrow_G$ is the 'derives' relation that uses rule instances. For given s, the set of all LCFRS derivations $\langle S(s)\rangle \Rightarrow^*_G \langle\rangle$ (in compact tabular form) can be obtained in polynomial time in the length of s (Seki et al., 1991).

Example 1

An example of a LCFRS is presented on the right. Terminals are lower case bold letters and nonterminals are upper-case italic letters. All derived strings are of the form $\mathbf{a}^m\mathbf{c}^n\mathbf{b}^m\mathbf{d}^n$ with $m, n \in \mathbb{N}$. The linguistic relevance lies in cross-serial dependencies in Swiss German (Shieber, 1985).

$$\begin{aligned}
S(x_1x_3x_2x_4) &\rightarrow A(x_1, x_2)\ B(x_3, x_4)\\
A(\mathbf{a}x_1, \mathbf{b}x_2) &\rightarrow A(x_1, x_2)\\
A(\langle\rangle, \langle\rangle) &\rightarrow \langle\rangle\\
B(\mathbf{c}x_1, \mathbf{d}x_2) &\rightarrow B(x_1, x_2)\\
B(\langle\rangle, \langle\rangle) &\rightarrow \langle\rangle
\end{aligned}$$

4.2 Definite clause programs

In this section we describe a particular kind of definite clause programs. Our definition is inspired by (Deransart and Małuszynski, 1985), which investigated the relation between logic programs and attribute grammars, together with the "syntactic single use requirement" from (Giegerich, 1988). The values produced are trees (or to be more precise s-terms).

A *simple definite clause program* (sDCP) is a tuple $G = (N, S, \Sigma, P)$, where N is a ranked alphabet of *nonterminals* and $\Sigma = \Sigma^{(1)}$ is a ranked alphabet of *terminals*.[2] Moreover, each nonterminal $A \in N$ has a fixed i-rank (the number of *inherited* arguments) and a fixed s-rank (the number of *synthesized* arguments), denoted by i-rk(A) and s-rk(A), respectively, satisfying i-rk(A) + s-rk(A) = $\mathrm{rk}_N(A)$. In our notation, the inherited arguments precede the synthesized arguments. The *start symbol* S has only one argument, which is synthesized, i.e. $\mathrm{rk}_N(S) =$ s-rk$(S) = 1$ and i-rk$(S) = 0$.

A rule is of the form:

$$A_0(x^{(0)}_{1,k_0}, s^{(0)}_{1,k'_0}) \to \langle A_1(s^{(1)}_{1,k_1}, x^{(1)}_{1,k'_1}), \ldots, A_n(s^{(n)}_{1,k_n}, x^{(n)}_{1,k'_n})\rangle \tag{2}$$

where $n \in \mathbb{N}$, $k_i =$ i-rk(A_i) and $k'_i =$ s-rk(A_i), for $i \in [n]_0$. The set of variables occurring in the lists $x^{(0)}_{1,k_0}$ and $x^{(i)}_{1,k'_i}$ ($i \in [n]$) equals X_m, where $m = k_0 + \sum_{i\in[n]} k'_i$. In other words, every variable from X_m occurs exactly once in all these lists together. This is where values 'enter' the rule. Further, the s-terms in $s^{(0)}_{1,k'_0}$ and $s^{(i)}_{1,k_i}$ ($i \in [n]$) are in $T^*_\Sigma(X_m)$ and together contain each variable in X_m exactly once. This is where values are combined and 'exit' the rule.

The 'derives' relation $\Rightarrow_G$ and other relevant notation are defined as for LCFRSs (where the s-terms in arguments are now trees due to the terminals having rank 1). If the rules in a derivation are given, then the relevant rule instances are uniquely determined, and can be computed in linear time in the size of the derivation, provided the sDCP contains no cycles. The existence of cycles is decidable, as we know from the literature on attribute grammars. There are sufficient conditions for absence of cycles, such as the grammar being L-attributed (Bochmann, 1976). In this article, we will assume that sDCPs contain no cycles.

Example 2

An example of a sDCP is presented on the right, where the first argument of B is inherited and all other arguments are synthesized. A derived s-term is e.g. **c B**(**c B**(**a A**($\langle\rangle$) **b**) **d**) **d**.

$$\begin{aligned}
S(x_2) &\to A(x_1)\ B(x_1, x_2)\\
A(\mathbf{a}\ \mathbf{A}(x_1)\ \mathbf{b}) &\to A(x_1)\\
A(\langle\rangle) &\to \langle\rangle\\
B(x_1, \mathbf{c}\ \mathbf{B}(x_2)\ \mathbf{d}) &\to B(x_1, x_2)\\
B(x_1, x_1) &\to \langle\rangle
\end{aligned}$$

5 Hybrid grammars

We couple derivations in two grammars in a way similar to how this is commonly done for synchronous grammars, namely by *indexed* symbols. However, we apply the mechanism not only to derivational nonterminals but also to terminals.

Let Γ be a ranked alphabet. We define the ranked set $\mathcal{I}(\Gamma) = \{\gamma^{[u]} \mid \gamma \in \Gamma, u \in \mathbb{N}_+\}$, with $\mathrm{rk}_{\mathcal{I}(\Gamma)}(\gamma^{[u]}) = \mathrm{rk}_\Gamma(\gamma)$. Let Δ be another ranked alphabet ($\Delta \cap \Gamma = \emptyset$) and $Y \subseteq X$, with X as in Section 2. We let $\mathcal{I}^*_{\Gamma,\Delta}(Y)$ be the set of all s-terms $s \in T^*_{\mathcal{I}(\Gamma)\cup\Delta}(Y)$ in which each index occurs at most once.

For an s-term s, let $ind(s)$ be the set of all indices occurring in s. The deindexing function $\mathcal{D}$ removes all indices from an s-term $s \in \mathcal{I}^*_{\Gamma,\Delta}(Y)$ to obtain $\mathcal{D}(s) \in T^*_{\Gamma\cup\Delta}(Y)$. The set $\mathcal{I}_{\Gamma,\Delta}(Y) \subseteq T_{\mathcal{I}(\Gamma)\cup\Delta}(Y)$ of terms with indexed symbols is defined much as above. We let $\mathcal{I}^*_{\Gamma,\Delta} = \mathcal{I}^*_{\Gamma,\Delta}(\emptyset)$ and $\mathcal{I}_{\Gamma,\Delta} = \mathcal{I}_{\Gamma,\Delta}(\emptyset)$.

A *LCFRS/sDCP hybrid grammar* (HG) is a tuple $G = ((N_1, S_1, \Gamma), (N_2, S_2, \Sigma), P)$, subject to the following restrictions. The objects Γ and Σ are ranked alphabets with $\Gamma = \Gamma^{(0)}$ and $\Sigma = \Sigma^{(1)}$. As mere sets of symbols, we demand $\Gamma \subseteq \Sigma$ but the rank functions associated with Γ and Σ differ. Let Δ be the ranked alphabet $\Sigma \setminus \Gamma$, with $\mathrm{rk}_\Delta(\delta) = 1$ for $\delta \in \Delta$.

The *hybrid rules* in P are of the form $[\rho_1, \rho_2]$ where ρ_1 has the form in Equation (1) of an LCFRS rule except that $s_i \in \mathcal{I}^*_{\Gamma,\emptyset}(X_{m_n})$ ($i \in [k_0]$) and $A_i \in \mathcal{I}(N_1)$ ($i \in [n]$) and each index in ρ_1 occurs exactly once, and ρ_2 has the form in Equation (2) of a sDCP rule except that the s-terms in $s^{(0)}_{1,k'_0}$ and $s^{(i)}_{1,k_i}$ ($i \in [n]$) are in $\mathcal{I}^*_{\Gamma,\Delta}(X_m)$ and $A_i \in \mathcal{I}(N_2)$ ($i \in [n]$) and each index in ρ_2 occurs exactly once. We require that $ind(\rho_1) = ind(\rho_2)$ and each index either couples a pair of identical terminals or couples a pair of (possibly distinct) nonterminals.

[2]The term 'simple' here has a more restrictive meaning than the term with the same name in (Deransart and Małuszynski, 1985).

Let P_1 and P_2 be the sets of all $\mathcal{D}(\rho_1)$ and $\mathcal{D}(\rho_2)$, respectively, of some hybrid rule $[\rho_1, \rho_2]$. Then we refer to the LCFRS (N_1, S_1, Γ, P_1) and the sDCP (N_2, S_2, Σ, P_2) as the first and second *components*, respectively, of G.

In order to define the 'derives' relation $\Rightarrow_G$, we need rule instantiation as before, in combination with *reindexing*, which is a common notion for synchronous grammars. This allows specification of a set of pairs $[s_1, s_2] \in \mathcal{I}^*_{\Gamma,\emptyset} \times \mathcal{I}^*_{\Gamma,\Delta}$ which are such that $[\langle S_1^{\boxed{1}}(s_1)\rangle, \langle S_2^{\boxed{1}}(s_2)\rangle] \Rightarrow^*_G [\langle\rangle, \langle\rangle]$. For each such pair we can construct a hybrid tree $(s, \leq_s)$ over (Γ, Σ), where $s = \mathcal{D}(s_2)$, and $\leq_s$ is defined as follows. If there is a combination of positions p_1, p'_1, p_2, p'_2 such that at p_1 in s_1 we find the same label as at p_2 in s_2 (this label must then be in $\mathcal{I}(\Gamma)$), and at p'_1 in s_1 we find the same label as at p'_2 in s_2, and p_1 occurs to the left of p'_1, then $p_2 \leq_s p'_2$. The language induced by G is defined as the set of all such hybrid trees.

Given an input string, the desired hybrid trees can be effectively enumerated. To be exact, after construction of the parse table by a LCFRS parser, which takes polynomial time in the length of the string, synchronous derivations can be enumerated. Extracting a single derivation from the table requires linear time in the size of that derivation. Given a derivation, an s-term can be constructed in linear time in the size of that derivation, applying sDCP rules in the second component. This s-term, in combination with the input string and the indices linking the two is then easily extended to a hybrid tree as outlined above.

Example 3

The hybrid tree in Figure 1 is obtained by the HG on the right. (All arguments in the second component are synthesized.) We derive:

$$[\mathit{VP}(x_1x_2x_3) \rightarrow V^{\boxed{1}}(x_1, x_3)\ \mathit{ADV}^{\boxed{2}}(x_2), \mathit{VP}(\mathbf{VP}(x_1x_2)) \rightarrow V^{\boxed{1}}(x_1)\ \mathit{ADV}^{\boxed{2}}(x_2)]$$
$$[V(\mathbf{h}^{\boxed{1}}, \mathbf{g}^{\boxed{2}}) \rightarrow \langle\rangle, V(\mathbf{V}(\mathbf{h}^{\boxed{1}}\mathbf{g}^{\boxed{2}})) \rightarrow \langle\rangle]$$
$$[\mathit{ADV}(\mathbf{s}^{\boxed{1}}) \rightarrow \langle\rangle, \mathit{ADV}(\mathbf{ADV}(\mathbf{s}^{\boxed{1}})) \rightarrow \langle\rangle]$$

$$[\mathit{VP}^{\boxed{1}}(\mathbf{h}^{\boxed{2}}\mathbf{s}^{\boxed{3}}\mathbf{g}^{\boxed{4}}), \mathit{VP}^{\boxed{1}}(\mathbf{VP}(\mathbf{V}(\mathbf{h}^{\boxed{2}}\mathbf{g}^{\boxed{4}})\ \mathbf{ADV}(\mathbf{s}^{\boxed{3}})))] \Rightarrow$$
$$[V^{\boxed{1}}(\mathbf{h}^{\boxed{2}}, \mathbf{g}^{\boxed{4}})\ \mathit{ADV}^{\boxed{5}}(\mathbf{s}^{\boxed{3}}), V^{\boxed{1}}(\mathbf{V}(\mathbf{h}^{\boxed{2}}\mathbf{g}^{\boxed{4}}))\ \mathit{ADV}^{\boxed{5}}(\mathbf{ADV}(\mathbf{s}^{\boxed{3}}))] \Rightarrow$$
$$[\mathit{ADV}^{\boxed{5}}(\mathbf{s}^{\boxed{3}}),\ \mathit{ADV}^{\boxed{5}}(\mathbf{ADV}(\mathbf{s}^{\boxed{3}}))] \Rightarrow [\langle\rangle, \langle\rangle]$$

Note that in the LCFRS that is the first component of the HG above, nonterminal V has rank 2. On the right is an alternative HG deriving the same hybrid tree, but now with all LCFRS nonterminals having rank 1, by which we obtain a syntactic variant of a CFG. Yet another HG for the same hybrid tree will be discussed in the next section, where we will see that the first and second components can be disconnected even further, departing from the traditional way of LCFRS parsing.

$$[\mathit{VP}(x_1) \rightarrow V^{\boxed{1}}(x_1), \mathit{VP}(\mathbf{VP}(x_1)) \rightarrow V^{\boxed{1}}(x_1)]$$
$$[V(\mathbf{h}^{\boxed{1}}x_1\mathbf{g}^{\boxed{2}}) \rightarrow \mathit{ADV}^{\boxed{3}}(x_1), V(\mathbf{V}(\mathbf{h}^{\boxed{1}}\mathbf{g}^{\boxed{2}})\ x_1) \rightarrow \mathit{ADV}^{\boxed{3}}(x_1)]$$
$$[\mathit{ADV}(\mathbf{s}^{\boxed{1}}) \rightarrow \langle\rangle, \mathit{ADV}(\mathbf{ADV}(\mathbf{s}^{\boxed{1}})) \rightarrow \langle\rangle]$$

Example 4

Hybrid trees as in Figure 2 can be obtained by the HG on the right.

$$[A(x_1x_2) \rightarrow S^{\boxed{1}}(x_1, x_2), A(x_1) \rightarrow S^{\boxed{1}}(x_1)]$$
$$[S(\mathbf{a}^{\boxed{1}}x_1, \mathbf{b}^{\boxed{2}}x_2) \rightarrow S^{\boxed{3}}(x_1, x_2), S(\mathbf{S}(\mathbf{a}^{\boxed{1}}x_1\mathbf{b}^{\boxed{2}}) \rightarrow S^{\boxed{3}}(x_1)]$$
$$[S(\langle\rangle, \langle\rangle) \rightarrow \langle\rangle, S(\langle\rangle) \rightarrow \langle\rangle]$$

6 Grammar induction

We define a *recursive partitioning* of a string $s = \alpha_1 \cdots \alpha_n$ as a tree whose nodes are labeled with subsets of $[n]$. The root is labeled with $[n]$. Each leaf is labeled with a single element of $[n]$. Each internal node is labeled with the union of the labels of its children, which furthermore must be disjoint. We say a subset of $[n]$ has *fanout* k if k is the smallest number such that it can be written as the union of k sets of consecutive numbers.

A derivation of an LCFRS relates straightforwardly to a recursive partitioning. Consider for example the derivation of string **h s g** by the LCFRS that is the first component of the first HG in Example 3. The root would be labeled $\{1, 2, 3\}$, with children labeled $\{1, 3\}$ and $\{2\}$. The node labeled $\{1, 3\}$ has children labeled $\{1\}$ and $\{3\}$. The fanout of $\{1, 3\}$ is 2, whereas it is 1 for all other node labels. One may also extract a recursive partitioning directly from a hybrid tree, by associating each node with the set of positions of terminals that it dominates. For example, Figure 1 gives rise to the same recursive partitioning as the one mentioned above.

One central observation of this paper is that for any hybrid tree $h = (s, \leq_s)$ and any recursive partitioning of $\mathrm{str}(h)$, not necessarily extracted from h, we can construct a hybrid grammar G allowing a derivation of h, and moreover, the first (LCFRS) component of that derivation parses $\mathrm{str}(h)$ according to the given recursive partitioning. This observation holds for both dependency structures and constituent structures. The proof for dependency structures is quite technical however, and requires that the second (sDCP) component of a hybrid grammar has rules with inherited arguments. For lack of space, we can only give an outline for constituent structures, or in other words, we consider only input hybrid trees over (Γ, Δ) where labels from Γ occur exclusively at the leaves. In the resulting hybrid grammars, all sDCP rules will have only synthesized arguments.

The intuition is the following. For each node of the given recursive partitioning, the numbers in its label correspond to leaves of s, for the given hybrid tree $h = (s, \leq_s)$. There is a smallest number of maximal disjoint subtrees in s that together contain all those leaves and no others. If we now relate a parent node of the recursive partitioning to its child nodes, then we see that the relevant disjoint subtrees in s for the children can be combined to give the relevant disjoint subtrees for the parent, possibly adding further internal nodes. This process can be expressed in terms of a hybrid rule. Each pair consisting of a hybrid tree and a recursive partitioning gives rise to a number of hybrid rules. For a collection of such pairs, we can combine all the rules into a hybrid grammar.

Example 5 Consider again the hybrid tree in Figure 1, in combination with a recursive partitioning whose root has children labeled $\{1, 2\}$ and $\{3\}$. The relevant disjoint subtrees for $\{1, 2\}$ are **hat** and **ADV**(**schnell**) and for $\{3\}$ there is the subtree **gearbeitet**. (In a real-world grammar we would have parts of speech occurring above all the words.) An appropriate hybrid rule that both respects the recursive partitioning (by the first component LCFRS rule) and puts together relevant parts of the hybrid tree (by the second component sDCP rule) would be of the form:

$$[A(x_1x_2) \rightarrow B^{\boxed{1}}(x_1)\ C^{\boxed{2}}(x_2), A(\mathbf{VP}(\mathbf{V}(x_1x_3)x_2)) \rightarrow B^{\boxed{1}}(x_1, x_2)\ C^{\boxed{2}}(x_3)]$$

Here A, B and C should to be chosen to be consistent with neighboring nodes in the recursive partitioning, to be discussed next. An alternative recursive partitioning whose root has children labeled $\{1, 3\}$ and $\{2\}$ leads to the first hybrid rule in Example 3 (apart from nonterminal names).

We have experimented with two ways of naming nonterminals in the derived hybrid rules. The first encodes the list of labels of the roots of the relevant disjoint subtrees. In the above example, we would have a name such as $\langle \mathbf{hat}, \mathbf{ADV} \rangle$ for A. For fanout greater than 1, the locations of the 'gaps' are explicitly indicated. For example, we might have $\langle \mathbf{hat}, \mathit{gap}, \mathbf{gearbeitet} \rangle$. We will call this *strict* labeling. The second, and less precise, way is to replace lists of labels of siblings by a single name of the form children-of(X), where X is the label of the parent. We will call this *child* labeling.

Because our construction of hybrid grammars works for all recursive partitionings, there is no need to limit ourselves to those extracted directly from the hybrid trees. Moreover, a given recursive partitioning can be transformed into a similar but different one in which fanout is restricted to some given value $k \geq 1$. One possible procedure is to start at the root. If the label J of the present node is a singleton, then we stop. Otherwise, we search breadth-first through the subtree of the present node to identify a descendant such that both its label J' and $J \setminus J'$ have fanout not exceeding k. (It is easy to see such a node always exists: ultimately breadth-first search will reach the leaves, which are labeled with singletons.) The present node is now given two children, the first is the node labeled J' that we identified above, and the second is a copy of the present subtree, but with J' subtracted from the label of every node. (Nodes

labeled with the empty set are removed, and if a node has the same label as its parent then the two are collapsed.) We repeat the procedure for both children recursively. Note that with $k = 1$, we can induce a 'CFG/sDCP' hybrid grammar, that is, with the first component having fanout 1.

Example 6

The recursive partitioning in the left half of Figure 3 has a node labeled $\{1, 3, 6, 7\}$, with fanout 3. With $J = \{1, 2, 3, 5, 6, 7\}$ and $k = 2$, one possible choice for J' is $\{3, 7\}$, as then both J' and $J \setminus J' = \{1, 2, 5, 6\}$ have fanout not exceeding 2. This leads to the partitioning in the right half of the figure. Because now all node labels have fanout not exceeding 2, recursive traversal will make no further changes. Other valid choices for J' would be $\{2\}$ and $\{5\}$. Not a valid choice for J' would be $\{1, 6\}$, as $J \setminus \{1, 6\} = \{2, 3, 5, 7\}$, which has fanout 3.

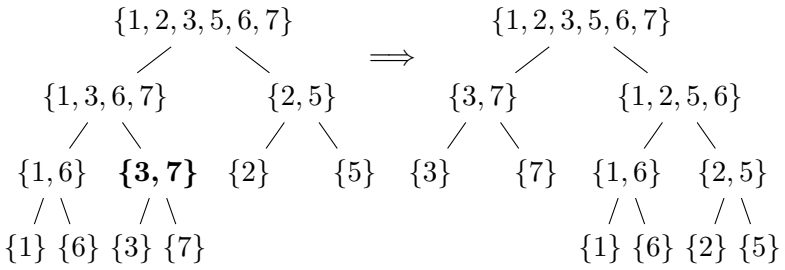


Figure 3: Transformation of recursive partitioning to restrict fanout to 2.

Our procedure ensures that subsequent grammar induction leads to binary grammars. Note that this contrasts with binarization algorithms (Gómez-Rodríguez and Satta, 2009; Gómez-Rodríguez et al., 2009) that are applied *after* a grammar is obtained. Unlike (van Cranenburgh, 2012), our objective is not to obtain a 'coarse' grammar for the purpose of coarse-to-fine parsing.

In experiments we also considered the *right-branching* partitioning, whose internal node labels are $\{m, m + 1, \ldots, n\}$, with children labeled $\{m\}$ and $\{m + 1, \ldots, n\}$. Similarly, there is a left-branching recursive partitioning. In this way, we can induce a 'FA/sDCP' hybrid grammar, with the first component having finite-state power, which means we can parse in linear time.

7 Experiments

The theory developed above shows that hybrid grammars allow considerable flexibility in the first component, leading to a wide range of different time complexities of parsing while, at least potentially, the same kinds of discontinuous structures can be obtained. We have run experiments to measure what impact different choices of the first component have on recall/precision and the degree of discontinuity.

The training data consisted of the first 7000 trees of the TIGER treebank (Brants et al., 2004). From these, recursive partitionings were straightforwardly obtained, and transformed for different values of k. Also the left-branching and right-branching recursive partitionings were considered. Hybrid grammars were then extracted using strict or child labeling. Probabilities of rules were determined by relative frequency estimation, without any smoothing techniques.

Test sentences were taken from the next 500 trees, excluding sentences of length greater than 20 and those where a single tree did not span the entire sentence, leaving 324 sentences. Parsing was on (gold standard) parts of speech rather than words. All punctuation was ignored. Labeled recall, precision and F-measure were computed on objects each consisting of the label of a node and a sequence of pairs of input positions delimiting substrings covered by that node. The algorithms were implemented in Python and the experiments were carried out on a desktop with four 3.1GHz Intel Core i5 CPUs.

Results are reported in Table 1. The choice of $k = 1$ can be seen as a baseline, the first component then being restricted to context-free power. Note that $k = 1, 2, 3$ imply parsing complexities $\mathcal{O}(n^3)$, $\mathcal{O}(n^6)$, $\mathcal{O}(n^9)$, respectively.

In the case of strict labeling, the change from $k = 1$ to $k = 2$ leads to significant changes in running time, but that from $k = 2$ to $k = 3$ less so, which can be explained by the smaller number of constituents that have two gaps, compared to those with zero or one gap. There was no significant change, neither in running time nor in F-measure, for values of k greater than 4, and therefore these values were omitted

here. Note that for $k = \infty$ one would obtain the conventional technique of discontinuous parsing using LCFRSs. For the right-branching recursive partitionings, the running time is significantly higher than that for the left-branching ones, although it is linear-time in both cases. This is due to the directional bias of the implemented parsing strategy. In order to allow a straightforward comparison we have taken the same parsing strategy in all cases. Note the large number of parse failures for the right-branching and left-branching partitionings, which is explained by the large number of very specific nonterminals.

Child labeling leads to much smaller numbers of nonterminals, and thereby also to more ambiguity, and as a result the increase from time complexity $\mathcal{O}(n^3)$ to $\mathcal{O}(n^6)$ is more noticeable in terms of the actual running time. Therefore carrying out the experiment for $k \geq 3$ was outside our reach. Surprisingly, the right-branching partitioning performed very well in this case, with a relatively low number of parse failures, F-measure competing with $k = 1, 2, 3, 4$ and strict labeling, although it is clearly worse than that with $k = 1, 2$ and child labeling, and running time smaller than in the case of any of the hybrid grammars where the first component has power beyond that of finite automata.

	fail	R	P	F1	# gaps	secs
strict labeling						
$k = 1$	16	73.0	70.4	71.2	0.0075	442
$k = 2$	12	73.1	70.7	71.4	0.0111	2,580
$k = 3$	12	73.1	70.7	71.4	0.0121	2,942
$k = 4$	12	73.1	70.7	71.4	0.0127	2,828
r-branch	151	65.6	62.4	63.2	0.0118	775
l-branch	266	82.0	78.9	79.5	0.0124	24
child labeling						
$k = 1$	4	74.3	74.2	73.9	0.0120	939
$k = 2$	4	75.0	75.1	74.7	0.0125	58,164
r-branch	15	73.1	73.0	72.6	0.0117	319
l-branch	56	75.7	76.6	75.7	0.0114	183

Table 1: Number of parse failures, recall, precision, F-measure, average number of gaps per constituent, and running time.

Child labeling generally gave better F-measure than strict labeling (ignoring strict labeling and left-branching partitioning, where the many parse failures distort the recall and precision). This seems to be due to the more accurate parameter estimation that was possible for the smaller numbers of rules obtained with child labeling.

The differences in F-measure are relatively small for varying k. This can be explained by the relatively small portion of discontinuous structures in the test set. We have looked closer at discontinuity in the test set in two ways. First, we measured the average number of gaps per constituent, which in the gold standard was 0.0171. None of the hybrid grammars came close to achieving this, but we do observe that more discontinuity is obtained for higher values of k. Secondly, we reran the experiments for only the 75 sentences out of the aforementioned 324 where the gold structure had at least one discontinuous phrase. For this smaller set, F1 increases from 59.5 ($k = 1$) to 61.9 ($k = 2, 3, 4$) for strict labeling, and it increases from 64.4 ($k = 1$) to 66.5 ($k = 2$) for child labeling. This suggests that with higher k, the additional discontinuous structures found have at least some overlap with those of the gold standard. Note again that there is no a priori bound on the fanout of produced hybrid trees, even when the first component has finite-state power, but the ability to abstract away from discontinuous structures in the training set seems to be enhanced if the first component is more powerful. This is consistent with observations made by (van Cranenburgh, 2012).

8 Limitations

The theory from Section 6 does not necessarily mean that any language of hybrid trees can be induced by a HG whose first-component LCFRS has arbitrarily low fanout. We illustrate this by means of the language of hybrid trees generated by the HG of Example 4, in which the LCFRS has fanout 2. No CFG/sDCP grammar in fact exists for the same language, or in other words, the fanout of the first-component LCFRS cannot be reduced to 1, regardless of how we choose the second-component sDCP.

For a proof, assume that a CFG/sDCP grammar does exist. Let m be the maximum number of members in the right-hand side of any CFG rule. Let k be the maximum rank of any nonterminal in the second-component sDCP. Now consider a CFG/sDCP derivation for a hybrid tree with yield $a^n b^n$, where $n \geq$

$2 \cdot k \cdot m$. In a top-down traversal, identify the first CFG nonterminal occurrence that covers a substring of the input string that has a length smaller than or equal to $n/2$ and greater than k. This substring may contain occurrences of a and of b, but because its length is at most $n/2$, there will not be any pair consisting of an occurrence of a and an occurrence of b that are both part of that substring, and that have a common parent labeled S in the hybrid tree. This means that more than k tree fragments or tree nodes with missing child nodes are involved, which translate to more than k synthesized or inherited arguments, contradicting the assumptions.

9 Conclusions

We have presented hybrid grammars as a novel framework for describing languages of discontinuous syntactic structures. This framework sheds light on the relation between various existing techniques, but it also offers potential for development of novel techniques. Much of what we have shown is merely an illustration of particular instances of this framework. For example, next to the hybrid grammars discussed here, we can consider those with macro grammars as first component, or simple context-free tree grammars as second component. Many variations exist on the illustrated grammar induction technique. For example, next to our strict labeling and child labeling, one can consider approaches using latent variables, combined with expectation-maximization.

Acknowledgments

We thank the anonymous reviewers for many helpful comments.

References

T. Becker, A.K. Joshi, and O. Rambow. 1991. Long-distance scrambling and Tree Adjoining Grammars. In *Fifth EACL*, pages 21–26.

G.V. Bochmann. 1976. Semantic evaluation from left to right. *Communications of the ACM*, 19(2):55–62.

A. Böhmová, J. Hajič, E. Hajičová, and B. Hladká. 2000. The Prague dependency treebank: A tree-level annotation scenario. In A. Abeillé, editor, *Treebanks: Building and using syntactically annotated corpora*, pages 103–127. Kluwer, Dordrecht.

A. Boyd. 2007. Discontinuity revisited: An improved conversion to context-free representations. In *Linguistic Annotation Workshop, at ACL 2007*, pages 41–44.

W.S. Brainerd. 1969. Tree generating regular systems. *Information and Control*, 14:217–231.

S. Brants, S. Dipper, P. Eisenberg, S. Hansen-Schirra, E. König, W. Lezius, C. Rohrer, G. Smith, and H. Uszkoreit. 2004. TIGER: Linguistic interpretation of a German corpus. *Research on Language and Computation*, 2:597–620.

J. Bresnan, R.M. Kaplan, S. Peters, and A. Zaenen. 1982. Cross-serial dependencies in Dutch. *Linguistic Inquiry*, 13(4):613–635.

H. Bunt. 1996. Formal tools for describing and processing discontinuous constituency structure. In H. Bunt and A. van Horck, editors, *Discontinuous Constituency*, pages 63–84. Mouton de Gruyter.

R. Campbell. 2004. Using linguistic principles to recover empty categories. In *42nd Annual Meeting of the ACL*, pages 645–652.

J.D. Choi and M. Palmer. 2010. Robust constituent-to-dependency conversion for English. In *Ninth International Workshop on Treebanks and Linguistic Theories*, pages 55–66.

P. Deransart and J. Małuszynski. 1985. Relating logic programs and attribute grammars. *Journal of Logic Programming*, 2:119–155.

K. Evang and L. Kallmeyer. 2011. PLCFRS parsing of English discontinuous constituents. In *12th International Conference on Parsing Technologies*, pages 104–116.

M.J. Fischer. 1968. Grammars with macro-like productions. In *IEEE Conference Record of 9th Annual Symposium on Switching and Automata Theory*, pages 131–142.

R. Gabbard, S. Kulick, and M. Marcus. 2006. Fully parsing the Penn Treebank. In *Human Language Technology Conference of the NAACL, Main Conference*, pages 184–191.

F. Gécseg and M. Steinby. 1997. Tree languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, Vol. 3*, chapter 1, pages 1–68. Springer, Berlin.

R. Giegerich. 1988. Composition and evaluation of attribute coupled grammars. *Acta Informatica*, 25:355–423.

C. Gómez-Rodríguez and G. Satta. 2009. An optimal-time binarization algorithm for linear context-free rewriting systems with fan-out two. In *47th ACL and 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 985–993.

C. Gómez-Rodríguez, M. Kuhlmann, G. Satta, and D. Weir. 2009. Optimal reduction of rule length in linear context-free rewriting systems. In *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL*, pages 539–547.

M. Johnson. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *40th ACL*, pages 136–143.

S. Kahane, A. Nasr, and O. Rambow. 1998. Pseudo-projectivity, a polynomially parsable non-projective dependency grammar. In *36th ACL and 17th International Conference on Computational Linguistics*, volume 1, pages 646–652.

K. Kallmeyer and M. Kuhlmann. 2012. A formal model for plausible dependencies in lexicalized tree adjoining grammar. In *Eleventh International Workshop on Tree Adjoining Grammar and Related Formalisms*, pages 108–116.

L. Kallmeyer and W. Maier. 2013. Data-driven parsing using probabilistic linear context-free rewriting systems. *Computational Linguistics*, 39(1):87–119.

A. Kathol and C. Pollard. 1995. Extraposition via complex domain formation. In *33rd ACL*, pages 174–180.

M. Kuhlmann. 2013. Mildly non-projective dependency grammar. *Computational Linguistics*, 39(2):355–387.

W. Lu, H.T. Ng, W.S. Lee, and L.S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *Conference on Empirical Methods in Natural Language Processing*, pages 783–792.

W. Maier and T. Lichte. 2009. Characterizing discontinuity in constituent treebanks. In P. de Groote, M. Egg, and L. Kallmeyer, editors, *14th Conference on Formal Grammar*, volume 5591 of *Lecture Notes in Artificial Intelligence*, Bordeaux, France.

M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

J.D. McCawley. 1982. Parentheticals and discontinuous constituent structure. *Linguistic Inquiry*, 13(1):91–106.

R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *11th EACL*, pages 81–88.

S. Müller. 2004. Continuous or discontinuous constituents? a comparison between syntactic analyses for constituent order and their processing systems. *Research on Language and Computation*, 2:209–257.

J. Nivre and J. Nilsson. 2005. Pseudo-projective dependency parsing. In *43rd ACL*, pages 99–106.

J. Nivre. 2009. Non-projective dependency parsing in expected linear time. In *Joint Conference of the 47th ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 351–359.

O. Rambow and A.K. Joshi. 1997. A formal look at dependency grammars and phrase structure grammars with special consideration of word-order phenomena. In L. Wenner, editor, *Recent Trends in Meaning-Text Theory*. John Benjamin.

O. Rambow. 2010. The simple truth about dependency and phrase structure representations: An opinion piece. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, Main Conference*, pages 337–340.

W.C. Rounds. 1970. Mappings and grammars on trees. *Mathematical Systems Theory*, 4:257–287.

G. Satta and E. Peserico. 2005. Some computational complexity results for synchronous context-free grammars. In *Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 803–810.

S. Seifert and I. Fischer. 2004. Parsing string generating hypergraph grammars. In H. Ehrig, G. Engels, F. Parisi-Presicce, and G. Rozenberg, editors, *2nd International Conference on Graph Transformations*, volume 3256 of *Lecture Notes in Computer Science*, pages 352–267. Springer-Verlag.

H. Seki and Y. Kato. 2008. On the generative power of multiple context-free grammars and macro grammars. *IEICE Transactions on Information and Systems*, E91-D:209–221.

H. Seki, T. Matsumura, M. Fujii, and T. Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88:191–229.

S.M. Shieber and Y. Schabes. 1990. Synchronous tree-adjoining grammars. In *Papers presented to the 13th International Conference on Computational Linguistics*, volume 3, pages 253–258.

S.M. Shieber. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8(3):333–343.

K. Sima'an, R. Bod, S. Krauwer, and R. Scha. 1994. Efficient disambiguation by means of stochastic tree substitution grammars. In *International Conference on New Methods in Language Processing*, pages 50–58.

S. Stucky. 1987. Configurational variation in English. In G.J. Huck and A.E. Ojeda, editors, *Discontinuous Constituency*, volume 20 of *Syntax and Semantics*, pages 377–404. Academic Press.

A. van Cranenburgh. 2012. Efficient parsing with linear context-free rewriting systems. In *13th EACL*, pages 460–470.

K. Vijay-Shankar and A.K. Joshi. 1985. Some computational properties of tree adjoining grammars. In *23rd ACL*, pages 82–93.

K. Vijay-Shanker, D.J. Weir, and A.K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *25th ACL*, pages 104–111.

From neighborhood to parenthood: the advantages of dependency representation over bigrams in Brown clustering

Simon Šuster
University of Groningen
Netherlands
s.suster@rug.nl

Gertjan van Noord
University of Groningen
Netherlands
g.j.m.van.noord@rug.nl

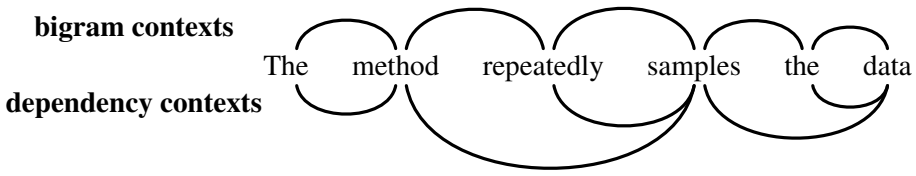
Abstract

We present an effective modification of the popular Brown et al. 1992 word clustering algorithm, using a dependency language model. By leveraging syntax-based context, resulting clusters are better when evaluated against a wordnet for Dutch. The improvements are stable across parameters such as number of clusters, minimum frequency and granularity. Further refinement is possible through dependency relation selection. Our approach achieves a desired clustering quality with less data, resulting in a decrease in cluster creation times.

1 Introduction

Semi-supervised approaches have been successful in various areas of natural language processing. Among a plethora of clustering techniques, Brown clustering (Brown et al., 1992) is popular for its conceptual simplicity, available implementations (Liang, 2005; Stolcke, 2002), and because the resulting word clusters can be helpful for several tasks. Clusters are used as syntactic and semantic *generalizations* of words, requiring fewer model parameters.

Brown clustering (section 2) groups words based on shared context. However, only immediately adjacent words are taken into account as recognized e.g. by Koo et al. (2008), Sagae and Gordon (2009), and Grave et al. (2013). For example, even though verbs constitute an informative context for object nouns, they are rarely considered in Brown clustering, unlike in dependency-based clustering. The difference between the contexts can be illustrated with the following example:



The bigram context thus fails to capture the relation between the object *data* and the predicate *samples*, as well as the one between the subject *method* and the predicate. Furthermore, the dependency representation rightly ignores some of the less informative contexts coming from immediately adjacent words. For example, there is no relation between the predicate *samples* and the article *the* to the right.

It might be preferable therefore to induce word clusters based on the dependency relations in which the words occur. In section 3, we present how this relates to Brown clustering, and we modify the code by Percy Liang, so that dependency clustering can be used. We evaluate clusters in a wordnet-based similarity experiment. Dependency clustering yields superior clusters for Dutch across different settings of parameters such as number of clusters, frequency threshold and level of granularity. Selecting specific dependency relation labels and using data obtained from them as input to clustering further improves the clustering quality. The proposed adaptation of Brown clustering does not change the complexity of the

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1382–1391, Dublin, Ireland, August 23-29 2014.

algorithm, and—although we assume that syntactically parsed text is available—it requires much less data for a desired level of clustering quality.

2 The Brown clustering algorithm

Brown clustering (Brown et al., 1992) is an agglomerative algorithm that induces a hierarchical clustering of words. It takes a tokenized corpus and groups words into k clusters identified by bit strings, representing paths in the induced binary tree in which the leaves are word clusters. Prefixes of the paths can be used to achieve clusters of coarser granularity (Sun et al., 2011; Turian et al., 2010). The obtained clusters contain words that are semantically related, or are paradigmatic or orthographic variants.[1]

The algorithm starts by putting k most frequent words into distinct clusters. Then, the k+1$^{\text{th}}$ most frequent word is assigned to a new cluster, and two among the resulting k+1 clusters are merged, i.e. the pair that maximizes the average mutual information of the current clustering. This process is repeated until all words have been merged. The resulting k clusters are then merged to build the binary tree. The version of the algorithm optimized for speed runs in $O(k^2|\mathcal{V}|)$, with $|\mathcal{V}|$ the vocabulary size.

Brown clustering has been used extensively in supervised NLP tasks such as parsing (Koo et al., 2008; Candito and Crabbé, 2009; Haffari et al., 2011), named-entity recognition (NER) and chunking (Turian et al., 2010), sentiment analysis (Popat et al., 2013), relation extraction (Plank and Moschitti, 2013), unsupervised semantic role labeling (Titov and Klementiev, 2012), question answering (Momtazi et al., 2010), POS tagging (Owoputi et al., 2013) and speech recognition with recursive neural networks (Shi et al., 2013). Recently, multilingual clustering has also been proposed (Täckström et al., 2012; Faruqui and Dyer, 2013).

Among the most frequently recognized limitations (cf. Koo et al. (2008); Chrupala (2011)) are a) the hard nature of the clustering, b) relatively long running time[2] and c) insensitivity to wider context. Our method attempts to overcome the final disadvantage. As it requires less data, it also reduces the running time.

Leveraging syntactic context for word representations has been explored, among others, in Lin (1998) on distributional thesauri; Haffari et al. (2011) on combining Brown clusters and word groupings from split non-terminals; Sagae and Gordon (2009) on using unlexicalized syntactic context in hierarchical clustering; Van de Cruys (2010) and Padó and Lapata (2007) on comparison of window- and syntactic-based word space models; and Boyd-Graber and Blei (2008) on syntactic topic models.

The work closest to ours is that of Grave et al. (2013). The authors show that clusters obtained from dependency trees outperform standard Brown clustering when used as features in super-sense tagging and NER. Their focus is on a generalization of Brown clustering with Hidden Markov models (extending Markov chains to trees), allowing the creation of soft clusters.[3] Learning and inference are done with online expectation-maximization and belief propagation.

Whereas Grave et al. focus on new learning methods for clustering with HMMs on dependency trees, we take an in-depth look at parameters and choices that are standardly considered using the (Brown et al., 1992) algorithm. We show that the advantage of dependency clustering can be observed throughout different parametrizations of cluster capacity, granularity level, frequency thresholding and other criteria (section 6), and that the advantage is roughly constant for varying amounts of input data. Finally, we provide new insight in the advantage of selective dependency clustering, in which the data obtained only from specific dependency relations lead to better clusters. Our approach constitutes a straightforward extension of Brown clustering, and only required a simple modification of the Brown clustering code.

[1] We are using the term *semantic relatedness* in its broadest possible scope. Words or clusters are semantically related when they have any kind of semantic relation: synonymy, meronymy, antonymy, hypernymy etc. (Turney and Pantel, 2010).

[2] Although coarser clustering (k<1000) can mean more practical running times, as the clustering depends quadratically on k.

[3] This approach allows to capture homonymy/polysemy, with the idea that when a word representation is needed, it can be obtained in a context-sensitive way (Huang et al., 2011; Nepal and Yates, 2014). This is certainly an important advantage over Brown clustering in which the mapping between a word and a cluster is deterministic; however, it comes with its own disadvantages: creating context-sensitive representations requires (potentially) costly inference; furthermore, HMM-based clustering does not build nor lends itself easily to a hierarchy, which is often exploited during feature creation in supervised learning to control cluster granularity (see the end of section 5.2)

3 Extension of the Brown clustering

The bigram language model underlying Brown clustering takes the probability of a sentence as the product of probabilities of words based on immediately preceding words. In contrast, we replace this by a *dependency* language model (DLM), which defines the probability of a sentence over dependency trees (Shen et al., 2008). This probability can be factorized in different ways (Chen et al., 2012; Charniak, 2001; Popel and Mareček, 2010), but the common idea is that a word is conditioned on some history, where the link between the two is a dependency. In practice, the history can include the immediate parent of the word, which can be either a lexical head or the artificial root node, as well as siblings between the child and the parent. Our take on DLM is similar to Charniak (2001) and Popel and Mareček (2010): the probability of a word is conditioned simply on its parent. This is the same view as taken by Grave et al. (2013).

The Brown clustering objective is to find such a deterministic clustering function $\mathcal{C}$ mapping each word from the vocabulary $\mathcal{V}$ to one of K clusters that maximizes the likelihood of the data. The likelihood of a sequence of word tokens, $\mathbf{w} = \langle w_i \rangle_{i=1}^{m}$, with each $w_i \in \mathcal{V}$, factors as

$$L(\mathbf{w}; \mathcal{C}) = \prod_{i=1}^{m} p(\mathcal{C}(w_i)|\mathcal{C}(w_{i-1}))p(w_i|\mathcal{C}(w_i)), \tag{3.1}$$

where $\mathcal{C}(w_0)$ is a special start-of-sequence symbol. As shown by Brown et al. (1992), by taking the negative logarithm and using the ML estimates, the equation 3.1 is decomposed to the negative entropy of the sequence $\mathbf{w}$ and mutual information between adjacent clusters. Since the entropy is independent of the clustering function, the objective amounts to finding such $\mathcal{C}$ that maximizes the mutual information.

For *dependency clustering*, we change the cluster transition probability so that conditioning is on the cluster of the parent of the word at position i, instead of on the cluster of the previous word:

$$L'(\mathbf{w}; \mathcal{C}) = \prod_{i=1}^{m} p(\mathcal{C}(w_i)|\mathcal{C}(w_{\pi(i)}))p(w_i|\mathcal{C}(w_i)), \tag{3.2}$$

where i ranges over all children in a tree and π is a function from the children to their unique parents (which include the special root of the tree). Calculation of the mutual information changes only to the extent that count tables no longer represent adjacency relationship (bigrams) between words but parenthood (child–parent relation).

4 Evaluation task

We evaluate our word clusters by following the method of Van de Cruys (2010) for evaluating vector space models. The method is based on a wordnet for Dutch and assumes that two semantically related words also occur close to each other in the wordnet hierarchy.[4] We use Cornetto (Vossen et al., 2013), which includes more than 92,000 form-POS pairs described in terms of lexical units, synsets and other criteria. For calculating similarity scores, we treat Cornetto as a digraph, with nodes constituting synsets and arcs constituting hypernymic relations, and adopt the Lin similarity measure (Lin, 1998)[5] in combination with the ontological variant of Information Content[6].

Evaluation is guided by a list of 10,000 most frequent words from SoNaR, a 500M-word reference corpus for Dutch.[7] Every word is compared to other words in the same cluster, and the average similarity for all comparisons is taken as the final score. The described method is well suited for measuring intracluster quality, yet useful information about word similarity is available also by looking at neighboring

[4]For English, several semantic similarity datasets are available (such as WordSimilarity-353 (Finkelstein et al., 2001)), some of which can identify the type of relatedness captured. We are not aware of such datasets for Dutch.

[5]Which is a function of the IC of the least common subsumer of two synsets and the IC of individual synsets. The score ranges between 0 and 1.

[6]Which is the negative logarithm of $(|\mathcal{L}| + 1)^{-1}((|\mathcal{L}_s|/|\mathcal{S}_s|) + 1)$, where $\mathcal{L}$ are the leaves of the hierarchy, $\mathcal{L}_s$ are the leaves reachable from a synset s, and $\mathcal{S}_s$ are the subsumers of s (Sánchez et al., 2011).

[7]`http://lands.let.ru.nl/projects/SoNaR`

clusters in the binary tree. This *inter*cluster quality, according to which clusters that are close in the binary tree are more similar than clusters that are far apart, can be captured indirectly by evaluating using different bit substrings. In this way, when a substring is used, two or more semantically related, but isolated clusters are merged, which should result in a drop in clustering quality (semantic relatedness tends to "dissolve" when merging).

For both standard and dependency Brown clustering, the same set of sentences is used. From SoNaR, we sampled sentences amounting to roughly 46M words, which is comparable to the count for English datasets of Koo et al. (2008) and Turian et al. (2010). The sentence length was restricted to five or more words to exclude noisy text. Corpus annotation was removed.

For dependency clustering, the dataset was lemmatized and parsed with the Alpino parser (Van Noord, 2006), an HPSG parser with a maxent disambiguation component, achieving labeled dependency accuracy of around 90.5 for Dutch.[8] The parsing accuracy is likely to be lower on our dataset, but we expect this effect to be small since Alpino has been shown to be relatively insensitive to domain shifts compared to some entirely data-driven parsers (Plank and van Noord, 2010). For default clustering, we only use first-order dependencies produced by the parser. The bilexical counts (head and dependent regardless of the relation label) serve as input for dependency clustering.

5 Experiments and Results

The main parameter for word clustering is the number of clusters k, which we set to either 1000 or 3200,[9] except when measuring clustering capacity, for which smaller values of k are used. Additionally, we limit the minimum frequency of words in clustering to three, unless stated otherwise. The vocabulary size for k=1000 clustering with applied frequency threshold is around 237,000. We use a paired t-test to check for statistical significance of observed differences in means.

5.1 Cluster examples

In Table 1, we show both the versatility of dependency clusters by dividing the examples in five groups (A–E), and the similarity of clusters within group. The longer the common bit substring between clusters, the closer they are in the hierarchy. Group **A** includes words describing professions or people's roles and functions. Group **B** lists personal pronouns, including reflexive pronouns (B2), where substantial differentiation exists with many singleton clusters. Clusters are capable of grouping orthographic variants (D1; *email* and *e-mail*) and diminutives (*sms_DIM*, corresponding to Dutch *smsje*). Because first and last names are extremely common in our corpus, clustering creates fine-grained distinctions between these (**C**). C1 groups names of presidents, whereas C2 and C3 distinguish between feminine and masculine names. Measurable concepts are included in **E**.

5.2 Cluster quality

Table 2 presents the general quality of standard and dependency clustering. The results for 1000 and 3200 clusters (in the latter we use a higher frequency threshold for faster computation) show that we obtain a higher similarity score for 3200 clusters compared to 1000, and a more marked difference between standard and dependency clustering in the case of k=3200 (Δ=0.019). We also looked at how many words from the frequency list were evaluated successfully. The recall depends on the success of mapping between words and synsets as well as the success of finding the word in one of the clusters. The latter factor influences the recall to a much lesser degree, as almost all words are found in the clustering. For 3200 clusters with the minimum frequency set to fifty, approximately 5000 words are successfully evaluated, whereas for 1000 clusters, this number is around 7000.[10] These numbers are not affected by the type of clustering (standard or dependency).

[8] Strictly speaking, the output of lemmatization is root forms. We perform this preprocessing step to increase the number of times that a word is successfully matched in the wordnet hierarchy and evaluated.

[9] Which are standardly encountered throughout the literature. For k above 3200, the algorithm falls short of practicality on current hardware assuming a single-core implementation.

[10] The difference between the figures occurs because of a different frequency threshold.

Group	Cluster id	Most frequent words	Left
A1	001010001011100	aannemer, huis_arts, bakker, notaris, apotheker, makelaar *contractor*, *family doctor*, *baker*, *lawyer*, *pharmacist*, *estate agent*	+57
A2	001010001011011	analist, criticus, waarnemer, kenner, commentator, mens_recht_organisatie *analyst*, *reviewer*, *observer*, *expert*, *commentator*, *human rights organization*	+8
A3	0010100010111110	ondernemer, zakenman, bedrijf_leider, zelfstandige, koopman, starter *entrepreneur*, *businessman*, *manager*, *self-employed*, *merchant*, *starter*	+18
B1	011101111011110	mij *me*	0
B2	01110111101110	zichzelf, mezelf, jezelf, onszelf, mijzelf, uzelf *him/herself*, *myself*, *yourself*, *ourselves*, *myself*, *yourself*	0
B3	01110111101101	hem *him*	0
B4	01110111101100	hen *them*	0
C1	0011010010	Bush, Obama, Clinton, Poetin, Chirac, Sarkozy *Bush*, *Obama*, *Clinton*, *Putin*, *Chirac*, *Sarkozy*	+95
C2	0011000111010	Sarah, Kim, Nathalie, Justine, Kirsten, Tia, Eline	+12
C3	0011000111011	David, Jimmy, Benjamin, Samuel, Tommy, Sean	+98
D1	001011100010101	email, mail, sms, sms_DIM, e-mail, mail_DIM	+13
D2	001011100010100	telefoon, satelliet, telefonie, telefoon_lijn, Explorer, muziek_speler, iTunes *telephone*, *satellite*, *telephony*, *telephone line*, *Explorer*, *music player*, *iTunes*	+7
E	001000010110101	inkomen, energie_verbruik, minimum_loon, cholesterol, opleidingsniveau, *income*, *energy consumption*, *minimum wage*, *cholesterol*, *level of education*, IQ, alcohol_gehalte *IQ*, *alcohol content*	+32

Table 1: Example dependency clusters obtained from a run with number of clusters set to 3200 and minimum frequency to 50. The underlined part of the bit string indicates the longest common substring within one group. English translation of the Dutch original is given in italics and is left out when clear from the original. Column *Left* indicates the remaining number of (less frequent) words in the cluster.

k	Brown	DepBrown	Δ
1000	0.191	0.196	+.005*
3200	0.279	0.298	+.019**

Table 2: Lin similarity scores for standard *Brown* clustering and dependency Brown clustering (*DepBrown*), with k the number of clusters. Δ=DepBrown – Brown. Frequency threshold of 50 is used for clustering with $k = 3200$. *: statistically significant with $p < 0.05$, **: statistically significant with $p < 0.001$.

Results for four different clustering parametrizations are shown in Table 3. One way of controlling the granularity is to choose the number of output clusters k. As shown in the table under CAP ("capacity"), dependency clustering achieves a better quality regardless of the choice of k, and in general, choosing a smaller k decreases quality, which is compatible with the observations of Turian et al. (2010) in their chunking experiments.

An effect similar to that of controlling capacity can be achieved by making use of the fact that the induced structure is a hierarchy.[11] By choosing a path prefix length that is shorter than the maximum length, we control the cluster granularity (denoted in the table as PREF-*). For different tasks, different path prefixes might be appropriate (Sun et al., 2011; Koo et al., 2008; Miller et al., 2004). For example, one might prefer coarser distinctions (i.e. shorter bit strings) in parsing, while finer granularity might be necessary to obtain effective representations of proper names in NER. We ran the experiment with prefix length ranging from one to eighteen, and show a selection of four settings in the table. Across the board, dependency clustering yields better results than standard clustering. Naturally, with shorter prefixes the quality decreases, which is explained by increasing word population in the clusters, with more and more

[11]The parameter k needs to be chosen before clustering, whereas the hierarchical structure can be exploited during feature preparation based on already existing clusters.

Setting	k	*min*	Brown	DepBrown	Δ
CAP	200	10	0.148	0.157	+.009
	400	10	0.169	0.175	+.006
	600	10	0.182	0.191	+.009
	800	10	0.191	0.205	+.014
PREF-16	1000	10	0.2	0.215	+.015
PREF-12	1000	10	0.187	0.202	+.015
PREF-8	1000	10	0.159	0.168	+.009
PREF-4	1000	10	0.114	0.127	+.013
FREQ	1000	5	0.196	0.204	+.008
	1000	10	0.202	0.216	+.014
	1000	20	0.206	0.221	+.015
	1000	30	0.209	0.224	+.015
	1000	50	0.216	0.227	+.011
NOUNS	1000	3	0.272	0.279	+.007

Table 3: Lin similarity scores for standard *Brown* clustering and dependency Brown clustering (*DepBrown*), with *k* the number of clusters, *min* the minimum frequency of words. CAP: varying k, fixed min; FREQ: varying min, fixed k; NOUNS: evaluating only nouns, PREF-n: size of bit-string prefix, Δ=DepBrown – Brown. All the results reported for *DepBrown* are significantly different from *Brown* with $p < 0.001$.

distant (both hierarchically and semantically) clusters being merged.

By inspecting individual clusters, we observe that frequent words in a cluster exhibit clear semantic relatedness, but that rare words are often semantically quite unrelated.[12] This is confirmed by our results in which the quality of the clustering improves approximately logarithmically with frequency threshold increasing (FREQ). The margin between standard and dependency clustering is also increasing as we increase the threshold. In practice, Brown clusters appear to be equally useful with a high frequency threshold (Owoputi et al., 2013) as without thresholding (Koo et al., 2008; Turian et al., 2010).

We also investigate the quality of nouns only, to facilitate the comparison to Van de Cruys (2010). We observe a considerable gain in quality when only nouns are used compared to using all parts of speech — the Lin score is increased by 0.08. In the noun-only evaluation, dependency clustering achieves a higher score (0.279) than standard clustering (0.272). Van de Cruys (2010) shows that syntactic vector space models outperform window-based models, which is confirmed by our finding for word clustering as well. In his work, syntactic vector space models yield a 0.04 advantage in Lin score, whereas our dependency clusters achieve a less marked advantage, reaching up to 0.019 in Lin score. A possible explanation for this difference is that in his evaluation an average over only five most similar nouns is taken, whereas we impose no such restriction. We would like to point out that our work does not aim to compare and discuss the merits of clustering and vector space models as possible techniques for obtaining word representations, but rather to provide a comprehensive comparison of standard Brown clustering and its dependency extension.

5.3 Learning curves

Figure 5.2 shows the amount of data needed to achieve a certain quality of clustering. For clustering on ten thousand sentences the similarity score is around 0.14, with a higher score for standard clustering. For each subsequent addition of data, dependency clustering outperforms standard clustering. In order to achieve the highest score attained by standard clustering (0.19), resulting from clustering on 2.4 million sentences (41 million words), dependency clustering requires only slightly more than 500 thousand sentences (8.5 million words). This observation is advantageous especially because less data means

[12] Although cf. Turian et al. (2010) who show that Brown clustering has a superior representation for rare words than neural word embeddings in their experiment.

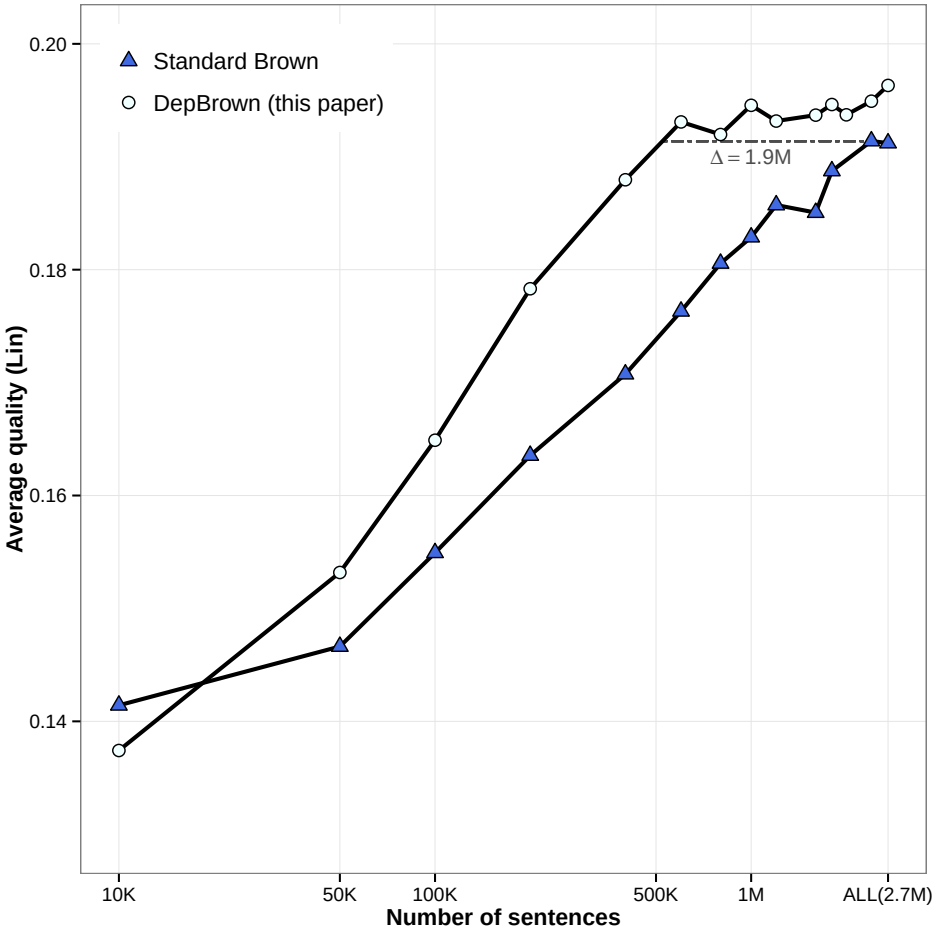


Figure 1: Learning curves for standard and dependency Brown clustering with 1000 clusters and a frequency threshold of 3. Dashed line displays the difference in amount of data needed for *DepBrown* to achieve the best quality of *Brown*. Using all, 2.7 million sentences from the corpus (*ALL*) corresponds to 46 million words.

shorter running time for clustering as the number of word types is reduced.

5.4 Refinement of dependency clusters

Our dependency clustering described in the previous sections operates on words appearing in all dependency relations. We now investigate whether selecting only a particular dependency relation—i.e. using as the input both parent and child words from that dependency relation—leads to clusters with higher semantic relatedness. Each relation can be characterized as either a first- or a second-order relation.[13] A second-order relation is between two words with an intervening preposition, e.g. between a verb and a noun of a directional complement introduced by a preposition, such as in the Dutch "eten achter pc" ("eating at the computer").[14] We ran clustering for each of the forty-five dependency relations separately and measured the quality of each resulting clustering. The cumulative baseline that does not distinguish between dependency relations is given as ALL for first-order relations in Table 4. This is the same result as reported on the first line in Table 2. The addition of second-order dependencies does not change the clustering quality of the baseline (0.196) but increases the number of types.

In the upper part of Table 4, we list six relations leading to clustering quality above the baseline.

[13]The experiments in previous sections included only first-order relations.

[14]The preposition should be seen only as an implicit link between two words and is not included in the input data for clustering. For the example fragment only "eating" and "computer" constitute the data instance actually used by the algorithm.

Type	Ord-1	Ord-2	DepBrown	Population
OBJ2		■	0.238	1,622
LD	■		0.233	2,419
PC		■	0.211	21,157
LD		■	0.208	12,149
OBJ1	■		0.203	108,037
SU	■		0.199	79,844
ALL	■		0.196	495,479
ALL	■	■	0.196	559,908
SU+OBJ1	■		0.202	156,645

Table 4: Lin similarity scores for dependency Brown clustering (*DepBrown*) per type of dependency relation. Ord-1: first-order relation; Ord-2: second-order relation (with intervening preposition); Population: number of word types in the clustering.

Two conclusions can be drawn from the results on these relations. First, some dependency relations contribute better context that leads to increased semantic relatedness compared to clustering without relation selection. Second, both first- and second-order relations appear among the relations outperforming the baseline. The highest score from the top six relations is achieved by taking words exclusively from the second-order secondary object (OBJ2) relation. However, relatively few word types are included in the clusters. The same is true for the first-order directional complements (LD). Of course, clustering with only one of these relations would have quite limited applicability if used in a supervised NLP task due to the low number of word types. However, the main point we want to make here is that these relations yield semantically superior clusters and demonstrate that syntactic functions truly merit further attention in learning semantic clusters using syntax. The remaining four among the top six relations are more frequent relations, and lead to clusterings with higher number of word types. These are the second-order prepositional complement (PC) and directional complement (LD) relations, and the first-order direct object (OBJ1) and subject (SU) relations. Finally, the setting SU+OBJ1 joins words obtained from subject and direct object relations, and achieves a quality that falls between the values obtained for the two relations separately, yet still increases the number of word types.

6 Conclusion and future work

We have presented a detailed study on a simple extension of Brown clustering with a dependency language model. In the first part, we have consolidated the advantage of dependency clustering over standard Brown clustering in a series of experiments, including cluster capacity, granularity level, frequency thresholding, amount of data and other. In the second part, we put forward the idea of selective clustering using data obtained only from specific dependency relations. Several relations lead to a clustering with improved intracluster similarity. We make the code as well as the induced clusters freely available at `https://github.com/rug-compling/dep-brown-cluster`.

Our findings from the selective clustering warrant the development of more complex models capable of including syntactic functions for obtaining semantic clusters. We reserve this work for the future. We find it interesting to apply dependency Brown clustering to languages of different families and compare it in this setting to the standard Brown clustering. The future work further includes a study of the effect of dependency clusters in downstream tasks. Another important point is the effect of parser accuracy on the quality of obtained clusters.

Acknowledgments

Thanks to Çağrı Çöltekin, Gregory Mills, Olga Yeroshina and the anonymous reviewers for valuable suggestions, and to Percy Liang for implementation-related comments.

References

Jordan Boyd-Graber and David M. Blei. 2008. Syntactic topic models. In *NIPS*.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Marie Candito and Benoît Crabbé. 2009. Improving generative statistical parsing with semi-supervised word clustering. In *IWPT*.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *ACL*.

Wenliang Chen, Min Zhang, and Haizhou Li. 2012. Utilizing dependency language models for graph-based dependency parsing models. In *ACL*.

Grzegorz Chrupala. 2011. Efficient induction of probabilistic word classes with LDA. In *IJCNLP*.

Manaal Faruqui and Chris Dyer. 2013. An information theoretic approach to bilingual word clustering. In *ACL*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *WWW*.

Edouard Grave, Guillaume Obozinski, and Francis Bach. 2013. Hidden Markov tree models for semantic class induction. In *CoNLL*.

Gholamreza Haffari, Marzieh Razavi, and Anoop Sarkar. 2011. An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing. In *ACL*.

Fei Huang, Alexander Yates, Arun Ahuja, and Doug Downey. 2011. Language models as representations for weakly-supervised nlp tasks. In *CoNLL*.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple Semi-supervised Dependency Parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio.

Percy Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *ICML*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *HLT-NAACL*.

Saeedeh Momtazi, Sanjeev Khudanpur, and Dietrich Klakow. 2010. A comparative study of word co-occurrence for term clustering in language model-based sentence retrieval. In *ACL-HLT*.

Anjan Nepal and Alexander Yates. 2014. Factorial Hidden Markov models for learning representations of natural language. In *ICLR*.

Gertjan Van Noord. 2006. At Last Parsing Is Now Operational. In *TALN*.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *HLT-NAACL*.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33:161–199.

Barbara Plank and Alessandro Moschitti. 2013. Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. In *ACL*.

Barbara Plank and Gertjan van Noord. 2010. Grammar-driven versus data-driven: Which parsing system is more affected by domain shifts? In *NLPLING Workshop*.

Kashyap Popat, Balamurali A.R, Pushpak Bhattacharyya, and Gholamreza Haffari. 2013. The haves and the have-nots: Leveraging unlabelled corpora for sentiment analysis. In *ACL*.

Martin Popel and David Mareček. 2010. Perplexity of n-gram and dependency language models. In *TSD*.

Kenji Sagae and Andrew S. Gordon. 2009. Clustering words by syntactic similarity improves dependency parsing of predicate-argument structures. In *IWPT*.

David Sánchez, Montserrat Batet, and David Isern. 2011. Ontology-based information content computation. *Knowledge-Based Systems*, 24(2):297–303.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *ACL*.

Yongzhe Shi, Wei-Qiang Zhang, Jia Liu, and Michael Johnson. 2013. Rnn language model with word clustering and class-based output layer. *EURASIP Journal on Audio, Speech, and Music Processing*, (1).

Andreas Stolcke. 2002. SRILM—an extensible language modeling toolkit. In *ICSLP*.

Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. Semi-supervised relation extraction with large-scale word clustering. In *HLT-ACL*.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *HLT-NAACL*.

Ivan Titov and Alexandre Klementiev. 2012. A Bayesian approach to unsupervised semantic role induction. In *EACL*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *ACL*.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Tim van de Cruys. 2010. *Mining for Meaning: The Extraction of Lexico-semantic Knowledge from Text*. Ph.D. thesis, University of Groningen.

Piek Vossen, Isa Maks, Roxanne Segers, Hennie van der Vliet, Marie-Francine Moens, Katja Hofmann, Erik Tjong Kim Sang, and Maarten de Rijke, editors, 2013. *Cornetto: A Combinatorial Lexical Semantic Database for Dutch*. Springer.

An Empirical Evaluation of Automatic Conversion from Constituency to Dependency in Hungarian

Katalin Ilona Simkó[1], **Veronika Vincze**[1,2], **Zsolt Szántó**[1], **Richárd Farkas**[1]
[1]University of Szeged
Department of Informatics
[2]MTA-SZTE Research Group on Artificial Intelligence
kata.simko@gmail.com
{vinczev,szantozs,rfarkas}@inf.u-szeged.hu

Abstract

In this paper, we investigate the differences between Hungarian sentence parses based on automatically converted and manually annotated dependency trees. We also train constituency parsers on the manually annotated constituency treebank and then convert their output to dependency trees. We argue for the importance of training on gold standard corpora, and we also demonstrate that although the results obtained by training on the constituency treebank and converting the output to dependency format and those obtained by training on the automatically converted dependency treebank are similar in terms of accuracy scores, the typical errors made by different systems differ from each other.

1 Introduction

Nowadays, two popular approaches to data-driven syntactic parsing are based on constituency grammar on the one hand and dependency grammar on the other hand. There exist constituency-based treebanks for many languages and dependency treebanks for most of these languages are converted automatically from constituent trees with the help of conversion rules, which is the case for e.g. the languages used in the SPMRL-2013 Shared Task (Seddah et al., 2013) with the exception of Basque, where constituency trees are converted from manually annotated dependency trees (Aduriz et al., 2003), and Hungarian, where both treebanks are manually annotated (Csendes et al., 2005; Vincze et al., 2010). However, the quality of automatic dependency conversion is hardly investigated.

Hungarian is one of those rare examples where there exist manual annotations for both constituency and dependency syntax on the same bunch of texts, the Szeged (Dependency) Treebank (Csendes et al., 2005; Vincze et al., 2010), which makes it possible to evaluate the quality of a rule-based automatic conversion from constituency to dependency trees, to compare the two sets of manual annotations and also the output of constituency and dependency parsers trained on converted and gold standard dependency trees.

We investigate the effect of automatic conversions related to the two parsing paradigms as well. It is well known that for English, the automatic conversion of a constituency parser's output to dependency format can achieve competitive unlabeled attachment scores (ULA) to a dependency parser's output trained on automatically converted trees[1] (cf. Petrov et al. (2010)). One of the possible explanations for this is that English is a configurational language, hence constituency parsers have advantages over dependency parsers here. We check whether this hypothesis holds for Hungarian too, which is the prototype of free word order languages.

In this paper, we compare three pairs of dependency analyses in order to evaluate the usefulness of converted trees. First, we examine the errors of the conversion itself by comparing the converted dependency trees with the manually annotated gold standard ones. Second, we argue for the importance of training parsers on gold standard trees by looking at the typical differences between the outputs of

[1]However, it has been pointed out that errors in the conversion script may significantly influence the results of parsing, see e.g. Petrov and McDonald (2012) and Pitler (2012)

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1392–1401, Dublin, Ireland, August 23-29 2014.

dependency parsers trained on converted (silver standard) trees, parsers trained on gold standard trees and the manual annotation itself. Third, we demonstrate that similar to English, training on a constituency treebank and converting the results to dependency format can achieve similar results in terms of ULA to the dependency parser trained on the automatically converted treebank, but the typical errors they make differ in both cases.

2 Parsing Hungarian on the Szeged Treebank

Hungarian is a morphologically rich language, where word order encodes information structure, which makes its syntactic analysis very different from English's as the arguments in a sentence cannot be determined by their position but by their suffixes, cf. É. Kiss (2002). Words' grammatical functions are signified by case suffixes and verbs are marked for the number and person of their subject and the definiteness of their object, thus these arguments may be often omitted from the sentence: *Látlak* (see-1SG2OBJ) "I see you". Due to word order reasons, words that form one syntactic phrase may not be adjacent (long-distance dependencies), which is true for the possessive construction as well: the possessor and the possessed may be situated in two distant positions: *A fiúnak elvette a kalapját* (the boy-DAT take-PAST-3SGOBJ the hat-POSS3SG-ACC) "He took the boy's hat". Verbless clauses are also common in Hungarian, as the copula in third person singular present tense indicative form is phonologically empty, while it is present in all other moods and tenses: *A kalap piros* (the hat red) "The hat is red", but *A kalap piros volt* (the hat red was) "The hat was red".

The Szeged Treebank (Csendes et al., 2005) is a manually annotated constituency treebank for Hungarian consisting of 82,000 sentences. Besides the phrase structure, grammatical roles of the verbs' arguments and morphological information are also annotated. It incorporates texts from six different domains: short business news, newspaper, law, literature, compositions and informatics, however, in this paper, we just focus on the short business news domain.

The Szeged Dependency Treebank (Vincze et al., 2010) contains manual dependency syntax annotations for the same texts. Certain linguistic phenomena – such as discontinuous structures – are annotated in this treebank, but not in the constituency treebank. In the dependency treebank, the possessor is linked to the possession while this connection is not annotated in the constituency treebank. The two types of trees can be seen in Figure 1.

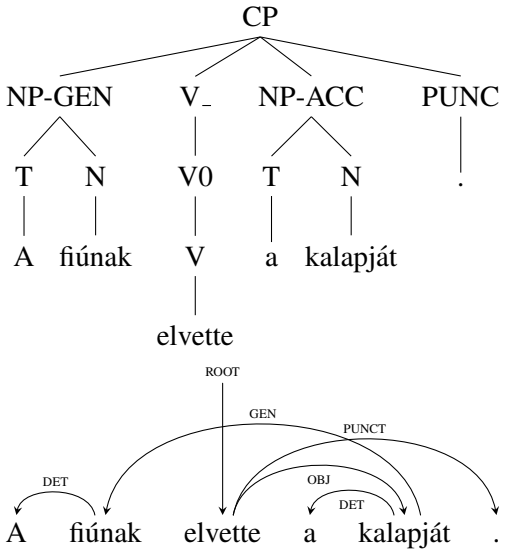


Figure 1: Discontinuous structure *A fiúnak elvette a kalapját* (the boy-DAT take-past3SGOBJ the hat-POSS3SG-ACC) "He took the boy's hat" in constituency and dependency analysis.

Another difference between the two treebanks is the way they represent different types of complex sentences, as can be seen in Figure 2. In the dependency treebank subordinations and coordinations are

handled very similarly. The head of one of the clauses (the subordinated clause or the second clause in the case of coordination) is linked to the head of the other clause (the matrix clause of the subordination or the first clause of the coordination), only the type of relation between the two heads differs in the two structures, in the dependency tree in Figure 2, the heads of the three clauses (*átjött* "came over", *megígérte* "promised" and *eljön* "come") are linked to one another through their conjunctions with either an ATT relation in the case of subordination or COORD for coordination. In the constituency treebank these sentences are represented very differently: in the case of subordination, the subordinated clause is within the matrix clause: CP_3 is within CP_2 in the constituency tree in Figure 2. Coordinated clauses appear at the same level in the structure, in the same figure CP_1 and CP_2 are coordinated clauses.

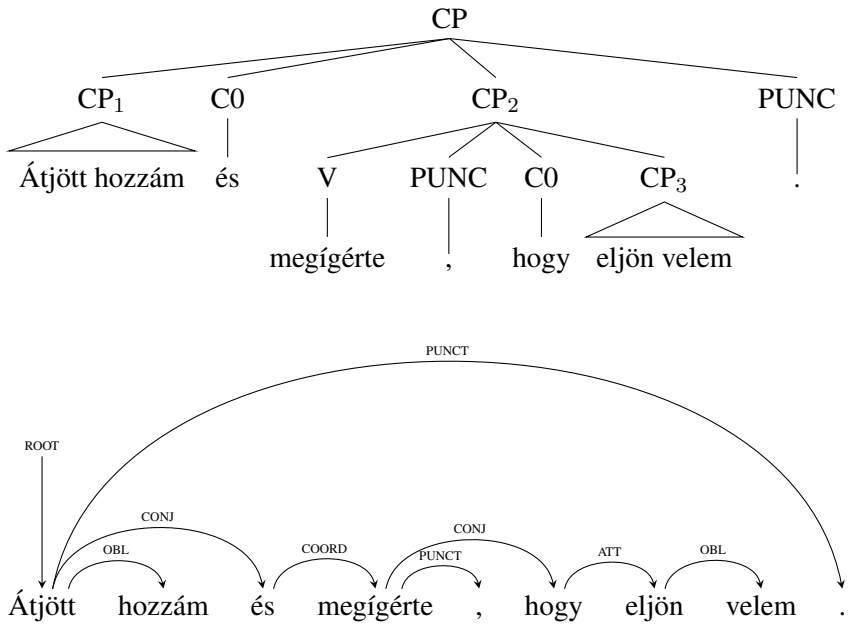


Figure 2: Constituency and dependency analysis of coordination and subordination in the sentence *Átjött hozzám és megígérte, hogy eljön velem* (through.come-PAST-3SG to.me and promise-PAST-3SG-OBJ that away.come-3SG with.me) "He came over and promised that he will come with me".

The parallels of these two manually annotated treebanks make them suitable for testing our hypotheses about automatic dependency conversion. The differences between them originate from the characteristics of constituent and dependency syntax.

3 Converting Constituency Trees to Dependency Trees

In this section, we present our methods to convert constituency trees to dependency trees and we also discuss the most typical sources of errors during conversion.

3.1 Conversion rules

In order to convert constituency trees to dependency trees, we used a rule based system. Sentences with virtual dependency nodes were omitted, as they are not annotated in the constituent treebank and their treatment in dependency trees is also problematic (Farkas et al., 2012; Seeker et al., 2012). As a result, we worked with 7,372 sentences and 162,960 tokens.

First, we determined the head of each clause (CP) and the relations between CPs in complex sentences. In most cases the head of the CP is a finite verb, if the CP contains no finite verb, the head is the either an infinitive verb or a participle, if none of these are present in the CP, the head can be a nominal expression. The relations between the CP heads make up the base of the dependency structure using ROOT relation for the sentence's main verb, COORD for coordination and ATT for subordination, as well as CONJ in the case of conjunctions between the CPs.

The arguments of verbs, infinitives and participles in the CP were linked to their governor and marked for their grammatical role in the Szeged Treebank. We used this information to construct the appropriate dependency relations between governors and their arguments. The main grammatical roles such as subject, object, dative have their own label in dependency syntax, while minor ones are assigned the oblique (OBL) relation. The argument's modifiers were then linked to the head or other modifiers based on the phrase structure with relations according to their morphological code.

Long distance dependencies, like the connection between a genitive case possessor and the possessed are not annotated in the constituency treebank. In these cases we used morphological information to link these elements together in the dependency tree. Figure 3 shows an example of converting a constituency tree to a dependency tree.

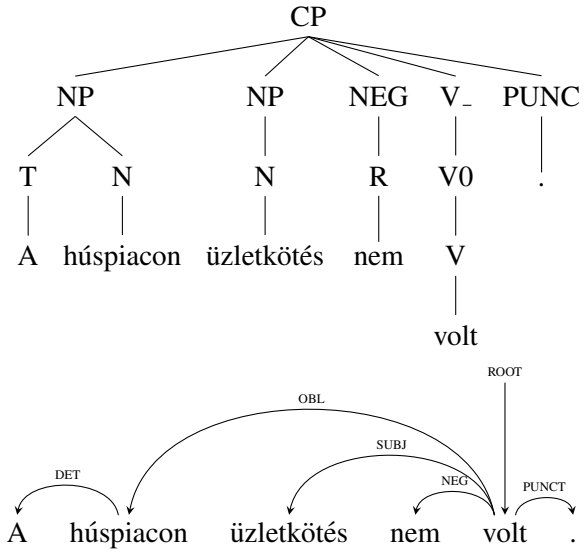


Figure 3: Conversion of the sentence *A húspiacon üzletkötés nem volt* (the meat.market-SUP transaction not was) "There were no transactions at the meat market." from constituency to dependency trees.

3.2 Error Analysis

We automatically converted the constituency treebank into dependency trees following the principles described above and detailed at our website (`http://www.inf.u-szeged.hu/rgai/SzegedTreebank`). For evaluation, we applied the metrics labeled attachment score (LAS) and unlabeled attachment score (ULA), without punctuation marks. The accuracy of the conversion was 96.51 (ULA) and 93.85 (LAS). The errors made during conversion were categorized manually in 200 sentences selected randomly from the short business news subcorpus of the Szeged Dependency Treebank, and the most typical ones are listed in Table 1, Column *convError*.

As it is shown, the most common source of error was when more than one modifier was within a phrase as the example in Figure 4 shows. In each figure, the gold standard parse can be seen on the left hand side while the erroneous one can be seen on the right hand side.

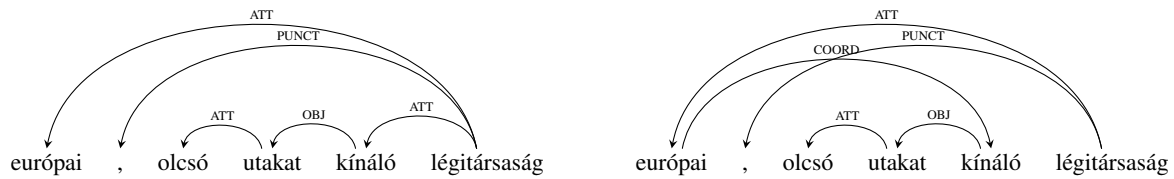


Figure 4: Multiple modifier error in *európai, olcsó utakat kínáló légitársaság* (European cheap trips-ACC offering airline) "European airline offering cheap trips".

Error type	convError		goldTrain		silverTrain		BerkeleyConv		convDep	
	#	%	#	%	#	%	#	%	#	%
Coordination	26	13.00	39	13.22	59	14.82	55	16.37	64	19.57
Multiple modifiers	26	13.00	30	10.17	49	12.31	52	15.48	47	14.37
Determiner	7	3.50	28	9.49	25	6.28	31	9.23	31	9.48
Conj./adverb attached	33	16.50	23	7.80	45	11.31	39	11.61	42	12.84
Arg. of verbal element	10	5.00	27	9.15	34	8.54	59	17.56	44	13.46
Sub- vs. coordination	7	3.50	9	3.05	12	3.02	–	–	–	–
Possessor	9	4.50	14	4.75	16	4.02	28	8.33	22	6.73
Wrong root	14	7.00	17	5.76	23	5.78	35	10.42	27	8.26
Consecutive nouns	4	2.00	11	3.73	14	3.52	13	3.87	15	4.59
Multiword NE	8	4.00	25	8.47	33	8.29	8	2.38	19	5.81
Wrong MOD label	25	12.50	26	8.81	34	8.54	–	–	–	–
Wrong other label	17	8.50	33	11.19	30	7.54	–	–	–	–
Other errors	14	7.00	13	4.41	24	6.03	16	4.76	16	4.89
Total	200	100	295	100	398	100	336	100	327	100

Table 1: Error Types. convError: errors made during converting constituency trees to dependency trees. goldTrain: errors in the output got by training the Bohnet parser on the gold standard data. silverTrain: errors in the output got by training the Bohnet parser on the silver standard data. BerkeleyConv: errors in the output got by training the Berkeley parser on the gold standard constituency data and converting the output into dependency format. convDep: errors in the output got by training the Bohnet parser without dependency labels on the silver standard data.

Coordination errors occurred when multiple members of a coordination were wrongly connected. On the other hand, the attachment of conjunctions and some adverbs was also problematic, for example in Figure 5 the conjunction *is* "also" is connected to the verb in the gold standard and to the noun in the converted version.

Figure 5: Conjunction attachment error in *a minisztérium is beszáll* (the ministry also steps.in) "the ministry also steps in".

Also, the constituency treebank did not mark all the grammatical relations (e.g. numerals and determiners were simply parts of an NP but had no distinct labeling, like *[NP az öt [ADJP fekete] kutya]* (the five black dog) "the five black dogs"), but it was necessary to assign them a dependency label and a parent node during conversion. However, in some cases it was not straightforward which modifier modifies which parent node: for instance, in *[NP nem [ADJP megfelelő] módszerek]* (not appropriate methods) "inappropriate methods", the negation word *nem* is erroneously attached to the noun instead of the adjective in the converted phrase. Determiner errors were those where the determiner was attached to the wrong noun in a NP with a noun modifier. In CPs with multiple verbal elements (both a finite verb and an infinitive or a participle in the CP) the arguments were sometimes linked to the wrong verb, as in Figure 6.

Figure 6: Verbal argument error in *a saját pecsenyéjükkel voltak elfoglalva* (the own roast-3PLPOSS-INS were busy) “they were busy with their own thing”.

Possessors are sometimes wrongly identified during conversion as long distance dependencies are not marked in the constituency treebank (see Figure 7).

Figure 7: Possessor attachment error in *a gyártó szárítóüzemében hasznosít* (the manufacturer drying.plant-3SGPOSS-INE utilizes) “the manufacturer utilizes it in its drying plant”.

In CPs with more verbal element, sometimes the wrong word is selected as the root, as in Figure 8.

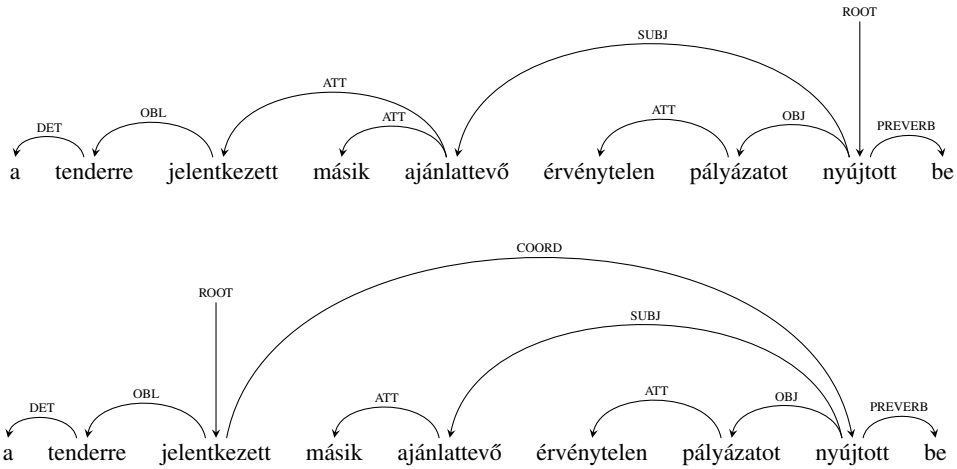


Figure 8: Root error in *a tenderre jelentkezett másik ajánlattevő érvénytelen pályázatot nyújtott be* (the tender-SUB applied other bidder invalid application-ACC submit-PAST-3SG) “the other bidder applying to the tender submitted an invalid application”.

In some cases, consecutive (but separate) noun phrases were taken as one unit as if one noun modified the other, for example in Figure 9.

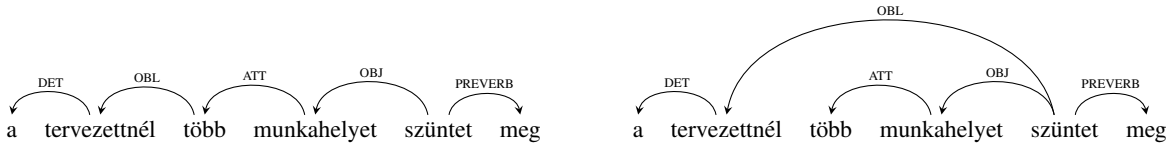


Figure 9: Consecutive noun error in *a tervezettnél több munkahelyet szüntet meg* (the planned-ADE more workplace-ACC terminates) “it terminates more workplaces than planned”.

Multiword NEs also caused some problems in the conversion, as in Figure 10.

Figure 10: Multiword NE error in *Beszállítói Befektető Rt.* (a name of a company) .

In other cases, divergences between the gold standard and the converted trees are due to some erroneous annotations either in the constituency treebank or in the dependency treebank. A typical example of this is the wrong MOD (modifier) label. In the treebank, locative and temporal modifiers were classified according to the tridirectionality typical of Hungarian adverbs and case suffixes: *where*, *from where* and *to where* (or *when*, *from what time and till what time*) the action is taken place. Thus, there are six dependency relations dedicated to these aspects and all the other adverbials are grouped under the relation MOD. However, this distinction is rather semantic in nature and was sometimes erroneously annotated in the constituency treebank, which was later corrected in the dependency one and thus now resulted in conversion errors, as shown in Figure 11.

Figure 11: MOD label error in *nyár vége felé kezdik* (summer end-3SGPOSS around begin) "they begin around the end of the summer".

There were also some atypical errors that occurred too rarely to categorize them in a different class, like cases when an article or determiner got erroneously attached to a verb and so on, so they were lumped into the category of "other errors" in Table 1.

4 Training on Gold Standard and Silver Standard Trees

We also experimented with training the Bohnet dependency parser (Bohnet, 2010) on the manually annotated (gold standard) and the converted (silver standard) treebank. The Bohnet parser (Bohnet, 2010) is a state-of-the-art[2] graph-based parser, which employs online training with a perceptron. The parser contains a feature function for the first order factor, one for the sibling factor, and one for the grandchildren.

From the corpus, 5,892 sentences (130,211 tokens) were used in the training dataset and the remaining 1,480 sentences (32,749 tokens) in the test dataset. For evaluation, we again applied the metrics LAS and ULA. Results are shown in Table 2, Rows *goldTrain* and *silverTrain*.

As the numbers show, better results can be achieved when the gold standard data are used as training database than when the parser is trained on the silver standard data, the differences being 1.6% (ULA) and 3.16% (LAS). Besides evaluation scores, we also compared the outputs of the two scenarios: we used the same set of randomly selected sentences as when investigating conversion errors and carried out a manual error analysis against the gold standard data in each case: see Table 1, Columns *goldTrain* and *silverTrain*.

There are some common error types that seem to cause problems for both ways of parsing. For instance, coordination and multiple modifiers are among the most frequent sources of errors in both cases as for the error rates are concerned. However, with regard to the absolute numbers, we can see that both error types are reduced when the gold standard dataset is used for training. On the other hand, finding the parent node of a conjunction or an adverb seems to improve significantly when the parser is trained on gold standard data. This is probably due to the fact that they are not marked in the constituency treebank and thus training data for these grammatical phenomena are very noisy in the silver standard treebank. All in all, we argue that there are some grammatical phenomena – e.g. the attachment of

[2]For a comparative evaluation with other dependency parsers on the same treebank see Farkas et al. (2012). According to their results, the Bohnet parser achieved the best scores on the treebank hence we also used this parser in our experiments.

Setting	LAS	ULA
Conversion	93.85	96.51
goldTrain	93.48	95.17
silverTrain	90.32	93.57
BerkeleyConv	–	92.78
convDep	–	93.23

Table 2: Results of the experiments. Conversion: converting constituency trees to dependency trees. goldTrain: training the Bohnet parser on the gold standard data. silverTrain: training the Bohnet parser on the silver standard data. BerkeleyConv: training the Berkeley parser on the gold standard constituency data and converting the output into dependency format. convDep: training the Bohnet parser without dependency labels on the silver standard data.

conjunctions or adverbs – that require manual checking even if automatic conversion from constituency to dependency is applied.

5 Pre- or Post Conversion?

It is well known that for English, converting a constituency parser's output to dependency format (post conversion) can achieve competitive ULA scores to a dependency parser's output trained on automatically converted trees (pre conversion) (Petrov et al., 2010; Farkas and Bohnet, 2012). One of the possible reasons for this may be that English is a configurational language, hence constituency parsers are expected to perform better here. In this paper, we investigate whether this is true for Hungarian, which is the prototype of morphologically rich languages with free word order.

We employed the product-of-grammars procedure (Petrov, 2010) of the Berkeleyparser (Petrov et al., 2006), where grammars are trained on the same dataset but with different initialization setups, which leads to different grammars. We trained 8 grammars and used tree-level inference. The output of the parser was then automatically converted to dependency format, based on the rules described in Section 3 (*BerkeleyConv*). Second, we used the silver standard dependency treebank for training the Bohnet parser (*convDep*). Since our constituency parser did not produce grammatical functions for the nodes, we trained the Bohnet parser on unlabeled dependency trees in order to ensure a fair comparison here (that is the difference between the columns *BerkeleyConv* and *convDep* in Table 1).

As the numbers show, competitive results can be obtained with both methods, yielding an ULA score of 92.78 and 93.23, respectively. This means that the same holds for Hungarian as for English and the surprisingly good results of post conversion are not related to the configurational level of the language.

Manually analysing the errors on the same set of sentences as before, there are again some error categories that occur frequently in both cases such as coordination, the attachment of conjunctions, modifiers and determiners. On the other hand, training on constituency trees seems to have some specific sources of errors. First, the possessor in possessive constructions is less frequently attached to its possessed, which may be due to the fact that the genitive possessor is not linked to the possessed in the constituency treebank and thus the parser is not able to learn this relationship. Second, arguments of verbal elements (i.e. verbs, participles and infinitives) are also somewhat more difficult to find when there are at least two verbal elements within the clause, which is especially true for adverbial participles and infinitives. In Figure 6, the differences between the two trees are shown. The noun *pecsenyéjükkel* (roast-3PLPOSS-INS) "with their thing" is linked to the adverbial participle in the correct analysis, but it connects to the main verb in the other. Third, identifying the root node of the sentence may also be problematic for this setting. As Farkas and Bohnet (2012) reported that preconversion can achieve better results for finding the root node in English, this seems to be a language-specific issue and it represents an interesting difference between English and Hungarian. Nevertheless, training on constituency trees has a beneficial effect on finding multiword named entities. Hence, it can be concluded that although the evaluation scores are similar, the errors the two systems make differ from each other.

6 Discussion and Conclusions

Here, we compared dependency analyses of Hungarian obtained in different ways. It was revealed that although the accuracy scores are similar to each other, each system makes different types of errors. On the other hand, there are some specific linguistic phenomena that seem to be difficult for dependency parsing generally as they were among the most frequent sources of errors in each case (e.g. coordination, multiple modifiers and the attachment of conjunctions and adverbs).

Converting constituency trees into dependency trees enabled us to experiment with a silver standard dependency corpus as well. Our results empirically showed that better results can be achieved on the gold standard corpus, hence manual annotation of dependency trees is desirable. However, when there is no access to manually annotated dependency data, converting the output of a constituency parser into dependency format or training the dependency parser on converted data may also be viable: similar to English, both solutions result in competitive scores but the errors the systems make differ from each other.

In the future, we would like to investigate how the advantages of constituency and dependency representations may be further exploited in parsing Hungarian and we also plan to carry out some uptraining experiments with both types of parsers.

Acknowledgements

This work was supported in part by the European Union and the European Social Fund through the project FuturICT.hu (grant no.: TÁMOP-4.2.2.C-11/1/KONV-2012-0013).

References

Itziar Aduriz, Maria Jesus Aranzabe, Jose Maria Arriola, Aitziber Atutxa, A. Diaz de Ilarraza, Aitzpea Garmendia, and Maite Oronoz. 2003. Construction of a Basque dependency treebank. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 201–204, Växjö, Sweden.

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

Dóra Csendes, János Csirik, Tibor Gyimóthy, and András Kocsor. 2005. The Szeged TreeBank. In Václav Matousek, Pavel Mautner, and Tomás Pavelka, editors, *Proceedings of the 8th International Conference on Text, Speech and Dialogue, TSD 2005*, Lecture Notes in Computer Science, pages 123–132, Berlin / Heidelberg, September. Springer.

Katalin É. Kiss. 2002. *The Syntax of Hungarian*. Cambridge University Press, Cambridge.

Richárd Farkas and Bernd Bohnet. 2012. Stacking of dependency and phrase structure parsers. In *Proceedings of COLING 2012*, pages 849–866, Mumbai, India, December. The COLING 2012 Organizing Committee.

Richárd Farkas, Veronika Vincze, and Helmut Schmid. 2012. Dependency Parsing of Hungarian: Baseline Results and Challenges. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 55–65, Avignon, France, April. Association for Computational Linguistics.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL).

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440.

Slav Petrov, Pi-Chuan Chang, Michael Ringgaard, and Hiyan Alshawi. 2010. Uptraining for accurate deterministic question parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 705–713, Cambridge, MA, October. Association for Computational Linguistics.

Slav Petrov. 2010. Products of random latent variable grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27, Los Angeles, California, June. Association for Computational Linguistics.

Emily Pitler. 2012. Conjunction representation and ease of domain adaptation. Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL).

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Yuval Marton, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, and Alina Wróblewska. 2013. Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 146–182, Seattle, Washington, USA, October. Association for Computational Linguistics.

Wolfgang Seeker, Richárd Farkas, Bernd Bohnet, Helmut Schmid, and Jonas Kuhn. 2012. Data-driven dependency parsing with empty heads. In *Proceedings of COLING 2012: Posters*, pages 1081–1090, Mumbai, India, December. The COLING 2012 Organizing Committee.

Veronika Vincze, Dóra Szauter, Attila Almási, György Móra, Zoltán Alexin, and János Csirik. 2010. Hungarian Dependency Treebank. In *Proceedings of LREC 2010*, Valletta, Malta, May. ELRA.

Deep-Syntactic Parsing

Miguel Ballesteros[1], **Bernd Bohnet**[2], **Simon Mille**[1], **Leo Wanner**[1,3]
[1]Natural Language Processing Group, Pompeu Fabra University, Barcelona, Spain
[2]School of Computer Science, University of Birmingham, United Kingdom
[3]Catalan Institute for Research and Advanced Studies (ICREA)
[1,3]{name.lastname}@upf.edu [2]bohnetb@cs.bham.ac.uk

Abstract

"Deep-syntactic" dependency structures that capture the argumentative, attributive and coordinative relations between full words of a sentence have a great potential for a number of NLP-applications. The abstraction degree of these structures is in-between the output of a syntactic dependency parser (connected trees defined over all words of a sentence and language-specific grammatical functions) and the output of a semantic parser (forests of trees defined over individual lexemes or phrasal chunks and abstract semantic role labels which capture the argument structure of predicative elements, dropping all attributive and coordinative dependencies). We propose a parser that delivers deep syntactic structures as output.

1 Introduction

Surface-syntactic structures (SSyntSs) as produced by data-driven syntactic dependency parsers are *per force* idiosyncratic in that they contain governed prepositions, determiners, support verb constructions and language-specific *grammatical functions* such as, e.g., SBJ, OBJ, PRD, PMOD, etc. (Johansson and Nugues, 2007). For many NLP-applications, including machine translation, paraphrasing, text simplification, etc., such a high idiosyncrasy is obstructive because of the recurrent divergence between the source and the target structures. Therefore, the use of more abstract "syntactico-semantic" structures seems more appropriate. Following Mel'čuk (1988), we call these structures *deep-syntactic structures* (DSyntSs). DSyntSs are situated between SSyntSs and PropBank- (Palmer et al., 2005) or Semantic Frame-like structures (Fillmore et al., 2002). Compared to SSyntSs, they have the advantage to abstract from language-specific grammatical idiosyncrasies. Compared to PropBank and Semantic Frame stuctures, they have the advantage to be connected and complete, i.e., capture all argumentative, attributive and coordinative dependencies between the meaningful lexical items of a sentence, while PropBank and Semantic Frame structures are not always connected, may contain either individual lexical items or phrasal chunks as nodes, and discard attributive and coordinative relations (be they within the chunks or sentential). In other words, they constitute incomplete structures that drop not only idiosyncratic, functional but also meaningful elements of a given sentence and often contain dependencies between chunks rather than individual tokens. Therefore, we propose to put on the research agenda the task of deep-syntactic parsing and show how a DSyntS is obtained from a SSynt dependency parse using data-driven tree transduction in a pipeline with a syntactic parser.[1] In Section 2, we introduce SSyntSs and DSyntSs and discuss the fundamentals of SSyntS–DSyntS transduction. Section 3 describes the experiments that we carried out on Spanish material, and Section 4 discusses their outcome. Section 5 summarizes the related work, before in Section 6 some conclusions and plans for future work are presented.

2 Fundamentals of SSyntS–DSyntS transduction

Before we set out to discuss the principles of the SSyntS–DSynt transduction, we must specify the DSyntSs and SSyntSs as used in our experiments.

[1]The term 'tree transduction' is used in this paper in the sense of Rounds (1970) and Thatcher (1970) to denote an extension of *finite state transduction* (Aho, 1972) to trees.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1402–1413, Dublin, Ireland, August 23-29 2014.

2.1 Defining SSyntS and DSyntS

SSyntSs and DSyntSs are directed, node- and edge-labeled dependency trees with standard feature-value structures (Kasper and Rounds, 1986) as node labels and dependency relations as edge labels.

The features of the node labels in **SSyntSs** are lex_{ssynt}, and "syntactic grammemes" of the value of lex_{ssynt}, i.e., *number*, *gender*, *case*, *definiteness*, *person* for nouns and *tense*, *aspect*, *mood* and *voice* for verbs. The value of lex_{ssynt} can be any (either full or functional) lexical item; in graphical representations of SSyntSs, usually only the value of lex_{ssynt} is shown. The edge labels of a SSyntS are grammatical functions 'subj', 'dobj', 'det', 'modif', etc. In other words, SSyntSs are syntactic structures of the kind as encountered in the standard dependency treebanks; cf., e.g., dependency version of the Penn TreeBank (Johansson and Nugues, 2007) for English, Prague Dependency Treebank for Czech (Hajič et al., 2006), Ancora for Spanish (Taulé et al., 2008), Copenhagen Dependency Treebank for Danish (Buch-Kromann, 2003), etc. In formal terms that we need for the outline of the transduction below, a SSyntS is defined as follows:

Definition 1 (SSyntS) *An SSyntS of a language $\mathcal{L}$ is a quintuple $T_{SS} = \langle N, A, \lambda_{l_s \to n}, \rho_{r_s \to a}, \gamma_{n \to g} \rangle$ defined over all lexical items L of $\mathcal{L}$, the set of syntactic grammemes G_{synt}, and the set of grammatical functions R_{gr}, where*

- *the set N of nodes and the set A of directed arcs form a connected tree,*
- *$\lambda_{l_s \to n}$ assigns to each $n \in N$ an $l_s \in L$,*
- *$\rho_{r_s \to a}$ assigns to each $a \in A$ an $r \in R_{gr}$, and*
- *$\gamma_{n \to g}$ assigns to each $\lambda_{l_s \to n}(n)$ a set of grammemes $G_t \in G_{synt}$.*

The features of the node labels in **DSyntSs** as worked with in this paper are lex_{dsynt} and "semantic grammemes" of the value of lex_{dsynt}, i.e., *number* and *determination* for nouns and *tense*, *aspect*, *mood* and *voice* for verbs.[2] In contrast to lex_{ssynt} in SSyntS, DSyntS's lex_{dsynt} can be any *full*, but not a *functional* lexeme. In accordance with this restriction, in the case of *look after a person*, AFTER will not appear in the corresponding DSyntS; it is a functional (or governed) preposition (so are TO or BY, in Figure 1).[3] In contrast, AFTER in *leave after the meeting* is a full lexeme; it will remain in the DSyntS because there it has its own meaning of "succession in time". The edge labels of a DSyntS are language-independent "deep-syntactic" relations I,...,VI, ATTR, COORD, APPEND. 'I',...,'VI' are argument relations, analogous to $A0$, $A1$, etc. in the PropBank annotation. 'ATTR' subsumes all (circumstantial) ARGM-x PropBank relations as well as the modifier relations not captured by the PropBank and FrameNet annotations. 'COORD' is the coordinative relation as in: *John*-COORD→*and*-II→*Mary*, *publish*-COORD→*or*-II→*perish*, and so on. APPEND subsumes all parentheticals, interjections, direct addresses, etc., as, e.g., in *Listen, John!*: *listen*-APPEND→*John*. DSyntSs thus show a strong similarity with PropBank structures, with four important differences: (i) their lexical labels are not disambiguated; (ii) instead of circumstantial thematic roles of the kind ARGM-LOC, ARGM-DIR, etc. they use a unique ATTR relation; (iii) they capture all existing dependencies between meaningful lexical nodes; and (iv) they are connected.[4] A number of other annotations have resemblance with DSyntSs; cf. (Ivanova et al., 2012) for an overview of deep dependency structures. Formally, a DSyntS is defined as follows:

Definition 2 (DSyntS) *An DSyntS of a language $\mathcal{L}$ is a quintuple $T_{DS} = \langle N, A, \lambda_{l_s \to n}, \rho_{r_s \to a}, \gamma_{n \to g} \rangle$ defined over the full lexical items L_d of $\mathcal{L}$, the set of semantic grammemes G_{sem}, and the set of deep-syntactic relations R_{dsynt}, where*

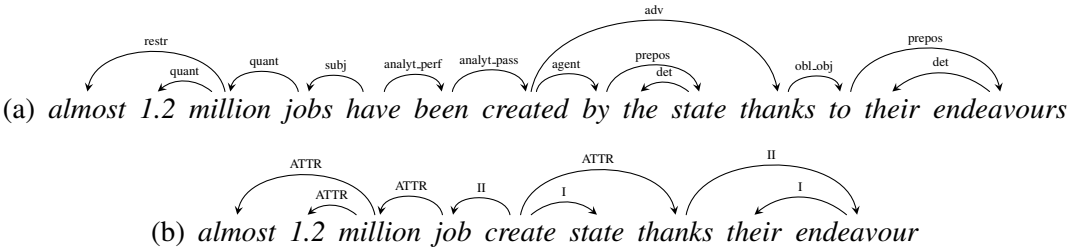
- *the set N of nodes and the set A of directed arcs form a connected tree,*
- *$\lambda_{l_s \to n}$ assigns to each $n \in N$ an $l_s \in L_d$,*
- *$\rho_{r_s \to a}$ assigns to each $a \in A$ an $r \in R_{dsynt}$, and*
- *$\gamma_{n \to g}$ assigns to each $\lambda_{l_s \to n}(n)$ a set of grammemes $G_t \in G_{sem}$.*

Consider in Figure 1 an example for an SSyntS and its corresponding DSyntS.

[2] Most of the grammemes have a semantic and a surface interpretation; see (Mel'čuk, 2013).

[3] Functional lexemes also include auxiliaries (e.g. HAVE, or BE when it is not a copula), and definite and indefinite determiners (THE, A); see Figure 1).

[4] Our DSyntSs are thus DSyntSs as used in the Meaning-Text Theory (Mel'čuk, 1988), only that our DSyntSs do not disambiguate lexical items and do not use *lexical functions* (Mel'čuk, 1996).



(a) *almost 1.2 million jobs have been created by the state thanks to their endeavours*

(b) *almost 1.2 million job create state thanks their endeavour*

Figure 1: An SSyntS (a) and its corresponding DSyntS (b)

2.2 Fleshing out the SSyntS–DSyntS transduction

It is clear that the SSyntS and DSyntS of the same sentence are not isomorphic. The following correspondences between the SSyntS S_{ss} and DSyntS S_{ds} of a sentence need to be taken into account during SSyntS–DSyntS transduction:

(i) a node in S_{ss} is a node in S_{ds};
(ii) a relation in S_{ss} corresponds to a relation in S_{ds};
(iii) a fragment of the S_{ss} tree corresponds to a single node in S_{ds};
(iv) a relation with a dependent node in S_{ss} is a grammeme in S_{ds};
(v) a grammeme in S_{ss} is a grammeme in S_{ds};
(vi) a node in S_{ss} is conflated with another node in S_{ds}; and
(vii) a node in S_{ds} has no correspondence in S_{ss}.

The grammeme correspondences (iv) and (v) and the "pseudo" correspondences in (vi) and (vii)[5] are few or idiosyncratic and are best handled in a rule-based post-processing stage. The main task of the SSyntS–DSyntS transducer is thus to cope with the correspondences (i)–(iii). For this purpose, we can view both SSyntS and DSyntS as vectors indexed in terms of two-dimensional matrices $I = N \times N$ (N being the set of nodes of a given tree $1, \ldots, m$), with $I(i,j) = \rho(n_i, n_j)$, if $n_i, n_j \in N$ and $(n_i, n_j) \in A$ and $I(i,j) = 0$ otherwise (where '$\rho(n_i, n_j)$' is the function that assigns to an edge a relation label and $i, j = 1, \ldots, m; i \neq j$ are nodes of the tree). That is, for a given SSyntS, the matrix $I(i,j)$ contains in the cells (i,j), $i,j = 1, \ldots, m$, the names of the SSynt-relations between the nodes n_i and n_j , and '0' otherwise, while for a given DSyntS, the cells of its matrix I_D contain DSyntS-relations.

Starting from the matrix I_S of a given SSyntS, the task is therefore to obtain the matrix I_D of the corresponding DSyntS, that is, to identify correspondences between i/j, (i,j) and groups of (i,j) of I_S with i'/j' and (i',j') of I_D; see (i)–(iii) above. In other words, the task consists in identifying and removing all functional lexemes, and attach correctly the remaining nodes between them.[6]

As a "token chain→surface-syntactic tree" projection, this task can be viewed as a classification task. However, while the former is isomorphic, we know that the SSyntS–DSyntS projection is not. In order to approach the task to an isomorphic projection (and thus simplify its modelling), it is convenient to interpret SSyntS and the targeted DSyntS as collections of *hypernodes*:

Definition 3 (Hypernode) *Given a SSyntS S_s with its index matrix I_S (a DSyntS S_d with its index matrix I_D), a node partition p (with $|p| \geq 1$) of I_S (I_D) is a hypernode h_{s_i} (h_{d_i}) iff p corresponds to a partition p' (with $|p'| \geq 1$) of S_d (S_s).*

In this way, the SSyntS–DSyntS correspondence boils down to a correspondence between individual hypernodes and between individual arcs, and the transduction embraces the following three (classification) subtasks: 1. Hypernode identification, 2. DSynt tree construction, and 3. DSynt arc labeling, which are completed by a post-processing stage.

[5](vi) covers, e.g., reflexive verb particles such as *se* in Spanish, which are conflated in the DSyntS with the verb: *se*←aux_refl_dir-*conocer* vs. CONOCERSE 'know each other'; (vii) covers, e.g., the zero subject in pro-drop languages (which is absent in the SSyntS and present in the DSyntS).

[6]What is particularly challenging is the identification of functional prepositions: based on the information found in the corpus only, our system must decide if a given preposition is a full or a functional lexeme. That is, we do not resort to any external lexical resources.

1. Hypernode identification. The hypernode identification consists of a binary classification of the nodes of a given SSyntS as nodes that form a hypernode of cardinality 1 (i.e., nodes that have a one-to-one correspondence to a node in the DSyntS) vs. nodes that form part of a hypernode of cardinality > 1. In practice, hypernodes of type one will be formed by: 1) noun nodes that do not govern determiner or functional preposition nodes, 2) full verb nodes that are not governed by any auxiliary verb nodes and that do not govern any functional preposition node, adjective nodes, adverbial nodes, and semantic preposition nodes. Hypernodes of type two will be formed by: 1) noun nodes + determiner / functional preposition nodes they govern, 2) verb nodes + auxiliary nodes they are governed by + functional preposition nodes they govern.

2. DSynt tree reconstruction. The outcome of the hypernode identification stage is thus the set $H_s = H_{s_{|p|=1}} \cup H_{s_{|p|>1}}$ of hypernodes of two types. With this set at hand, we can define an isomorphy function $\tau : H_s \rightarrow H_{d_{|p|=1}}$ (with $h_d \in H_{d_{|p|=1}}$ consisting of $n_d \in N_{ds}$, i.e., the set of nodes of the target DSyntS). τ is the identity function for $h_s \in H_{s_{|p|=1}}$. For $h_s \in H_{s_{|p|>1}}$, τ maps the functional nodes in h_s onto grammemes (attribute-value pairs) of the lexically meaningful node in h_d and identifies the lexically meaningful node as head. Some of the dependencies of the obtained nodes $n_d \in N_{ds}$ can be recovered from the dependencies of their sources. Due to the projection of functional nodes to grammemes (which can be also seen as node removal), some dependencies will be also missing and must be introduced. Algorithm 1 recalculates the dependencies for the target DSyntS S_d, starting from the index matrix I_S of SSyntS S_s to obtain a connected tree.

Algorithm 1: DSyntS tree reconstruction

for $\forall n_i \in N_d$ **do**
  if $\exists n_j : (n_j, n_i) \in S_s \wedge \tau(n_j) \in N_d$ **then**
  $(n_j, n_i) \rightarrow S_d$ // the equivalent of the head node of n_i is included in DSyntS
  else if $\exists n_j, n_a : (n_j, n_i) \in S_s \wedge \tau(n_j) \notin N_d \wedge$
      $\tau(n_a) \in N_d$ **then**
      //n_a is the first ancestor of n_j that has an equivalent in DSyntS
      //the equivalent of the head node of n_i is not included in DSyntS, but the ancestor n_a is
      $(n_a, n_i) \rightarrow S_d$
    else
      //the equivalent of the head node of n_i is not included in DSyntS, but several ancestors of it are
      $n_b := BestHead(n_i, S_s, S_d)$
      $(n_b, n_i) \rightarrow S_d$
endfor

$BestHead$ recursively ascends S_s from a given node n_i until it encounters one or several head nodes $n_d \in N_{ds}$. In case of several encountered head nodes, the one which governs the highest frequency dependency is returned.

3. Label Classification. The tree reconstruction stage produces a "hybrid" connected dependency tree $S_{s \rightarrow d}$ with DSynt nodes N_{ds}, and arcs A_s labelled by SSynt relation labels, i.e., an index matrix we can denote as I^-, whose cells (i, j) contain SSynt labels for all $n_i, n_j \in N_{ds} : (n_i, n_j) \in A_s$ and '0' otherwise. The next and last stage of SSynt-to-DSyntS transduction is thus the projection of SSynt relation labels of $S_{s \rightarrow d}$ to their corresponding DSynt labels, or, in other words, the mapping of I^- to I_D of the target DSyntS.

4. Postprocessing. As mentioned in Section 2, there is a limited number of idiosyncratic correspondences between elements of SSyntS and DSyntS (the correspondences (iv–vii) which can be straightforwardly handled by a rule-based postprocessor because (a) they are non-ambiguous, i.e., $a \leftrightarrow b, c \leftrightarrow d \Rightarrow a = b \wedge c = d$, and (b) they are few. Thus, only determiners and auxiliaries in SSyntS map onto a grammeme in DSyntS, both SSyntS and DSyntS count with less than a dozen grammemes, etc.

3 Experiments

In order to validate the outlined SSyntS–DSyntS transduction and to assess its performance in combination with a surface dependency parser, i.e., starting from plain sentences, we carried out a number of

experiments in which we implemented the transducer and integrated it into a pipeline shown in Figure 2.

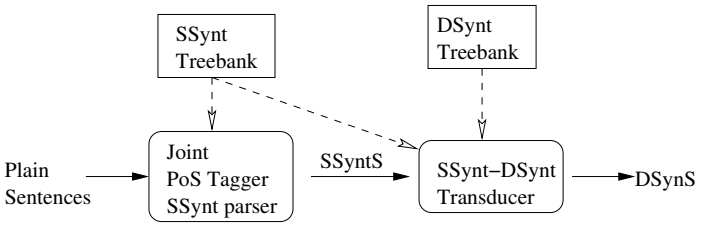


Figure 2: Setup of a deep-syntactic parser

For our experiments, we use the AnCora-UPF SSyntS and DSyntS treebanks of Spanish (Mille et al., 2013) in CoNLL format, adjusted for our needs. In particular, we removed from the 79-tag SSyntS treebank the semantically and information structure influenced relation tags to obtain an annotation granularity closer to the ones used for previous parsing experiments (55 relation tags, see (Mille et al., 2012)).

Our development set consisted of 219 sentences (3271 tokens in the DSyntS treebank and 4953 tokens in the SSyntS treebank), the training set of 3036 sentences (57665 tokens in the DSyntS treebank and 86984 tokens in the SSyntS treebank), and the *test set* held-out for evaluation of 258 sentences (5641 tokens in the DSyntS treebank and 8955 tokens in the SSyntS treebank).

To obtain the SSyntS, we use Bohnet and Nivre (2012)'s transition-based parser, which combines lemmatization, PoS tagging, and syntactic dependency parsing—tuned and trained on the respective sets of the SSyntS treebank. Cf. Table 1 for the performance of the parser on the development set.

POS	LEMMA	LAS	UAS
96.14	91.10	78.64	86.49

Table 1: Results of Bohnet and Nivre's surface-syntactic parser on the development set

In what follows, we first present the realization of the SSyntS–DSyntS transducer and then the realization of the baseline.

3.1 SSyntS–DSyntS transducer

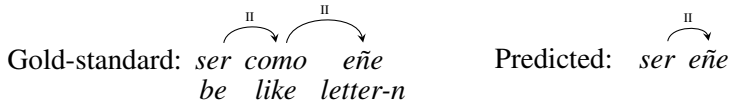
As outlined in Section 2.2, the SSyntS–DSyntS transducer is composed of three submodules and a post-processing stage:

1. Hypernode identification. For the hypernode identification, we trained a binary polynomial (degree 2) SVM from LIBSVM (Chang and Lin, 2001). The SVM allows both features related to the processed node and higher-order features, which can be related to the head node of the processed node or to its sibling nodes. After several feature selection trials, we chose the following features for each node n:

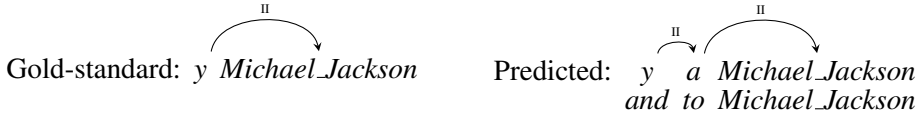
- lemma or stem of the label of n,
- label of the relation between n and its head,
- surface PoS of n's label (the SSynt and DSyntS treebanks distinguish between surface and deep PoS),
- label of the relation between n's head to its own head,
- surface PoS of the label of n's head node.

After an optimization round of the parameters available in the SVM implementation, the hypernode identification achieved over the gold development set 99.78% precision and 99.02% recall (and thus 99.4% F1). That is, only very few hypernodes are not identified correctly. The main error source are *governed prepositions*: the classifier has to learn when to assign a preposition an own hypernode (i.e., when it is lexically meaningful) and when it should be included into the hypernode of the governor (i.e., when it is functional). Our interpretation is that the features we use for this task are appropriate, but that the training data set is too small. As a result, some prepositions are erroneously left out from or introduced into the DSyntS.

2. Tree reconstruction. The implementation of the tree reconstruction module shows an unlabelled dependency attachment precision of 98.18% and an unlabelled dependency attachment recall of 97.43% over the gold development set. Most of the errors produced by this module have their origin in the previous module, i.e., hypernode identification. When a node has been incorrectly removed, the module errs in the attachment because it cannot use the node in question as the destination or the origin of a dependency, as it is the case in the gold-standard annotation:



When a node has erroneously not been removed, no dependencies between its governor and its dependent can be established since DSyntS must remain a tree (which gives the same LAS and UAS errors as when a node has been erroneously removed):



3. Relation label classification. For relation label classification, we use a multiclass linear SVM. The label classification depends on the concrete annotation schemata of the SSyntS and DSyntS treebanks on which the parser is trained. Depending on the schemata, some DSynt relation labels may be easier to derive from the original SSyntS relation labels than others. Table 2 lists all SSynt relation labels that have a straightforward mapping to DSyntS relation labels in the used treebanks, i.e., neither their dependent nor their governor are removed, and the SSyntS label always maps to the same DSynt label.

SSynt	DSynt	SSynt	DSynt	SSynt	DSynt	SSynt	DSynt
abbrev	ATTR	aux_refl_indir	III	dobj_clitic	II	prepos_quot	II
abs_pred	ATTR	bin_junct	ATTR	dobj_quot	II	prolep	APPEND
adv	ATTR	compl1	II	elect	ATTR	quant	ATTR
adv_mod	ATTR	compl2	III	juxtapos	APPEND	quasi_coord	COORD
agent	I	compl_adnom	ATTR	modal	II	quasi_subj	I
appos	ATTR	coord	COORD	modif	ATTR	relat	ATTR
attr	ATTR	copul	II	num_junct	COORD	restr	ATTR
aux_phras	—	copul_clitic	II	obj_copred	ATTR	sequent	ATTR
aux_refl_dir	II	copul_quot	II	prepos	II	subj	I
						subj_copred	ATTR

Table 2: Straightforward SSynt to DSyntS mappings

Table 3 shows SSyntS relation–DSyntS relation label correspondences that are not straightforward.

SSynt DepRel$_A$	Mapping to DSynt
analyt_fut	remove Gov and Dep; add tense=FUT
analyt_pass	remove Gov; invert I and II; add voice=PASS
analyt_perf	remove Gov; add tense=PAST
analyt_progr	remove Gov; add tem_constituency=PROGR
aux_refl_lex	remove Dep; add *se* at the end of Gov's lemma
aux_refl_pass	remove Dep; invert I and II; add voice=PASS
compar	remove Dep if conjunction
compar_/coord_/sub_conj	remove Dep if governed preposition
det	**IF** Dep=el—un **THEN** remove Dep; add definiteness=DEF/INDEF
	IF Dep=possessive **THEN** DepRel ATTR‖I‖II‖III
	IF Dep=other **THEN** DepRel ATTR
dobj	remove Dep if governed preposition
iobj	remove Dep if governed preposition; DepRel II‖III‖IV‖V‖VI
iobj_clitic	DepRel II‖III‖IV‖V‖VI
obl_compl	remove Dep if governed preposition; DepRel I‖II‖III‖IV‖V‖VI
obl_obj	remove Dep if governed preposition; DepRel II‖III‖IV‖V‖VI
punc	—
punc_init	—

Table 3: Complex SSynt to DSynt mappings

The final set of features selected for label classification includes: (i) lemma of the dependent node, (ii) dependency relation to the head of the dependent node, (iii) dependency relation label of the head node to its own head, (iv) dependency relation to the head of the sibling nodes of the dependent node, if any.

After an optimization round of the parameter set of the SVM-model, relation labelling achieved 94.00% label precision and 93.28% label recall on the development set. The recall is calculated considering all the nodes that are included in the gold standard. The error sources for relation labelling were mostly the dependencies that involved possessives and the various types of objects (see Table 3) due to their differing valency. For instance, the relation *det* in *su*$\leftarrow det-$*coche* 'his/her car' and *su*$\leftarrow det-$*llamada* 'his/her phone call' have different correspondences in DSyntS: *su*$\leftarrow ATTR-$*coche* vs. *su*$\leftarrow I-$*llamada*. That is, the DSyntS relation depends on the lexical properties of the governor.[7] Once again, more training data is needed in order to classify better those cases.

4. Postprocessing In the postprocessing stage for Spanish, the following rules capture non-ambiguous correspondences between elements of the SSynt-index matrix $I_S = N_s \times N_s$ and DSyntS index matrix $I_D = N_d \times N_d$, with $n_s \in N_s$ and $n_d \in N_d$, and n_s and n_d corresponding to each other (we do not list here identity correspondences such as between the number grammemes of n_s and n_d):

- if n_s is dependent of *analyt_pass* or *analyt_refl_pass* relation, then the voice grammeme in n_d is *PASS*;
- if n_s is dependent of *analyt_progr*, then the voice grammeme in n_d is *PROGR*;
- if n_s is dependent of *analyt_refl_lex*, then add the particle -SE as suffix of node label (word) of d_d;
- if any of the children of n_s is labelled by one of the tokens UN 'a$_{masc}$', UNA 'a$_{fem}$', UNOS 'some$_{masc}$' or UNAS 'some$_{fem}$', then the definiteness grammeme in n_d *INDEF*, otherwise it is *DEF*;
- if the n_s label is a finite verb and n_s does not govern a *subject* relation, then add to I' the relation $n_d - I \rightarrow n'_d$, with n'_d being a newly introduced node.

3.2 Baseline

As point of reference for the evaluation of the performance of our SSyntS–DSyntS transducer, we use a rule-based baseline that carries out the most direct transformations extracted from Tables 2 and 3. The baseline detects hypernodes by directly removing all the nodes that we are sure need to be removed, i.e. punctuation and auxiliaries. The nodes that are only *potentially* to be removed, i.e., all dependents of DepRels that have a possibly governed preposition or conjunction in Table 3, are left in the DSyntS. The new relation labels in the DSyntS are obtained by selecting the label that is most likely to substitute the SSyntS relation label according to classical grammar studies. The rules of the rule-based baseline look as follows:

```
1  if (deprel==abbrev) then deep_deprel=ATTR
2  if (deprel==obl_obj) then deep_deprel=II
...
n  if (deprel==punc) then remove(current_node)
```

4 Results and Discussion

Let us look in this section at the performance figures of the SSyntS parser, the SSyntS–DSyntS transducer, and the sentence–DSyntS pipeline obtained in the experiments.

4.1 SSyntS–DSyntS transducer results

In Table 4, the performance of the subtasks of the SSyntS–DSyntS transducer is contrasted to the performance of the baselines; the evaluation of the postprocessing subtask is not included because the one-to-one projection of SSyntS elements to DSyntS guarantees an accuracy of 100% of the operations performed. The transducer has been applied to the gold standard test set, which is the held-out test set, with gold standard PoS tags, lemmas and dependency trees. It outputs in total 5610 nodes; the rule-based baseline outputs 8653 nodes. As mentioned in Section 3, our gold standard includes 5641 nodes.

[7] Note that lexemes are not generalized: a verb and its corresponding noun (e.g., *construct*/*construction*) are considered distinct lexemes.

Hyper-Node Detection		
Measure	Rule-based Baseline	Tree Transducer
p	64.31 (5565/8653)	99.79 (5598/5610)
r	98.65 (5565/5641)	99.24 (5598/5641)
$F1$	77.86	99.51

Attachment and Labelling		
Measure	Rule-based Baseline	Tree Transducer
LAP	50.02 (4328/8653)	91.07 (5109/5610)
UAP	53.05 (4590/8653)	98.32 (5516/5610)
LA-P	57.66 (4989/8653)	92.37 (5182/5610)
LAR	76.72 (4328/5641)	90.57 (5109/5641)
UAR	81.37 (4590/5641)	97.78 (5516/5641)
LA-R	88.44 (4989/5641)	91.86 (5182/5641)

Table 4: Performance of the SSyntS–DSyntS transducer and of the rule-based baseline over the gold-standard held-out test set (LAP: labelled attachment precision, UAP: unlabelled attachment precision, LA-P: label assignment precision, LAR: labelled attachment recall, UAR: Unlabelled attachment recall and LA-R: Label assignment recall)

Our data-driven SSyntS–DSyntS transducer is much better than the baseline with respect to all evaluation measures.[8] The transducer relies on distributional patterns identified in the training data set, and makes thus use of information that is not available for the rule-based baseline, which studies one node at a time. However, the rule-based baseline results also show that transduction that would remove a few nodes would provide results close to a 100% recall for the hypernode detection because a DSynt tree is a subtree of the SSynt tree (if we ignore the nodes introduced by post-processing). This is also evidenced by the labeled and attachment recall scores. The results of the transducer on the test and development sets are quite comparable. The hypernode detection is even better on the test set. The label accuracy suffers most from using unseen data during the development of the system. The attachment figures are approximately equivalent on both sets.

4.2 Results of deep-syntactic parsing

Let us consider now the performance of the complete DSynt parsing pipeline (PoS-tagger+surface-dependency parser → SSyntS–DSyntS transducer) on the held-out test set. Table 5 displays the figures of the Bohnet and Nivre parser. The figures are in line with the performance of state-of-the-art parsers for Spanish (Mille et al., 2012).

POS	LEMMA	LAS	UAS
96.05	92.10	81.45	88.09

Table 5: Performance of Bohnet and Nivre's joint PoS-tagger+dependency parser trained on Ancora-UPF

Table 6 shows the performance of the pipeline when we feed the output of the syntactic parser to the rule-based baseline SSyntS–DSyntS module and the tree transducer. We observe a clear error propagation from the dependency parser (which provides 81.45% LAS) to the SSyntS–DSyntS transducer, which loses in tree quality more than 18%.

Hyper-Node Detection		
Measure	Baseline	Tree Transducer
p	63.87 (5528/8655)	97.07 (5391/5554)
r	98.00 (5528/5641)	95.57 (5391/5641)
$F1$	77.33	96.31

Labelling and Attachment		
Measure	Baseline	Tree Transducer
LAP	38.75 (3354/8655)	68.31 (3794/5554)
UAP	44.69 (3868/8655)	77.31 (4294/5554)
LA-P	49.66 (4298/8655)	80.47 (4469/5554)
LAR	59.46 (3354/5641)	67.26 (3794/5641)
UAR	68.57 (3868/5641)	76.12 (4294/5641)
LA-R	76.19 (4298/5641)	79.22 (4469/5641)

Table 6: Performance of the deep-syntactic parsing pipeline

5 Related Work

To the best of our knowledge, data-driven deep-syntactic parsing as proposed in this paper is novel. As *semantic role labeling* and *frame-semantic analysis*, it has the goal to obtain more semantically oriented structures than those delivered by state-of-the-art syntactic parsing. Semantic role labeling received considerable attention in the CoNLL shared tasks for syntactic dependency parsing in 2006 and 2007

[8] We also ran MaltParser by training it on the DSynt-treebank to parse the SSynt-test set; however, the outcome was too weak to be used as baseline.

(Buchholz and Marsi, 2006; Nivre et al., 2007), the CoNLL shared task for joint parsing of syntactic and semantic dependencies in 2008 (Surdeanu et al., 2008) and the shared task in 2009 (Hajič et al., 2009). The top ranked systems were pipelines that started with a syntactic analysis (as we do) and continued with predicate identification, argument identification, argument labeling, and word sense disambiguation; cf. (Johansson and Nugues, 2008; Che et al., 2009). At the end, a re-ranker that considers jointly all arguments to select the best combination was applied. Some of the systems were based on integrated syntactic and semantic dependency analysis; cf., e.g., (Gesmundo et al., 2009); see also (Lluís et al., 2013) for a more recent proposal along similar lines. However, all of them lack the ability to perform structural changes—as, e.g., introduction of nodes or removal of nodes necessary to obtain a DSyntS. Klimeš (2006)'s parser removes nodes (producing tectogrammatical structures as in the Prague Dependency Treebank), but is based on rules instead of classifiers, as in our case. The same applies to earlier works in the TAG-framework, as, e.g., in (Rambow and Joshi, 1997).

However, this is not to say that the idea of the surface→surface syntax→deep syntax pipeline is new. It goes back at least to Curry (1961) and is implemented in a number of more recent works; see, e.g., (de Groote, 2001; Klimeš, 2006; Bojar et al., 2008).

6 Conclusions and Future Work

We have presented a deep-syntactic parsing pipeline which consists of a state-of-the-art dependency parser and a novel SSyntS–DSyntS transducer. The obtained DSyntSs can be used in different applications since they abstract from language-specific grammatical idiosyncrasies of the SSynt structures as produced by state-of-the art dependency parsers, but still avoid the complexities of genuine semantic analysis.[9] DSyntS-treebanks needed for data-driven applications can be bootstrapped by the pipeline. If required, a SSyntS–DSyntS structure pair can be also mapped to a pure predicate-argument graph such as the DELPH-IN structure (Oepen, 2002) or to an approximation thereof (as the Enju conversion (Miyao, 2006), which keeps functional nodes), to an DRS (Kamp and Reyle, 1993), or to a PropBank structure. On the other hand, DSyntS-treebanks can be used for automatic extraction of deep grammars. As shown by Cahill et al. (2008), automatically obtained resources can be of an even better quality than manually-crafted resources. In this context, especially research in the context of CCGs (Hockenmeier, 2003; Clark and Curran, 2007) and TAGs (Xia, 1999) should be also mentioned.

To validate our approach with languages other than Spanish, we carried out an experiment on a Chinese SSyntS-DSyntS Treebank (training the DSynt-transducer on the outcome of the SSynt-parser). The results over predicted input showed an accuracy of about 75%, i.e., an accuracy comparable to the accuracy achieved for Spanish. We are also investigating multilingual approaches, such as the one proposed by McDonald et al. (2013).

In the future, we will carry out further in-depth feature engineering for the task of DSynt-parsing. It proved to be crucial in semantic role labelling and dependency parsing (Che et al., 2009; Ballesteros and Nivre, 2012); we expect it be essential for our task as well. Furthermore, we will join surface syntactic and deep-syntactic parsing we kept so far separate; see, e.g., (Zhang and Clark, 2008; Lluís et al., 2013; Bohnet and Nivre, 2012) for analogous proposals. Further research is required here since although joint models avoid error propagation from the first stage to the second, overall, pipelined models still proved to be competitive; cf. the outcome of CoNLL shared tasks.

The deep-syntactic parser described in this paper is available for downloading at `https://code.google.com/p/deepsyntacticparsing/`.

Acknowledgements

This work has been supported by the European Commission under the contract number FP7-ICT-610411. Many thanks to the three anonymous COLING reviewers for their very helpful comments and suggestions.

[9] The motivation to work with DSyntS instead of SSyntS is thus similiar to the motivation of the authors of the *Abstract Meaning Representation* (AMR) for Machine Translation (Banarescu et al., 2013), only that AMRs are considerably more semantic than DSyntSs.

References

Alfred V. Aho. 1972. *The theory of parsing, translation and, compiling*. Prentice Hall, Upper Saddle River, NJ.

Miguel Ballesteros and Joakim Nivre. 2012. MaltOptimizer: A System for MaltParser Optimization. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 12)*.

L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop & Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *EMNLP-CoNLL*.

O. Bojar, S. Cinková, and J. Ptáček. 2008. Towards English-to-Czech MT via Tectogrammatical Layer. *The Prague Bulletin of Mathematical Linguistics*, 90:57–68.

Mathias Buch-Kromann. 2003. The Danish dependency treebank and the dtag treebank tool. In *2nd Workshop on Treebanks and Linguistic Theories (TLT), Sweden*, pages 217–220.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 149–164.

Aoife Cahill, Michael Burke, Ruth O'Donovan, Stefan Riezler, Josef van Genabith, and Andy Way. 2008. Wide-coverage deep statistical parsing using automatic dependency structure annotation. *Computational Linguistics*, 34(1):81–124.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: A Library for Support Vector Machines*. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

Wanxiang Che, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin, and Ting Liu. 2009. Multilingual dependency-based syntactic and semantic parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 49–54, Boulder, Colorado, June. Association for Computational Linguistics.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33:493–552.

R. Curry. 1961. Some logical aspects of grammatical structure. In R. Jakobson, editor, *Structure of Language and Its Mathematical Aspects*, pages 56–68. American Mathematical Society, Providence, RI.

Ph. de Groote. 2001. Towards abstract categorial grammar. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Charles J. Fillmore, Collin F. Baker, and Hiroaki Sato. 2002. The FrameNet database and software tools. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, volume IV, Las Palmas. LREC, LREC.

A. Gesmundo, J. Henderson, P. Merlo, and I.Titov. 2009. Latent variable model of synchronous syntactic-semantic parsing for multiple languages. In *CoNLL 2009 Shared Task., Conf. on Computational Natural Language Learning*, pages 37–42, Boulder, Colorado, USA.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeňk Žabokrtský. 2006. Prague Dependency Treebank 2.0. Linguistic Data Consortium, Philadelphia.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task*, pages 1–18.

J. Hockenmeier. 2003. Parsing with generative models of predicate-argument structure. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 359–366, Sapporo, Japan.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012. Who did what to whom? a contrastive study of syntacto-semantic dependencies. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 2–11, Jeju, Republic of Korea, July. Association for Computational Linguistics.

R. Johansson and P. Nugues. 2007. Extended constituent-to-dependency conversion for english. In J. Nivre, H.-J. Kaalep, K. Muischnek, and M. Koit, editors, *Proceedings of NODALIDA 2007*, pages 105–112, Tartu, Estonia.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *CoNLL 2008: Proceedings of the Twelfth Conference on Natural Language Learning*, pages 183–187, Manchester, United Kingdom.

H. Kamp and U. Reyle. 1993. *From Discourse to Logic*. Kluwer Academic Publishers, Dordrecht, NL.

R.T. Kasper and W.C. Rounds. 1986. A logical semantics for feature structures. In *Proceedings of the 24th annual meeting on Association for Computational Linguistics*, pages 257–266.

Václav Klimeš. 2006. *Analytical and Tectogrammatical Analysis of a Natural Language*. Ph.D. thesis, UFAL, MFF UK, Prague, Czech Republic.

Xavier Lluís, Xavier Carreras, and Lluís Màrquez. 2013. Joint arc-factored parsing of syntactic and semantic dependencies. *Transactions of the Association for Computational Linguistics*, pages 219–230.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97.

Igor Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press.

Igor Mel'čuk. 1996. Lexical functions: A tool for the description of lexical relations in the lexicon. In L. Wanner, editor, *Lexical functions in lexicography and natural language processing*, pages 37–102. Benjamins Academic Publishers, Amsterdam.

Igor Mel'čuk. 2013. *Semantics: From meaning to text, Volume 2*. Benjamins Academic Publishers, Amsterdam.

Simon Mille, Alicia Burga, Gabriela Ferraro, and Leo Wanner. 2012. How does the granularity of an annotation scheme influence dependency parsing performance? In *Conference on Computational Linguistics, COLING 2012*.

Simon Mille, Alicia Burga, and Leo Wanner. 2013. AnCora-UPF: A Multi-Level Annotation of Spanish . In *Proceedings of the Second International Conference on Dependency Linguistics (DEPLING 2013).*

Yusuke Miyao. 2006. *From Linguistic Theory to Syntactic Analysis: Corpus-Oriented Grammar Development and Feature Forest Model*. Ph.D. thesis, University of Tokyo.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007*, pages 915–932.

Stephan Oepen. 2002. *Collaborative Language Engineering: A Case Study in Efficient Grammar-based Processing*. Stanford Univ Center for the Study.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank. *Computational Linguistics*, 31:71–106.

Owen Rambow and Aravind Joshi. 1997. A formal look at dependency grammar and phrase structure grammars, with special consideration of word-order phenomena. In L. Wanner, editor, *Recent Trends in Meaning-Text Theory*, pages 167–190. Benjamins Academic Publishers, Amsterdam.

W.C. Rounds. 1970. Mappings and grammars on trees. *Mathematical Systems Theory*, 4(3):257–287.

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

M. Taulé, M. Antònia Martí, and Marta Recasens. 2008. Ancora: Multilevel annotated corpora for Catalan and Spanish. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA).

J.W. Thatcher. 1970. Generalized sequential machine maps. *Journal of Computer and System Sciences*, 4(4):339–367.

F. Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium*, pages 398–403, Beijing, China.

Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, pages 888–896, Columbus, Ohio, June. Association for Computational Linguistics.

Modeling Newswire Events using Neural Networks for Anomaly Detection

Pradeep Dasigi
Language Technologies Institute
5000 Forbes Avenue
Pittsburgh, PA 15213
USA
pdasigi@cs.cmu.edu

Eduard Hovy
Language Technologies Institute
5000 Forbes Avenue
Pittsburgh, PA 15213
USA
hovy@cmu.edu

Abstract

Automatically identifying anomalous newswire events is a hard problem. We discuss the complexity of the problem and introduce a novel technique to model events based on recursive neural networks to represent events as composition of their semantic arguments. Our model learns to differentiate between normal and anomalous events. We model anomaly detection as a binary classification problem and show that the model learns useful features to classify anomaly. We use headlines from the weird news category publicly available on newswire websites to extract anomalous training examples and those from Gigaword as normal examples. We evaluate the classifier on human annotated data and obtain an accuracy of 65.44%. We also show that our model is at least as competent as the least competent human annotator in anomaly detection.

1 Introduction

Understanding events is a fundamental prerequisite for deeper semantic analysis of language. We introduce the problem of automatic anomalous event detection in this paper and propose a novel event model that can learn to differentiate between normal and anomalous events. We generally define anomalous events as those that are unusual compared to the general state of affairs and might invoke surprise when reported. For example, given the event mention in the following sentence

Man recovering after being shot by his dog.

one might think it is strange because *dogs* are not expected to shoot *men*. But the mentions

Man recovering after being shot by cops.

Man recovering after being bitten by a dog.

are not as unusual as the previous one. While all three sentences are equally valid syntactically, and it is not unclear what any of them means, it is our knowledge about the role fillers —both individually and specifically in combination— that enables us to differentiate between normal and anomalous events. Hence we hypothesize that *anomaly is a result of unexpected or unusual combination of semantic role fillers*. Given this idea, an automatic anomaly detection algorithm has to encode the goodness of semantic role filler coherence.

It has to be noted that event level anomaly is not the same as semantic incoherence. An event constructed by randomly choosing words to form each of the semantic arguments is not anomalous since we cannot argue whether the event is normal or anomalous when it is unclear what the event means. Hence, we define anomalous events to be the sub class of those that are semantically coherent, but are unusual only based on real world knowledge.

Automatic anomalous event detection is a hard problem since determining what a good combination of role fillers requires deep semantic and pragmatic knowledge. Moreover, manual judgment of anomaly

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1414–1422, Dublin, Ireland, August 23-29 2014.

itself may be difficult and people often may not agree with each other in this regard. We describe the difficulty in human judgment in greater detail in Section 4.4. Automatic detection of anomaly requires encoding complex information, which has to be composed from the semantics of the individual words in the sentence. A fundamental problem in doing so is the sparsity in semantic space due to the discrete representations of meaning of words.

In this paper, we describe an attempt to model newswire events as a composition of the predicate with its semantic arguments. Our approach is based on the recent models used for semantic composition using recursive neural networks (RNN). It has been previously shown by Socher et al. (2010) and Socher et al. (2013b) among others that RNN can effectively deal with sparsity in semantic space by representing meaning at a higher level of abstraction than the surface forms of words, and thus being able to learn more general patterns. These models are very relevant to modeling event semantics because the sparsity problem ranges from polysemy and synonymy at the lexical semantic level to entity and event co-reference at the discourse level.

2 Background

2.1 Selectional Preference and Thematic Fit

Selectional preference, a notion introduced by Wilks (1973), refers to the phenomenon of the predicate and the fillers of its arguments affecting the likelihood of fillers of other arguments. Thus the idea is that predicate and the role fillers "prefer" some fillers for other roles. For example, given that the predicate is *writes*, the agent *author* prefers the patient *book*, while the agent *programmer* prefers the patient *code*. This idea is used by Elman (2009), and is very similar to the role-filler composition that we use for anomaly detection.

Erk et al. (2010) also model selectional preferences using vector spaces. They measure the goodness of the fit of a noun with a verb in terms of the similarity between the vector of the noun and some "exemplar" nouns taken by the verb in the same argument role. Baroni and Lenci (2010) also measure selectional preference similarly, but instead of exemplar nouns, they calculate a prototype vector for that role based on the vectors of the most common nouns occurring in that role for the given verb. Lenci (2011) builds on this work and models the phenomenon that the expectations of the verb or its role-fillers change dynamically given other role fillers.

2.2 Recursive Neural Networks

Recursive Neural Networks (RNN), first introduced by Goller and Kuchler (1996), are multilayer neural network models used for efficient processing of structured objects of arbitrary shape. These have been successfully used for modeling semantics of sentences of arbitrary length by Socher et al. (2010), for sentiment analysis by Socher et al. (2013b), for syntactic parsing by Socher et al. (2013a) and for learning morphologically aware word representations by Luong et al. (2013). RNN are attractive because they can encode compositions of meaning guided by syntax or some other linguistic structure known a priori. Moreover, they provide flexibility in terms of learning composition weights based on supervised or unsupervised objectives. Consequently RNN learn feature representations depending on the task. Hence, this is a good choice for modeling event composition.

In its simplest form, an RNN processes information backed by a Directed Acyclic Graph (DAG), where each node represents a neural network with the same parameters. The output produced at each intermediate step of encoding usually has the same dimensionality as each of the inputs, hence RNN projects the representation of a structure of arbitrary length into the same space as the inputs. This property is what makes RNN recursive. An example RNN with a binary DAG (tree) structure is shown in Figure 1. The activation from each neural network node is

$$c = g(y_1 \| y_2) = Sg(W(y_1 \| y_2) + b)$$

where $\|$ represents concatenation of vector representations of the inputs, $y_1, y_2 \in \mathbb{R}^{n \times 1}$ are the inputs, $W \in \mathbb{R}^{n \times 2n}$ is the composition weight matrix and $b \in \mathbb{R}^{n \times 1}$ is the bias. Sg is a element wise sigmoid

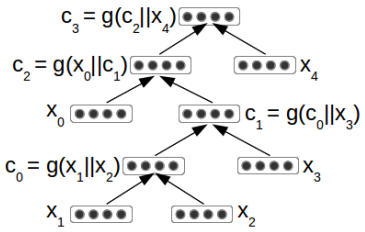


Figure 1: Example of a Recursive Neural Network backed by a binary tree

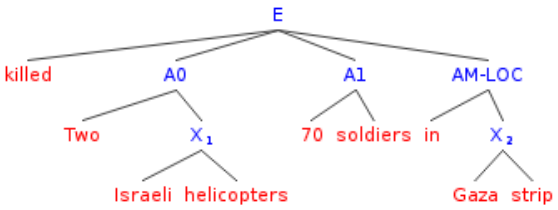


Figure 2: Example of an event tree

function. Apart from encoding the composition, RNN also produce a score of composition

$$s = S^{\mathsf{T}} c$$

where $S \in \mathbb{R}^{n \times 1}$ is a scoring operator and s is a score that shows how good the composition is. (Collobert et al., 2011) take an unsupervised approach to training RNN for semantic composition based on the contrastive estimation technique proposed by (Smith and Eisner, 2005) and assuming that any word and its context is a positive example and a random word in the same context is a negative training example. (Socher et al., 2013b) among others use a supervised objective that is based on the label error at the topmost node in the RNN. The parameters of the simplest model are W, b and S. For representation learning, the inputs x_i are also made parameters. Goller and Kuchler (1996) propose Backpropagation through structure (BPTS), that respects the underlying DAG structure during backpropagation of gradients.

3 Neural Event Model

We define an event as the pair $(V, \mathbf{A})$, where V is the predicate or a semantic verb[1], and $\mathbf{A}$ is the set of its semantic arguments like agent, patient, time, location, so on. Our aim is to obtain a vector representation of the event that is composed from representations of individual words, while explicitly guided by the semantic role structure. This representation can be understood as an embedding of the event in an event space.

Neural Event Model (NEM) is a kind of RNN that is guided by a tree representation of events like the one shown in Figure 2. The edges connected to the root of the tree correspond to the predicate and its semantic roles (arguments). All the other edges form binary sub-trees of arguments. NEM is a supervised model that learns to differentiate between anomalous and normal events by classifying the event embeddings. The inputs to NEM are the semantic arguments, and the representations of words in each argument. We recursively compose the words in each argument to obtain argument level representations, which are then composed to obtain an event embedding.

Intra-argument composition (called argument composition henceforth) is unsupervised, and we use contrastive estimation to learn the parameters. The structure of the binary tree backing argument composition is determined dynamically, composing at each stage the two nodes which give the best composition

[1]By semantic verb, we mean an action word whose syntactic category is not necessarily a verb. For example, in *Terrorist attacks on the World Trade Center..*, *attacks* is not a verb but is still an action word.

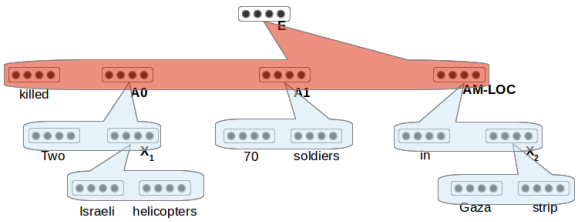


Figure 3: Neural Event Model: Encoding

score. Inter-argument composition (called event composition henceforth) is supervised and we use label error to learn the parameters. Figure 3 shows how NEM encodes the event shown in Figure 2. The blue boxes show argument composition and the red box shows event composition.

3.1 Training

NEM is trained in two phases. The first, argument composition, is unsupervised while the second, event composition, is supervised.

3.1.1 Argument Composition

An argument composition node takes inputs of dimensionality $2n$ and produces an composed output representation of dimensionality n and a composition score. Accordingly, we define the node in terms of the parameters $\theta_{arg} = \{W_{arg} \in \mathbb{R}^{n \times 2n}; b_{arg}, S_{arg} \in \mathbb{R}^{n \times 1}; V\}$ where W_{arg}, b_{arg} and S_{arg} are the composition weight, bias and the scoring operators respectively as described previously, and V is the set of representations of all the words in the vocabulary. All nodes performing argument composition use the same parameters. Training is done in contrastive estimation fashion and the objective is

$$\arg\min_{\theta_{arg}} J_{arg} = \arg\min_{\theta_{arg}} max(0, 1 - s + s_c)$$

where s is the score of the composition of the entire argument produced by the root node of the argument, and s_c is the score produced by randomly replacing one of the words in the argument at a time. The structure of the binary tree backing each argument is determined dynamically. This is done by starting with leaf nodes in the tree for each of the words in the argument, comparing the composition scores of every pair of adjacent leaf nodes, and actually composing the pair that gives the highest score, which gives a new node. The process is repeated until we build a complete binary tree for each argument.

3.1.2 Event Composition

Event composition takes argument representations and produces the event representation and label indicating whether the event is normal or anomalous. We define the event composition node in terms of the parameters $\theta_{event} = \{W_{event} \in \mathbb{R}^{n \times kn}; b_{event}, L_{event} \in \mathbb{R}^{n \times 1}\}$ where k is the number of semantic arguments per event. L_{event} is the label operator. The objective of this phase is

$$\arg\min_{\theta_{event}} J_{event} = \arg\min_{\theta_{event}} \left(-l \log h(e) + (1 - l) \log(1 - h(e))\right)$$

where l is the reference binary label indicating whether the event is normal or anomalous, e is the event representation and $h(e)$ is the output of the logistic function. Concretely,

$$h(e) = \frac{1}{1 + e^{-L_{event}^{\intercal} e}}$$

We implement the functions and perform stochastic gradient descent using Theano (Bergstra et al., 2010).

4 Experiments

4.1 Event Extraction

We extract events by running the Semantic Role Labeling (SRL) tool in SENNA (Collobert et al., 2011). SENNA uses PropBank (Palmer et al., 2005) style semantic tags. We consider only the roles *A0*, *A1*, *AM-TMP* and *AM-LOC* as the arguments of our events[2]. For example, the event in the tree shown in Figure 2 is extracted from the sentence

Two Israeli helicopters killed 70 soldiers in Gaza strip.

and SENNA identifies the following as the semantic roles

verb:*killed* ***A0:****Two Israeli helicopters* ***A1:****70 soldiers* ***AM-LOC:****in Gaza strip*

4.2 Data

Since the second phase of training NEM is supervised, we need newswire events that are normal and those that are anomalous. We crawl 3684 "weird news" headlines available publicly on the website of NBC news[3], such as the following:

- *India weaponizes world's hottest chili.*
- *Man recovering after being shot by his dog.*
- *Thai snake charmer puckers up to 19 cobras.*

We assume that the events extracted from this source, called NBC Weird Events (NWE) henceforth, are anomalous for training. NWE contains 4271 events extracted using SENNA's SRL. We use 3771 of those events as our negative training data, and the remaining for testing. Similarly, we extract events also from headlines in the AFE section of Gigaword, called Gigaword Events (GWE) henceforth. We assume these events are normal. To use as positive examples for training event composition, we sample roughly the same number of events from GWE as our negative examples from NWE. It has to be noted that each headline may contain multiple events and some may not contain events at all.

For argument composition, we use about 100k whole sentences from AFE headlines and the weird news headlines from which NWE are extracted. Since we are training argument composition, we do not use the event structure in the first phase. It has to be noted that all our training data are easily available and do not require any human annotation.

We test the performance of NEM on 1003 events which are not part of the training dataset. These events are sampled with equal probabilities from NWE and GWE and are human annotated for anomaly. Section 4.4 has details of the annotation task.

4.3 Word Vector Initialization

We initialize the vector representations of the words in our vocabulary using the embeddings available in SENNA 3.0 (Collobert et al., 2011) if available, and randomly if not. For event composition, if the event does not have a specific role filler, we input a zero vector for the role.

4.4 Annotation

We post the annotation of the test set containing 1003 events as Human Intelligence Tasks (HIT) on Amazon Mechanical Turk (AMT). We break the task into 20 HITs and ask the workers to select one of the four options - *highly unusual*, *strange*, *normal* and *cannot say* for each event. We ask them to select *highly unusual* when the event seems too strange to be true, *strange* if it seems unusual but still plausible, and *cannot say* only if the information present in the event is not sufficient to make a decision. We present each event along with the original headline and the semantic arguments. Along with marking

[2]These four types cover about 85% of all arguments in our training and test datasets.

[3]`http://www.nbcnews.com/html/msnbc/3027113/3032524/4429950/4429950_1.html`

Total number of annotators	22
Normal annotations	56.3%
Strange annotations	28.6%
Highly unusual annotations	10.3%
Cannot Say annotations	4.8%
Avg. events annotated per worker	344
4-way Inter annotator agreement (α)	0.34
3-way Inter annotator agreement (α)	0.56

Table 1: Annotation Statistics

one of the four options above, if an event is *strange* or *highly unusual*, we ask the annotators to select the parts of the headline that make it so. Since there can be multiple events in the headline, the annotators decision regarding the parts of the sentence that cause anomaly help us identify which particular event in the headline is anomalous.

Table 1 shows some statistics of the annotation task. We compute the Inter Annotator Agreement (IAA) in terms of Kripendorff's alpha (Krippendorff, 1980). The advantage of using this measure instead of the more popular Kappa is that the former can deal with missing information, which is the case with our task since annotators work on different overlapping subsets of the test set. The 4-way IAA shown in the table corresponds to agreement over the original 4-way decision (including *cannot say*) while the 3-way IAA is measured after merging the *highly unusual* and *strange* decisions.

Additionally we use MACE (Hovy et al., 2013) to assess the quality of annotation. MACE models the annotation task as a generative process of producing the observed labels conditioned on the true labels and the competence of the annotators, and predicts both the latent variables. The average of competence of annotators, a value that ranges from 0 to 1, for our task is 0.49 for the 4-way decision and 0.59 for the 3-way decision.

We generate true label predictions produced by MACE, discard the events for which the prediction remains to be *cannot say*, and use the rest as reference for evaluating NEM, which is described in Section 4.5. This leaves 949 events as our reference dataset, of which only 41% of the labels are *strange* or *highly unusual*. It has to be noted that even though our test set has equal size samples from both NWE and GWE, the true distribution is not uniform.

Language Model Separability Given the annotations, we test to see if the sentences corresponding to anomalous events can be separated from normal events by simpler features. We build a n-gram language model from the training data set used for argument composition and measure the perplexity of the sentences in the test set. Figure 4 shows a comparison of the perplexity scores for different labels. If the n-gram features are enough to separate different classes of sentences, one would expect the sentences corresponding to *strange* and *highly unusual* labels to have higher perplexity ranges than *normal* sentences, because the language model is built from a dataset that is expected to have a distribution of sentences where majority of them contain normal events. As it can be seen in Figure 4, except for a few outliers, most data points in all the categories are in similar perplexity ranges. Hence, sentences with different labels cannot be separated based on an n-gram language model features.

4.5 Evaluation

We evaluate the performance of event composition by comparing the predicted labels from the classifier against the ones given by MACE. We merge the two anomaly classes and calculate accuracy of the binary classifier, and the precision and recall of anomaly detection.

Baseline We compare the performance of our model against a baseline that is based on how well the semantic arguments in the event match the selectional preferences of the predicate. We measure selectional preference using Point-wise Mutual Information (PMI) (Church and Hanks, 1990) of the head words of each semantic argument with the predicate. The baseline model is built as follows. We perform

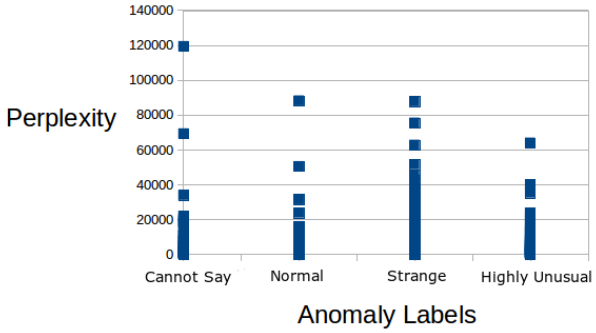


Figure 4: Comparison of perplexity scores for different labels

		NEM	Baseline
Accuracy		65.44%	45.22%
Anomalous	Precision	56.55%	36.30%
	Recall	48.22%	59.50%
Normal	Precision	64.62%	42.08%
	Recall	77.66%	33.60 %

Table 2: Classification Performance and Comparison with Baseline

dependency parsing using MaltParser (Nivre et al., 2007) on the sentences in the training data used in the first phase of training to obtain the head words of the semantic arguments. We then calculate the PMI values of all the pairs $< h_A, p >$ where h is the head word of argument A and p is the predicate of the event. For training our baseline classifier, we use the labeled training data from the event composition phase. The features to this classifier are the PMI measures of the $< h_A, p >$ pairs estimated from the larger dataset. The classifier thus trained to distinguish between anomalous and normal events is applied to the test set.

Table 2 shows the results and a comparison with the PMI based baseline. The accuracy of the baseline classifier is lower than 50%, which is the expected accuracy of a classifier that assigns labels randomly. The precision of that random classifier in predicting anomalous events is expected to be 41%, since that is the percentage of anomaly labels in our reference set as described in Section 4.4. The accuracy of NEM is higher than the baseline model. One possible reason for the PMI based baseline having higher recall in predicting anomaly and lower precision is that the statistics estimated from larger training data cannot be generalized to the test set due to sparsity issues. This indicates the advantage of using continuous representations at a higher level of abstraction as features for classification.

To further compare NEM with human annotators, we give to MACE, the binary labels produced by NEM along with the annotations and measure the competence. For the sake of comparison, we also give to MACE, a list of random binary labels as one of the annotations to measure the competence of a hypothetical worker that made random choices. These results are reported in Table 3. It can be seen that the performance of NEM is comparable at least to the least competent human.

5 Discussion and Future Work

The two evaluation experiments show that the neural network does learn to distinguish between normal and anomalous events. Future improvements to this model will include better event extraction techniques.

Since the current approach is supervised, the training data size for learning event composition is limited. We plan to develop unsupervised approaches that can learn good models of normal events, and detect anomalies based on how well new events fit in the model. One possible approach is to do learning

Human average	0.59
Human highest	0.70
Human lowest	0.26
Random	0.02
NEM	0.26

Table 3: Anomaly Detection Competence

based on contrastive estimation in the second phase as well. The assumption behind taking this approach for learning is that a randomly generated data point is likely to be a negative example, which is not necessarily true for learning event composition. Generating malformed events that are syntactically valid but anomalous without much human effort can greatly help in developing such an unsupervised algorithm.

One important aspect of anomaly that is currently not handled by NEM is the level of generality of the concepts the events contain. Usually more general concepts cause events to be more normal since they convey lesser information. For example, an American soldier shooting another American soldier may be considered unusual, while a soldier shooting another soldier may not be as unusual, and at the highest level of generalization, a person shooting another person is normal. This information of generality has to be incorporated into the event model. This can be achieved by integrating real world knowledge from knowledge bases like Wordnet (Miller, 1995) or from corpus statistics like the work by Lin (1998) into the event model. Bordes et al. (2011) learn continuous representations of entities and relations in knowledge bases. More recently, an alternative approach for doing the same was proposed by Chen et al. (2013). These representations can greatly help modeling events.

Finally, the idea of modeling event composition can help processing event data in general and can be applied to other tasks like finding co-referent events.

6 Conclusion

We introduced the problem of anomalous newswire event detection and illustrated its difficulty. Our approach is similar to the ones successfully used for modeling semantic composition. We showed that while our event composition model does learn to distinguish between normal and anomalous events, there is scope for improved models that can effectively incorporate real world information and can be trained in an unsupervised fashion. We note that in general event composition is more difficult than traditional semantic composition since the former also deals with pragmatics. Consequently the set of nonsensical events is different from the set of anomalous sentences, and while meaningless events and well composed normal events are two ends of the semantic spectrum, semantically valid anomalous events lie somewhere between them.

Acknowledgements

This research was supported in part by DARPA grant FA8750-12-2-0342 funded under the DEFT program.

References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, volume 4.

Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. 2011. Learning structured embeddings of knowledge bases. In *AAAI*.

Danqi Chen, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2013. Learning new facts from knowledge bases with neural tensor networks and semantic word vectors. *arXiv preprint arXiv:1301.3618*.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Jeffrey L Elman. 2009. On the meaning of words and dinosaur bones: Lexical knowledge without a lexicon. *Cognitive science*, 33(4):547–582.

Katrin Erk, Sebastian Padó, and Ulrike Padó. 2010. A flexible, corpus-driven model of regular and inverse selectional preferences. *Computational Linguistics*, 36(4):723–763.

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 347–352. IEEE.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with mace. In *Proceedings of NAACL-HLT*, pages 1120–1130.

Klaus Krippendorff. 1980. *Content analysis: An introduction to its methodology*. Sage Publications (Beverly Hills).

Alessandro Lenci. 2011. Composing and updating verb argument expectations: A distributional semantic model. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 58–66. Association for Computational Linguistics.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics.

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. *CoNLL-2013*, 104.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9.

Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. 2013a. Parsing with compositional vector grammars. In *In Proceedings of the ACL conference*. Citeseer.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642.

Yorick Wilks. 1973. Preference semantics. Technical report, DTIC Document.

Million-scale Derivation of Semantic Relations from a Manually Constructed Predicate Taxonomy

Motoki Sano[*] **Kentaro Torisawa**[†] **Julien Kloetzer**[‡] **Chikara Hashimoto** [§]
István Varga[¶] **Jong-Hoon Oh**[‖]
[*†‡§‖] National Institute of Information and Communications Technology, Kyoto, 619-0289, Japan
[¶]NEC Knowledge Discovery Research Laboratories, Kanagawa, 211-8666, Japan
{[*] msano, [†] torisawa, [‡] julien, [§] ch, [‖] rovellia}@nict.go.jp, [¶] vistvan@az.jp.nec.com

Abstract

We manually created a semantic taxonomy called *Phased Predicate Template Taxonomy* (*PPTT*) that covers 12,023 predicate templates (i.e., predicates with one argument slot like "*rescue X*") and derived from it various semantic relations between these templates on a million-instance scale (70%-80% precision level). The derived relations include entailment (e.g., *rescue X*$\supset$*X is alive*), happens-before (e.g., *buy X*$\Rightarrow$*drink X*), and a novel relation type *anomalous obstruction* (e.g., *X is sold out*$\rightsquigarrow$*cannot buy X*). Such derivation became possible thanks to PPTT's design and the use of statistical methods.

1 Introduction

Databases of various semantic relations between natural language expressions are indispensable knowledge for many NLP applications. For instance, entailment relations are crucial in information extraction and QA (Dagan et al., 2009; Weisman et al., 2012; Berant et al., 2012; Turney and Mohammad, 2014). Temporal relations such as happens-before (Chklovski and Pantel, 2004b; Regneri et al., 2010) are important for enhancing deep semantic processing. A problem, however, is that it is difficult to acquire those relations with a broad coverage. Although many sophisticated machine learning techniques have been applied to various kinds of corpora for this task (Szpektor et al., 2007; Chambers and Jurafsky, 2008; Hashimoto et al., 2009; Chambers and Jurafsky, 2009; Hashimoto et al., 2012; Talukdar et al., 2012; Kloetzer et al., 2013), no satisfactory coverage has been achieved, probably due to data sparseness in the input data. In this work we take a completely different approach: we manually construct a semantic lexicon called *Phased Predicate Template Taxonomy* (*PPTT*), and derive various types of semantic relations on a large-scale by using it. Our target language is Japanese, but examples are given in English for simplicity throughout this paper.

PPTT is a taxonomy of predicate templates (predicates with one argument slot like *rescue X*, "Template" hereafter) that classifies templates according to *phases of story* concerning an entity denoted by *X*. In the story, or the "life" of the entity *X*, *X* can be *anticipated*, *created*, then *execute* its function and finally it may *collapse* and become *deficient*. Anticipation, creation, execution, collapse, deficiency of *X* can be seen as such *phases of story* concerning *X*, and PPTT classifies templates into 41 semantic classes each of which corresponds to a phase. In other words, PPTT provides a way to describe the stories of various entities that constitute this world, and we believe that PPTT (partly) reflects how we understand the world and its entities. Accordingly, PPTT can also provide a way to derive various semantic knowledge about this world such as the happens-before relation between events involving an entity, e.g., since the creation phase usually occurs before the execution phase, *invent X* (creation phase) is likely to happen-before *use X* (execution phase). In addition, entailment relations can be derived: since the creation phase of an object *X must* have occurred if *X* is in its execution phase, it implies that *use X* is likely to entail *invent X*.

In addition, there are ups and downs in stories; some entities suffer setbacks in their stories. PPTT describes such "ups and downs" by means of a recently proposed semantic polarity, *excitation* (Hashimoto

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1423–1434, Dublin, Ireland, August 23-29 2014.

et al., 2012). Excitation classifies templates into *excitatory*, *inhibitory*, and *neutral*; an excitatory template like *install X* and *buy X* indicates that the main function, effect, purpose or role of the entity referred to by the *X* of the template is *activated*, *enhanced*, or *prepared*,[1] while an inhibitory template like *uninstall X* and *X is canceled* roughly indicates that it is *deactivated* or *suppressed*. Neutral templates are neither excitatory nor inhibitory (e.g., *consider X*). Roughly speaking, an excitatory template expresses the events that contribute to *turn on* the function of X, while an inhibitory template expresses the events that contribute to *turn off* or *not to turn on* the function of X. Then, in PPTT, excitatory and inhibitory respectively correspond to "ups" and "downs" in the story of *X*. The phases in PPTT are marked according to these ups and downs. Accordingly, PPTT can derive many antonymous contradiction pairs like *install X*⇔*uninstall X*, as Hashimoto et al. did, though we omit the detail for space limitation. Moreover, PPTT can derive a huge volume of *anomalous obstruction*, a contradiction-like novel semantic relation that we propose in this paper, like *X is canceled*⇝*(cannot) buy X* and *X is sold out*⇝*(cannot) buy X*, which indicate that if *X* is canceled or sold out, you cannot buy X. Anomalous obstruction should be used for Why-type QA (Oh et al., 2013), as well as a novel system that warns a user who wants to buy a commercial product that the product is started to be *sold out* or *canceled* in various e-commerce sites without any application-specific coding.

As suggested, a story has a temporal order between its phases, which we call the *canonical temporal order*. In addition, some phases in a story would *enable* or *necessitate* another phase in the same story to occur. In PPTT, these relations are embodied in various temporal-semantic links between phases. Note that each link between two phases does not *guarantee* that every possible pair of templates taken from the two phases has such semantic relations; it just indicates that there exists such *tendencies*. Despite the absence of the guarantee, PPTT's links enable a million-scale derivation of semantic relations with the help of distributional similarity. In existing resources such as WordNet (Fellbaum, 1998), the links are assumed to be 100% correct, but it would be hard to have such absolutely correct links in a million-scale. Hence, we believe that our *approximate* links are more useful for a large-scale relation derivation.

Note that the goal of our PPTT project is to derive a wide range of semantic relations on a large scale, rather than to complete a comprehensive template taxonomy. As such, PPTT lacks some templates as described in later sections. Nevertheless, we believe that our design brings much more good than harm, since we could generate various semantic relations on a million scale thanks to PPTT. Our experimental results show that we can derive 4.4 million happens-before relation instances with 79.5% precision, 0.5 million entailment relation instances with 70.0% precision, and one million anomalous obstruction relation instances with 73.5% precision. Constructing the PPTT taxonomy requires a manual labor cost, which amounted to three man-months in our case; however, we believe that this cost is *lower* than the cost for developing highly-precise automatic acquisition methods for all of happens-before, entailment, contradiction, and anomalous obstruction relations.

We plan to release PPTT and the derived relation instances after the manual annotation of the derived instances to the NLP community.

2 Related Works

PPTT might resemble other semantic lexicons created in the long history of NLP (Levin, 1993; Kipper et al., 2006; Fellbaum, 1998; Bond et al., 2009; Fillmore, 1976; Baker et al., 1998; Halliday, 1985; Pustejovsky et al., 2003; Puscasu and Mititelu, 2008; Bejar et al., 1991; Jurgens et al., 2012). PPTT is different in that it primarily aims at deriving various types of semantic relations on a large scale exploiting the notion of the *phase of story*, rather than being a comprehensive taxonomy like those existing semantic lexicons. As a result, PPTT can derive more varieties of semantic relations between templates than any one of those existing lexicons. From **WordNet** (Fellbaum, 1998; Bond et al., 2009), we can derive entailment and contradiction relations using synsets and synset-links that represent relations such as 'troponym', 'antonym' and 'entailment'. However, happens-before and anomalous obstruction relations

[1]The above definition is slightly different from the original one in Hashimoto et al. (2012). We inserted the verb "prepared" into the original definition. This clarifies that various preparation processes for X, such as *buy X*, can be regarded as excitatory templates. We also assume that such templates as *X exists* and *have X*, which mean little more than just existence, are regarded as excitatory templates in PPTT based on the assumption that *existence* can be regarded as *preparation* for the function of X.

cannot be derived from it, since there is no information on temporal ordering except that on causality. From **VerbNet** (Levin, 1993; Kipper et al., 2006), the hyponymy/synonymy type of entailment relations may be derived using templates in the same verb classes constructed based on shared syntactic behavior, possibly with the help of statistical methods. However, the other types of relations that can be derived from PPTT cannot be derived from VerbNet, since there is no link representing relationships between the verb classes. **FrameNet** (Fillmore, 1976; Baker et al., 1998) was used to derive hyponymy/synonymy types of entailment (Coyne and Rambow, 2009; Aharon et al., 2010) using information such as a Frame-to-frame relation 'Inheritance' (*is-a* relation). In addition, happens-before relations can be derived using 'Precedes' (*Later-Earlier* relations). However, since it does not contain semantic constraints like *enablement* and *necessity* that PPTT contains, it is not trivial to derive presupposition type of entailment or anomalous obstruction instances from it. **TimeML** (Pustejovsky et al., 2003; Puscasu and Mititelu, 2008) contains various temporal information and can be used to derive *context-dependent* happens-before relations such as the relation between "leaves" and "will not hear" in the sentence "If Graham leaves today, he will not hear Sabine" through TLINK (Pustejovsky et al., 2003) annotated manually; thus, it is difficult to derive *context-independent* relations from it, while they can be derived from PPTT. Besides, since it covers only temporal information, it is difficult to derive other types of relations from it. From **Bejar et al.'s semantic relation taxonomy of lexical pairs** (Bejar et al., 1991; Jurgens et al., 2012), using semantic relation categories such as "act: act attribute" (e.g., creep:slow), lexical entailment relations were extracted (Turney and Mohammad, 2014). However, it is not trivial to derive happens-before or anomalous obstruction relations from it since it does not contain information on temporal sequences between verbs.

Furthermore, our work differs from automatic methods for extracting temporal or causal relations (Szpektor et al., 2007; Chambers and Jurafsky, 2008; Chambers and Jurafsky, 2009; Talukdar et al., 2012; Hashimoto et al., 2012; Hashimoto et al., 2014) in that our method does not require that target pairs co-occur in a document, unlike the previous methods. Hence, our method is likely to be immune to data sparseness. We could actually derive a wide range of relation instances that were rarely written in documents because they were too commonsensical (e.g., *X is constructed* happens-before *sew (something) at X*). Needless to say, such commonsensical knowledge is often needed to develop intelligent systems.

3 PPTT Design

In PPTT, templates are organized hierarchically into three levels. In each level, there are classes that correspond to phases of stories, which we call Level-0 (L0), Level-1 (L1), and Level-2 (L2) classes. Each template belongs to only one class at each level. In the following, we describe each level.

3.1 L0-Classes and L0-Links

First we divided the *entire story* concerning an entity *X* into five phases: *non-existence*, *existence*, *functioning*, *non-existence to existence transition* and *existence to non-existence transition*. Then we created the five L0-classes listed below, each of which corresponds to one of these five phases.

Non-existence Class The class of templates that do not entail the existence of X, e.g., *plan X*.[2]

Existence Class The class of templates that entail X's existence but does not imply the execution of its main function or the achievement of its objectives, e.g., *buy X*, *X exists*.

Functioning Class The class of templates that imply the execution of X's main function or the achievement of its objectives, e.g., *use X*, *eat X*.

Non-existence to Existence Transition Class (NET Class) The class of templates that express the transition from a situation in which X does not exist to a situation in which it exists, e.g., *manufacture X*.

[2]One might think the definition of the Non-existence Class should be "the templates that DO entail X's NON-EXISTENCE". We did not use such a definition because it would overlook many templates that are consistent with X's NON-EXISTENCE but DO NOT entail X's NON-EXISTENCE, like *order X*.

Existence to Non-existence Transition Class (ENT Class) The class of templates that express the transition from a situation in which X exists to a situation in which it does not exist, e.g., *dismantle X*.

As mentioned in the introduction, we assume a canonical temporal order among L0-classes. For instance, templates in the **NET class** (e.g., *manufacture X*) should refer to events that usually happen before those events referred to by templates in the **Existence class** (e.g., *buy X*), **Functioning class** (e.g., *use X*) and **ENT class** (e.g., *dismantle X*). We enumerated such temporal restrictions, each of which is represented by a link in Figure 1, which we call *L0-links* and used them for deriving relations. Note that we did not set any L0-link between the **Existence class** and the **Functioning class** because the events described by them may happen in various orders or have temporal overlap. For example, *X exists* should have temporal overlap with *use X*.

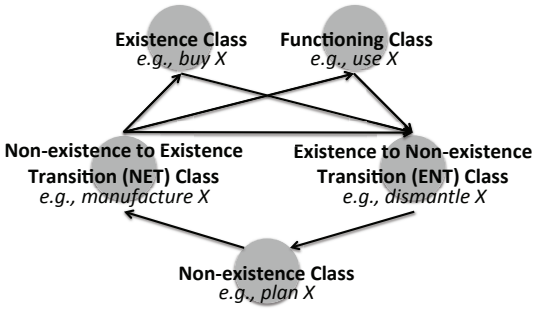


Figure 1: L0-links among L0-classes.

Of course, such *metaphysical* notions as the canonical temporal order and the phases must have many complications and exceptions. First, many templates that have the neutral excitation polarity (Hashimoto et al., 2012) did not seem to follow the canonical temporal order among L0-classes. For instance, since the neutral template *think about X* does not entail the existence of *X*, it belongs to the **Non-existence** class but one can consider *X* while *X* exists or while it is functioning or even after it is collapsed and violate canonical temporal ordering. For this reason, we excluded neutral templates from PPTT and will deal with them in a different framework as a future work. In addition, although we did not assume a temporal order between the **Existence class** and the **Functioning class**, some templates in these classes have a happens-before relation as special cases (e.g., *buy X* in the **Existence class** happens before *eat X* in the **Functioning class**). The proposed L0-links also cause problems. For instance, *order X* (**Non-Existence class**) may not always happen before *create X* (**NET class**) even though the L0-links indicate a happens-before relation between their classes. We dealt as far as possible with such cases in level 2 with L2-classes, which are finer than L0-classes. Nonetheless, we stress that the overall plausibility of the canonical temporal order among L0-classes was experimentally confirmed through the derivation of happens-before relations only using L0-links. Note that the design of the L0-classes was inspired by the Generative Lexicon (Pustejovsky, 1998) and Aristotle's *Entelecheia* (Aristotle, 1987).

3.2 L1-Classes

Next, we divided some L0-classes into L1-classes using the excitation polarity (Hashimoto et al., 2012) to introduce "ups and downs" to PPTT, which enables to capture semantic inconsistencies between templates (e.g., *install X⇔uninstall X*) and negative interaction between the events referred to by the templates in PPTT (e.g., *X is canceled⇝(cannot) hold X*). Excitation was originally proposed for recognizing contradictions and causal relations between templates and then was successfully applied to other deep semantic processing (Oh et al., 2013; Varga et al., 2013; Kloetzer et al., 2013; Hashimoto et al., 2014).

L0-class	Excitation	
	Excitatory	**Inhibitory**
Non-existence class	POTENTIAL class e.g., *plan X*	FORECLOSING class e.g., *prevent X*
Existence class	ENABLING class e.g., *buy X*	INCOMMODE class e.g., *weaken X*
Functioning class	ACTUALIZING class e.g., *X functions*	DISORDERING class e.g., *X loses*
NET class	GENERATING class e.g., *X is born*	N/A
ENT class	N/A	CORRUPTING class e.g., *destroy X*

Table 1: L1-classes.

As shown in Table 1, we divided each of three L0-classes (**Non-existence class**, **Existence class** and **Functioning class**) into two L1-classes, each of which corresponds to excitatory and inhibitory. Since the transition to an existence situation can be interpreted as an enhancement of an entity's function, we assumed that all the templates in the NET classes are excitatory because they express a transition of entity *X* from a non-existence situation to an existence situation. Similarly, we assume all the templates from the ENT class are inhibitory. Also, L1-classes do not have specific links between them beside the L0-links from their parent classes.

3.3 L2-Classes and L2-Links

Finally, we divided L1-classes into 41 L2-classes. Specifically, we first roughly grouped together semantically similar templates from the same L1-class and identified the common semantic properties among them. Note that in the rough grouping, we classified templates so that the resulting groups fit into fine-grained phases in the story concerning *X*.

After this initial grouping, we classified all the templates into the L2-classes that are listed in Table 3 alongside the classification criteria and the number of templates in each class. As the classification criteria, we used the identified common semantic properties among members of each class. Note that some L2-classes can be regarded as a subset of another L2-class. For instance, the PROHIBIT L2-class can be seen as a subset of the PREVENTION L2-class. When a template meets the classification criteria of both a subset class and its superset class, we classified it into the subset class.

We also made links called L2-links between the L2-classes. The motivation behind this is to capture finer temporal-semantic constraints that could not be specified at Level-0 and Level-1 as well as to capture the temporal-semantic constraints inside a single L0 or L1-class. For example, the temporal order between *buy X* and *eat X* is encoded in a L2-link between the ACQUISITION and EXECUTION L2-classes, while there is no L0-link between the Existence L0-class (class of *buy X*) and the Functioning L0-class (class of *eat X*). This exemplifies that the L2- and L0-links complement each other.

Each L2-link has one of the six types of temporal-semantic links that are summarized in Table 2 with the number of links of each type. The link types were designed to capture how the events referred to by the templates in a class *affect* the *occurrence or non-occurrence* of the events referred to by the templates in a class in the *past*, *present*, or *future*. C_1 and C_2 being two L2-classes, C_1's effect on the occurrence or non-occurrence of C_2 is represented by Positive ($^+$) and Negative ($^-$) links, respectively, while C_1's effect on the past, present, or future phase of *X* expressed by C_2 is represented by Past, Present, and Future links, respectively. For instance, the Past$^+$ link from the ABANDONMENT class to the ACQUISITION class indicates that a template from the ACQUISITION class (e.g., *obtain X*) *must occur* before a template from the ABANDONMENT class (e.g., *get rid of X*), and the Future$^-$ link from the PROHIBIT class to the EXECUTION class indicates that templates from the PROHIBIT class (e.g., *ban X*) *disable* templates from the EXECUTION class (e.g., *utilize X*). Notice that L2-links represent such semantic constraints as *enablement* and *necessity* in addition to temporal order, and they are useful for deriving various kinds of semantic relations including entailment and anomalous obstruction, as shown in a later section. The first author of this paper hand-labeled the links between every combination of L2-class pairs by considering the name of the classes and a few example templates in each.

	Positive	**Negative**
Past	If C_1 occurred, C_2 must have occurred. e.g.,FORGETTING $\overset{Past^+}{\rightarrow}$ RECOGNITION; *X is forgotten* $\overset{Past^+}{\rightarrow}$ *X is recognized* (55 links)	If C_1 occurred, C_2 COULD NOT have occurred. e.g.,CREATION $\overset{Past^-}{\rightarrow}$ PREVENTION; *X is generated* $\overset{Past^-}{\rightarrow}$ *X is prevented* (438 links)
Present	While C_1 is taking place, C_2 must be taking place. e.g.,INITIATION $\overset{Present^+}{\rightarrow}$ BEING; *X is started* $\overset{Present^+}{\rightarrow}$ *X exists* (73 links)	While C_1 is taking place, C_2 CANNOT take place. e.g.,ENHANCEMENT $\overset{Present^-}{\rightarrow}$ DEGRADATION; *X is enhanced* $\overset{Present^-}{\rightarrow}$ *X is deteriorated* (496 links)
Future	C_1 enables C_2 to occur. e.g.,PREPARATION $\overset{Future^+}{\rightarrow}$ EXECUTION; *X is customized* $\overset{Future^+}{\rightarrow}$ *X is executed* (90 links)	C_1 DISABLEs C_2 to occur. e.g., DEFICIENCY $\overset{Future^-}{\rightarrow}$ PROVISION; *X does not exist* $\overset{Future^-}{\rightarrow}$ *X is provided* (210 links)

Table 2: Types and numbers of L2-links in PPTT. Link direction is $C_1 \rightarrow C_2$.

Non-existence L0-class: **Potential L1-class (578) / Foreclosing L1-class (178)**	
DESIRE	entails that *X* is desired but unlike PLANNING or DEMAND, it does not entail that *X* is planned or requested, e.g., *desire X, want X* (48).
PLANNING	entails that *X* is planned but does not entail that *X* is requested. Unlike DEMAND, it does not assume that a person other than the Planner will carry out X, e.g., *plan X, conspire X* (72).
DEMAND	entails that *X* is requested. Unlike PLANNING, it assumes that a person other than the Demander will carry out X, e.g., *order X* (252).
APPROVAL	entails that *X* is approved or permitted and that there was a plan or a demand before approving, e.g., *permit X, accept X* (80).
FEAR	entails that *X* is expected and that *X* is a source of anxiety or fear, e.g., *fear X, worry about X* (13).
ANTICIPATION	entails that *X* is expected but unlike FEAR, does not entail that *X* is a source of anxiety or fear, e.g., *forecast X, predict X* (24).
SEARCH	entails that *X* is searched for but unlike DESIRE or DEMAND, does not entail that X is desired or requested, e.g., *search for X* (89).
PREVENTION	entails that *X* is prevented. Unlike CANCELATION, it does not entail that there was a plan or a demand before preventing, e.g., *preclude X* (54).
CANCELATION	entails that *X* is canceled and that there was a plan or demand before canceling, e.g., *cancel X, give up X* (34).
PROHIBIT	entails that *X* is prohibited. X's right or ability to be generated or used is taken away. e.g., *ban X, forbid X* (39).
POSTPONE	entails that *X* is postponed, e.g., *postpone X, defer X* (15).
DEFICIENCY	entails that *X* does not exist but does not entail that it is prevented, canceled, prohibited, or postponed, as in other L2-classes of Foreclosing L1-class. e.g., *lack X, X is absent* (36).
NET L0-class: **Generating L1-class (596)**	
SYMBOLIZATION	entails that *X* transits from non-existence to existence as a kind of (semiotic) representation, e.g., *write X, compose (music) X* (13).
CREATION	entails that *X* transits from non-existence to existence. Unlike SYMBOLIZATION, *X* is not limited to a semiotic representation, and unlike TRANSFORMATION, it focuses less on transformation from another entity. *generate X, cause X* (509).
TRANSFORMATION	entails that *X* transits from non-existence to existence as a result of transformation. Unlike CREATION, it focuses on the transformation from another entity, e.g., *turn into X* (74).
ENT L0-class: **Corrupting L1-class (622)**	
COLLAPSE	entails that *X* transits from existence to non-existence by dying, being eliminated, or being destroyed. Unlike CONVERSION , it focuses less on transformation, e.g., *destroy X, kill X* (588).
CONVERSION	entails that *X* transits from existence to non-existence by transforming *X* into an another entity, e.g., *turned from X, changed from X* (34).
Existence L0-class: **Enabling L1-class (3,536) / Incommode L1-class (1,355)**	
RECOGNITION	entails that *X* is recognized or sensed, e.g., *find X, feel X* (308).
SELECTION	entails that *X* is selected, e.g., *appoint X, choose X* (139).
ENCOUNTER	entails that *X* emerges as a result of transportation, e.g., *send X, X arrives* (407).
ACQUISITION	entails that *X* is obtained and possessed, e.g., *buy X, catch X* (482).
PROVISION	entails that *X* is handed to be possessed, e.g., *sell X, render X* (422).
ENHANCEMENT	entails that *X* is extended, improved, or supported, e.g., *increase X, help X* (880).
PREPARATION	entails that *X* is arranged, connected, or qualified in preparation to execute its function, e.g., *cook X, install X* (822).
BEING	entails that *X* is existing or living but does not entail that *X* is recognized, selected, encountered, acquired, enhanced, or prepared, as in other L2-classes of the Enabling L1-class, e.g., *X exists, X lives* (76).
UNRECOGNIZING	entails that *X* is not recognized or sensed but unlike FORGETTING, does not entail that *X* was previously recognized, e.g., *overlook X* (8).
FORGETTING	entails that *X* is forgotten and that *X* was once recognized, e.g., *forget X, lose memory of X* (8).
UNSELECTING	entails that *X* is not selected, e.g., *alternate X, reject X* (46).
SEPARATION	entails that *X* is left or separated as a result of transportation, e.g., *X leaves, send X away* (114).
ABANDONMENT	entails that *X* is not possessed as a result of being thrown away, e.g., *throw X away, renounce X* (58).
DEPRIVATION	entails that *X* was taken away without the permission of a possessor, e.g., *steal X, take X away* (102).
DEGRADATION	entails that *X* is reduced, deteriorated, or interrupted, e.g., *X is weakened, attack X* (908).
UNPREPARED	entails that *X* is unprepared, disconnected, or unqualified, e.g., *X is uninstalled, X is disconnected* (111).
Functioning L0-class: **Actualizing L1-class (4,460) / Disordering L1-class (698)**	
EXECUTION	entails that the function of *X* is executed but unlike WORKING, does not entail that *X* successfully satisfies its function, e.g., *ignite X* (966).
WORKING	entails that the function of *X* is carried out and that *X* successfully satisfies its function, e.g., *X functions, cleaned by X* (3,106).
INITIATION	entails that *X* is started or continued, e.g., *start X, open X* (185).
SUCCESS	entails that *X* accomplished its goal and the result of the execution of its function is evaluated positively, e.g., *accomplish X, X wins* (203).
SUSPENSION	entails that the function of *X* is suspended but unlike FINISHING, does not entail that its function is terminated, e.g., *suspend X* (133).
DYSFUNCTION	entails that the function of *X* is executed but *X* is performing poorly, e.g., *X is sluggish, bored by X* (196).
FINISHING	entails that *X* is terminated, e.g., *end X, finish X*. (110).
FAILURE	entails that *X* fails to accomplish its goal and the result of the execution of its function is evaluated negatively, e.g., *X is defeated* (259).

Table 3: PPTT classes. The number in parentheses indicates the number of templates in PPTT.

Note that the existence of an L2-link does not guarantee that the semantic properties specified by it hold for all the possible template pairs taken from the class pair it connects. The cost of hand-labelling the links with such guarantees is prohibitively high because we would have to check all of the template combinations. We empirically evaluated the validity of the links in our experiments below although this is not a *direct* evaluation since the relations we derived are different from the ones given to the links.

4 Construction of PPTT and Relation Derivation

Using the automatic acquisition method proposed by Hashimoto et al. (2012), we collected 10,825 candidates of excitatory/inhibitory templates from a 600-million-page web corpus (hereafter, $WCorpus$). Hashimoto et al.'s method constructs a network of templates based on their co-occurrence in sentences with a small number of seed templates of which excitation polarity are assigned manually, and infers the polarity of all the templates in the network by a constraint solver based on the spin model (Takamura et al., 2005). Then, we added the 20,000 most frequent templates in the corpus that could not be extracted automatically for a total of 30,825 templates.

Three human annotators (not the authors) judged the polarity of the templates, and we included the excitatory and the inhibitory templates but excluded the neutral templates in PPTT due to the reason discussed in Section 3.1. We also excluded templates whose variable *X* is the subject of a transitive verb. This is because the subject position is often occupied by living things, and since the *functions/objectives* of such subjects seem difficult to identify, it is often difficult to judge whether such templates should be classified into the **Functioning** class or another. After applying these two restrictions, the first author classified the remaining 12,023 templates in PPTT.

In this work, we derived happens-before, entailment and anomalous obstruction relations among templates from PPTT. The target data is the set of all the template pairs such that a noun exists with which both templates of the pair co-occur at least 100 times in $WCorpus$. We denote this set of the template pairs by $TP100$, and all the relation derivations pick up template pairs as relation instances from it. This is because in our preliminary experiments, we found that the relation instance candidates taken from outside of $TP100$ had much lower precision. The relation derivation itself is quite simple and consists of the following two steps.

Step 1 Select L0-links or types of L2-links that are expected to represent a target semantic relation (e.g., *Present*$^+$ links are expected to represent entailment, since they represent the relations between classes where "While C_1 is taking place, C_2 must be taking place".) and extract all the class pairs connected by the selected links (e.g., INITIATION L2-class $\overset{Present^+}{\rightarrow}$ BEING L2-class). Enumerate all the template pairs from the intersection between $TP100$ and the extracted class pairs (e.g., *X is started* $\overset{Present^+}{\rightarrow}$ *X exists*).

Step 2 If necessary, rank the relation instance candidates that are extracted in Step1 by distributional similarity scores between the templates that compose the candidates, computed with $WCorpus$.

5 Experiments

This section reports our experiments on semantic relation derivation. Derived relation instances were marked by three human annotators (not the authors) who voted to break ties. Unless stated otherwise, we asked them to mark a template pair as negative if they found any noun that can be placed in both templates' argument slots and makes the template pair a negative sample for the target relation, and positive otherwise.

5.1 Happens-Before Relation

Following Regneri et al. (2010), we assumed template$_1$ (T_1) has a *happens-before relation* with template$_2$ (T_2) iff one event expressed by T_1 normally happens before another expressed by T_2, provided that both events occur. Below are our four methods to derive happens-before relation instances, each of which uses different links. Note that we did not use distributional similarity in this experiment.

H1 uses the 55 pairs of L2-classes connected by L2-link Past$^+$, meaning that a template in a class *must occur before* another.

H2 uses the 90 pairs of L2-classes connected by L2-link Future$^+$, i.e., a template in a class often *enables* another to occur.

H3 uses the 474 pairs of L2-classes connected by one of the seven L0-links in Figure 1, i.e., the canonical temporal order links.

All is the union of **H1**-**H3** results.

We prepared two baselines; **HB-Ptn** is a pattern-based method based on Chklovski and Pantel (2004a). It extracts template pairs in $TP100$ that were connected in $WCorpus$ by one of manually collected 73 conjunctives expressing temporal order, such as *after* and *before*, and which either shared the same argument or the second template was filled by the pronouns *it, this, or that*. **Random** is a random sampling from $TP100$.

Three annotators annotated 200 random samples from each method's output. Fleiss' kappa was .56 (moderate agreement). The results of their majority vote are summarized in Table 4. The recall was estimated against the number of positive samples in $TP100$ based on the precision of **Random**. The precision of all of our four methods is reasonably high for such a difficult task, and the number of relations derived by **All** reached about 4.4 million. The recall of **All** exceeds 65%, which we believe is quite high. **HB-Ptn** suffered from low recall, probably due to the data sparseness in $WCorpus$. Table 5 shows examples of the derived happens-before relations alongside L2-classes of the templates, the L2-links between the classes and the original Japanese templates. The acquired relations included many unexpected but correct happens-before relations, like *compose (a piece of music) X⊃relax by X*.

Actually, it is difficult to fairly compare our work and previous works on temporal relation acquisition, due to differences in language, the data used, and the methodologies. Nonetheless, our result with 79.5% precision is at least five times larger than the English data released by Chambers et al. (`cs.stanford.edu/people/nc/schemas`), which contains around 870,000 "before" relation candidates and happens-before database in the VerbOcean (Chklovski and Pantel, 2004a) that covers 4,205 relations. Considering our method is completely different from theirs, we believe that our contribution is valuable.

Setting/Method	Precision (%)	# of Pairs	Recall (%)
H1	**83.5**	1,113,280	18.0
H2	70.5	1,524,557	20.8
H3	67.0	3,837,116	49.7
All	79.5	4,387,781	**67.5**
HB-Ptn	53.0	32,288	0.3
Random	18.0	28,717,454	100.0

Table 4: Happens-before derivation performance.

boil X⇒eat X

PREPARATION Class $\overset{Future^+}{\rightarrow}$ EXECUTION Class

X `wo niru` ⇒ X `wo taberu`

compose (a piece of music) X⇒relax by X

SYMBOLIZATION Class $\overset{Past^+}{\leftarrow}$ WORKING Class

X `wo sakkyoku-suru` ⇒ X `de rirakkusu-suru`

Table 5: Examples of happens-before relation.

5.2 Entailment Relation

Below are our proposed methods to derive entailment relations.

Present+.DIFF extracts the 32 class pairs that are composed of *DIFFERENT* L2-classes and are connected by the *Present*$^+$ links, meaning that a template in a class *must occur simultaneously with* another template in another class, and ranks all the possible template pairs taken from each class pair using Hashimoto et al.'s (2009) conditional probability based similarity measure for entailment recognition.

Present+.SAME extracts the 41 class pairs that are composed of the *SAME* L2-classes and are connected with the *Present*$^+$ links, and ranks all the template pairs from each class pair using Hashimoto et al.'s similarity.

Past+ extracts the 55 pairs of L2-classes that are connected with the *Past*$^+$ links, meaning that a template in a class *must occur before* another, and ranks all the template pairs from each class pair using Hashimoto et al.'s similarity.

Baseline-HAS is our baseline which is our implementation of Hashimoto et al. (2009) for entailment recognition; it ranks all the template pairs in $TP100$ by Hashimoto et al.'s score. Our methods can be seen as the restrictions of the output of the baseline method using the extracted PPTT's class pairs.

Three annotators hand-labeled 500 random samples from the top 100,000 template pairs for each method. The kappa was .59 (moderate agreement), and the results of their majority vote are presented in Figure 2. Table 6 shows examples of Proposed methods' outputs. The restriction of the class pairs in our method contributed to much higher precision than using the state-of-the-art method alone.

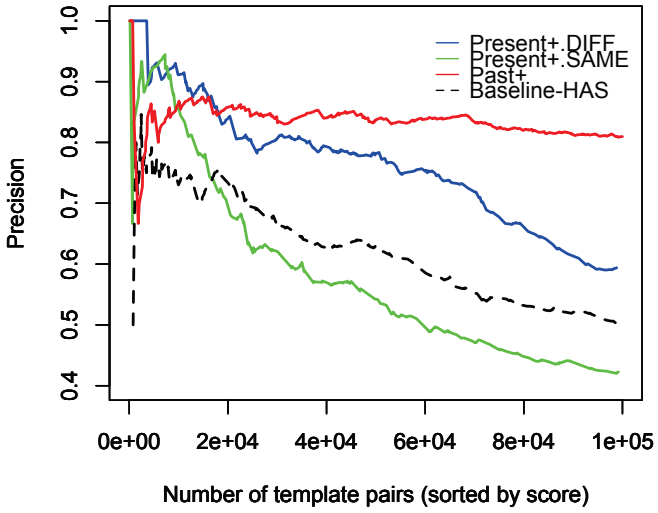


Figure 2: Entailment derivation performance.

Since the precision of **Past+** is quite high for the top 100,000 pairs, we annotated an additional 500 random samples from the top 500,000 pairs. According to this annotation, the top 408,610 pairs had 70% precision, implying that after merging all the top pairs extracted by **Present+.DIFF**, **Present+.SAME** and **Past+** whose precisions exceeded 70%, we had 0.49 million entailment pairs with 70% precision. With **Baseline-HAS**, we derived only 24,000 with the same precision. Also, the Japanese WordNet (v.1.1) covers only 2.4% of the pairs in the manually annotated positive samples from our proposed methods through the 'synsets' or any 'synlinks'. We analyzed 200 samples from the positive samples not covered by WordNet and found that 49.5% are the hyponymy type (e.g., *boil X⊃heat X*), 39.0% are the backward presupposition type (e.g., *complete X⊃start X*), and 11.5% are the synonymy type (e.g., *X passes away⊃X dies*). This seems to imply that our methods are better at deriving all types of entailment, while WordNet might be effective for only the synonymy type. In addition, by analyzing all the positive samples, we confirmed that the different types of entailment pairs were derived with different L2-links; 88.1% of the positive samples from *Present+.DIFF* and *Present+.SAME* require that two events referred to by the two templates occur with temporal overlap (e.g., *equip X⊃X exists*, i.e. *X* is equipped while *X* exists), while 96.7% of those from *Past+* were the backward presupposition type, in which an event entails another event that happened before it. This shows that the L2-links were useful for deriving various fine-grained types of entailment.

get X⊃X exists (X `wo nyuushu-suru` ⊃ X `ga sonzai-suru`) ACQUISITION Class $\overset{Present^+}{\rightarrow}$ BEING Class
evolve into X⊃change into X (X `ni shinka-suru` ⊃ X `ni kawaru`) TRANSFORMATION Class $\overset{Present^+}{\rightarrow}$ TRANSFORMATION Class
close (a shop) X⊃make X (X `wo heiten-suru` ⊃ X `wo tsukuru`) FINISHING Class $\overset{Past^+}{\rightarrow}$ CREATION Class

Table 6: Examples of entailment.

5.3 Anomalous Obstruction Relation

We assumed that template$_1$ (T_1) like *X is sold out* has an *anomalous obstruction relation* with template$_2$ (T_2) like *buy X* (denoted as *X is sold out*$\rightsquigarrow$*(cannot) buy X*) iff: (A) the event expressed by T_1 prevents the event expressed by T_2 from occurring; (B) T_1 expresses an event that should not happen if everything about the variable *X* goes as expected; and (C) T_2 expresses another event in which the function of *X* is executed, enhanced, or prepared. We derived anomalous obstructions, by generating all of the possible template pairs from the 88 L2-class pairs connected by Future$^-$ L2-links. These indicate that the events expressed by the templates in the first class of a pair *disable* the events expressed by the templates in the second class. Also, to confirm that the templates of the first class in a pair express an unexpected event,

we required the *disabler* class to have the inhibitory polarity and the *disabled* class to be excitatory. Otherwise, we would obtain such pairs as INITIATION⇝PLANNING (e.g., *start X⇝schedule X*), which indeed express the *prevention* relation (Barker and Szpakowicz, 1995), i.e., "*scheduling X* would not occur after *starting X*," which is different from anomalous obstruction.

Three annotators annotated 200 random samples for each method, and the results of their majority vote are summarized in Table 7, where **Random** refers to a random baseline using $TP100$. The recall was estimated using the number of positive samples provided by **Random**. The kappa was .60 (moderate agreement). 73.5% precision, 26.4% recall against the positive samples in $TP100$, and more than one million outputs of our proposed method are reasonably high/large results for this difficult task. Table 8 shows examples of **Proposed**'s outputs. "(cannot)" was attached to *disabled* templates for readability.

Setting/Method	Precision	# of Pairs	Recall
Proposed	**73.5**	1,081,405	26.4
Random	10.5	28,717,454	100.0

Table 7: Performance of anomalous obstruction derivation.

prohibit X⇝(cannot) exhibit X PROHIBIT Class $\overset{Future^-}{\rightarrow}$ EXECUTION Class
X `wo kinshi-suru⇝X wo kookai-suru`
break X⇝(cannot) utilize X COLLAPSE CLASS $\overset{Future^-}{\rightarrow}$ EXECUTION CLASS
X `wo kowasu⇝X wo riyo-suru`

Table 8: Examples of anomalous obstruction.

6 Conclusion

In this work, we manually constructed a Phased Predicate Template Taxonomy (PPTT), which is a network of semantically coherent classes of templates and derived semantic relations including entailment from it in a million-instance scale. Future work will extend PPTT to cover non-excitatory/non-inhibitory templates and generate richer structural knowledge similar to *full-fledged* scripts (Schank and Abelson, 1977) and narrative schemas (Chambers and Jurafsky, 2011).

Acknowledgements

We would like to thank three anonymous reviewers for many useful comments and advices on the manuscript of this paper.

References

Roni Ben Aharon, Idan Szpektor, and Ido Dagan. 2010. Generating entailment rules from framenet. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 241–246, Stroudsburg, PA, USA. ACL.

Aristotle. 1987. *De Anima (Translated by Hugh Lawson-Tancred)*. Penguin Classics, London.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 86–90, Stroudsburg, PA, USA. ACL.

Ken Barker and Stan Szpakowicz. 1995. Interactive semantic analysis of clause level relationships. In *Proceedings of PACLING '95*, Brisbane.

I.I. Bejar, R. Chaffin, and S.E. Embretson. 1991. *Cognitive and Psychometric Analysis of Analogical Problem Solving*. Springer-Verlag, New York.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2012. Learning entailment relations by global graph structure optimization. *Comput. Linguist.*, 38(1):73–111, March.

Francis Bond, Hitoshi Isahara, Sanae Fujita, Kiyotaka Uchimoto, Takayuki Kuribayashi, and Kyoko Kanzaki. 2009. Enhancing the japanese wordnet. In *Proceedings of the 7th Workshop on Asian Language Resources*, ALR7, pages 1–8, Stroudsburg, PA, USA. ACL.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio, June. ACL.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 602–610, Stroudsburg, PA, USA. ACL.

Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA, June. ACL.

Timothy Chklovski and Patrick Pantel. 2004a. Path analysis for refining verb relations. In *In Proceedings of KDD Workshop on Link Analysis and Group Detection (LinkKDD-04)*, Seattle, WA.

Timothy Chklovski and Patrick Pantel. 2004b. Verbocean: Mining the web for fine-grained semantic verb relations. In Dekang Lin and Dekai Wu, editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, EMNLP '04, pages 33–40, Barcelona, Spain, July. ACL.

Robert Coyne and Owen Rambow. 2009. Lexpar: A freely available english paraphrase lexicon automatically extracted from framenet. In *Proceedings of the Third IEEE International Conference on Seman- tic Computing*.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2009. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 15(4):i–xvii.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Charles J. Fillmore. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech*, 280(1):20–32.

Michael A.K. Halliday. 1985. *An Introduction to Functional Grammar*. Arnold, 1st edition.

Chikara Hashimoto, Kentaro Torisawa, Kow Kuroda, Stijn De Saeger, Masaki Murata, and Jun'ichi Kazama. 2009. Large-scale verb entailment acquisition from the Web. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1172–1181, Singapore, August. ACL.

Chikara Hashimoto, Kentaro Torisawa, Stijn De Saeger, Jong-Hoon Oh, and Jun'ichi Kazama. 2012. Excitatory or inhibitory: a new semantic orientation extracts contradiction and causality from the web. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 619–630, Stroudsburg, PA, USA. ACL.

Chikara Hashimoto, Kentaro Torisawa, Julien Kloetzer, Motoki Sano, István Varga, Jong-Hoon Oh, and Yutaka Kidawara˙ 2014. Toward future scenario generation: Extracting event causality exploiting semantic relation, context, and association features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, USA, June. Association for Computational Linguistics.

David A. Jurgens, Peter D. Turney, Saif M. Mohammad, and Keith J. Holyoak. 2012. Semeval-2012 task 2: Measuring degrees of relational similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, pages 356–364, Stroudsburg, PA, USA. Association for Computational Linguistics.

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2006. Extending verbnet with novel verb classes. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, pages 731–738, Genoa, Italy, June.

Julien Kloetzer, Stijn De Saeger, Kentaro Torisawa, Chikara Hashimoto, Jong-Hoon Oh, Motoki Sano, and Kiyonori Ohtake. 2013. Two-stage method for large-scale acquisition of contradiction pattern pairs using entailment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 693–703, Seattle, Washington, USA, October. ACL.

Beth Levin. 1993. *English Verb Classes and Alternations*. The University of Chicago Press, Chicago and London.

Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger, and Kiyonori Ohtake. 2013. Why-question answering using intra- and inter-sentential causal relations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1733–1743, Sofia, Bulgaria, August. ACL.

Georgiana Puscasu and Verginica Barbu Mititelu. 2008. Annotation of wordnet verbs with timeml event classes. In Bente Maegaard Joseph Mariani Jan Odijk Stelios Piperidis Daniel Tapias Nicoletta Calzolari (Conference Chair), Khalid Choukri, editor, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

James Pustejovsky, Jos Castao, Robert Ingria, Roser Saur, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003. Timeml: Robust specification of event and temporal expressions in text. In *in Fifth International Workshop on Computational Semantics (IWCS-5*.

James Pustejovsky. 1998. *The Generative Lexicon*. MIT Press, Cambridge.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 979–988, Uppsala, Sweden, July. ACL.

Roger C. Schank and Robert P. Abelson. 1977. *Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures*. L. Erlbaum, Hillsdale, NJ.

Idan Szpektor, Eyal Shnarch, and Ido Dagan. 2007. Instance-based evaluation of entailment rule acquisition. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 456–463, Prague, Czech Republic, June. ACL.

Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting semantic orientations of words using spin model. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 133–140, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Partha Pratim Talukdar, Derry Wijaya, and Tom Mitchell. 2012. Acquiring temporal constraints between relations. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 992–1001, New York, NY, USA. ACM.

P. D. Turney and S. M. Mohammad. 2014. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering*, FirstView:1–40, 5.

István Varga, Motoki Sano, Kentaro Torisawa, Chikara Hashimoto, Kiyonori Ohtake, Takao Kawai, Jong-Hoon Oh, and Stijn De Saeger. 2013. Aid is out there: Looking for help from tweets during a large scale disaster. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1619–1629, Sofia, Bulgaria, August. ACL.

Hila Weisman, Jonathan Berant, Idan Szpektor, and Ido Dagan. 2012. Learning verb inference rules from linguistically-motivated evidence. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 194–204, Jeju Island, Korea, July. ACL.

Combining Supervised and Unsupervised Parsing for Distributional Similarity

Martin Riedl, Irina Alles and **Chris Biemann**
FG Language Technology
Computer Science Department, Technische Universität Darmstadt
Hochschulstrasse 10, D-64289 Darmstadt, Germany
{riedl,biem}@cs.tu-darmstadt.de, ialles@gmx.de

Abstract

In this paper, we address the role of syntactic parsing for distributional similarity. On the one hand, we are exploring distributional similarities as an extrinsic test bed for unsupervised parsers. On the other hand, we explore whether single unsupervised parsers, or their combination, can contribute to better distributional similarities, or even replace supervised parsing as a preprocessing step for word similarity. We evaluate distributional thesauri against manually created taxonomies both for English and German for five unsupervised parsers. While for English, a supervised parser is the best single parser in this evaluation, we find an unsupervised parser to work best for German. For both languages, we show significant improvements in word similarity when combining features from supervised and unsupervised parsers. To our knowledge, this is the first work where unsupervised parsers are systematically evaluated extrinsically in a semantic task, and the first work to show that unsupervised parsing can complement and even replace supervised parsing, when used as a pre-processing feature.

1 Introduction

While the field has seen increased interest in automatically inducing syntactic structures from raw or part-of-speech (POS) tagged text, the evaluation of unsupervised data-driven parsers has almost exclusively been conducted either by introspection or by automatic comparison to treebanks. It might be due to comparatively low scores on reproducing a treebank's syntactic annotation that hardly anyone has yet attempted to use the output of unsupervised parsers for an NLP task other than parsing itself.

A further complication with unsupervised parsers – be it dependency parsers, constituency parsers or combinatory categorial grammar parsers – is that the categories induced by such parsers cannot be straightforwardly mapped to linguistically-inspired categories as defined in a treebank. But also when considering only unlabeled syntactic annotations, an unsupervised parser is hardly to blame if it does not adhere to sometimes arbitrary conventions: e.g. for dependencies, it is not a priori clear how to connect auxiliary and main verbs, where to attach the complementizer of subordinate clauses, how to represent a conjunction and its conjuncts, how to relate the preposition and the nominal in prepositional phrases, and how to handle punctuation, cf. Nivre and Kübler (2006), Schwartz et al. (2011).

When it comes to *utilizing* syntactic structures, however, it is more important that they are consistent across different sentences than that they adhere to specific syntactic theories and conventions. Here, we choose a task that makes only intermediary use of syntactic structures: we employ unsupervised parsing for preprocessing corpora for the purpose of computing distributional similarities. Since it is generally accepted (e.g. (Lin, 1997; Curran and Moens, 2002)), that syntactic preprocessing plays an important role for the quality of distributional thesauri, and comparing words along their syntactic contexts does rely on the existence of such a structure rather than its actual representation, we believe that distributional similarities are an excellent test bed for addressing the following two research questions: (1) How do unsupervised parsers compare to supervised parsers when used as feature providers for building Dis-

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1435–1446, Dublin, Ireland, August 23-29 2014.

tributional Thesauri (DTs) in comparison to supervised parsers? (2) Can the combination of syntactic parsers increase DT quality?

2 Related Work

2.1 Unsupervised Parser Evaluation

As with other unsupervised approaches, the premise of unsupervised induction of syntactic structure is to alleviate the bottleneck of expensive manual annotations for improving NLP applications. For grammar induction, the potential is extremely high due to the complexity of the subject matter: treebanks belong to the most work-intensive NLP datasets. On the other hand, this complexity is hard to grasp for unsupervised systems, which is probably the reason why unsupervised parsing technology is still in its infancy, despite more than a decade of work on this topic.

One of the early works inducing structure from raw sentences and yielding better performance than a random baseline was achieved by van Zaanen and of Leeds. School of Computer Studies (2001), who used an Alignment Based Learning (ABL) approach. This algorithm compares all sentences of a given set and considers matching sequences as constituents. Klein and Manning (2002) presented another approach focusing on constituent sequences called the Constituent-Context Model (CCM). It is an EM-based iterative approach that makes use of the linguistic phenomenon that long constituents often have shorter representations of the same grammatical function that occur in similar contexts. A hybrid approach combining CCM with a dependency model, called Dependency Model with Valence (DMV), shows even better performance and is the first unsupervised system to outperform the right-branching baseline (Klein and Manning, 2004). A great number of recent works are based on DMV, such as the system by Headden III et al. (2009), who improved DMV by adding lexical information, and Gillenwater et al. (2010) who added posterior regularization during the training process. Bod (2007) takes a slightly different direction by following an "all subtrees approach", where all possible binary trees are generated for each sentence. It generates all possible binary trees for each sentence. The parse of a new sentence is determined by selecting the most probable tree based on the previously accumulated subtree frequencies. Most of the evaluation of these parsers was performed against a treebank, offering manually annotated and linguistically motivated parse trees. Schwartz et al. (2011) underline the fact that treebanks contain linguistically problematic annotations, cases without linguistic consensus, such as the decision on the head of a verb phrase or a sequence of nouns. They show that the neglectance of these cases has a significant but unjustified negative influence on the evaluation outcomes and propose a new measure, Neutral Edge Direction (NED), which alleviates this problem. Bod (2007) argues that parser evaluation against a treebank favors supervised approaches and therefore measures the parser quality on the outcome of a syntax based Machine Translation (MT) task where the dependency parsers are evaluated as language models. In Motazedi et al. (2012), a single unsupervised parser is evaluated in an extrinsic evaluation for realisation ranking, and does not compare favorably against a supervised parser. Other extrinsic evaluations with supervised dependency parsers have been performed in information extraction systems (Miyao et al., 2008; Buyko and Hahn, 2010) or semantic role labeling (Johansson and Nugues, 2008).

2.2 Evaluating Distributional Similarity

Distributional thesauri have been evaluated both extrinsically and intrinsically. Extrinsic evaluations have been performed e.g. for automatic set expansion (Pantel et al., 2009) or phrase polarity identification (Goyal and Daumé, 2011). In this work, we will conduct an intrinsic evaluation, which is more common for the evaluation of DTs and lexical semantic similarity. Lin (1997; 1998) introduced two measures using WordNet (Miller, 1995) and Roget's Thesaurus. Using WordNet, he defines context (synsets a word occurs in Wordnet or subsets when using Roget's Thesaurus) and then builds a gold standard thesaurus using a similarity measure on these contexts. Then he evaluates his automatically computed Distributional Thesaurus (DT) with respect to the gold standard thesauri. Weeds et al. (2004) evaluate various similarity measures based on 1000 frequent and 1000 infrequent target terms. Curran (2004) created a gold standard thesaurus by manually extracting entries from several English thesauri for 70 words. His automatically generated DTs are evaluated against this gold standard thesaurus. All these

systems employ context representations based on syntactic parsing for computing word similarity.

We are going to use a comparatively simple WordNet-based measure, which calculates the similarity between two terms using the WordNet::Similarity path measure (Pedersen et al., 2004), and averages path scores between a target term and its n most similar terms. The score between two terms is inversely proportional to the shortest path between all the synsets of both terms. If two terms share a synset, the highest possible score of one is assigned. The score is 0.5 for terms that stand in a direct hypernym relation, and so on. While the absolute scores are hard to interpret due to inhomogeneity in the granularity of WordNet, they are well-suited for relative comparison when operating on the same set of target terms. The evaluation in this work is performed by comparing the average score of the top ten entries in the DT for each of the target terms and report separately on frequent and rare words. Riedl and Biemann (2013) also show that the results, using the WordNet based approach, are highly correlated to the results observed with Curran's approach using a manually created thesaurus. This justifies the usage of manually created taxonomies for this evaluation.

3 Methodology

3.1 Parsers

In our evaluation, we use five unsupervised parsers, which we will describe briefly. They have been selected to span several paradigms of unsupervised syntax induction, and due to software availability.

Gillenwater et al. (2010)[1] use a model based on the DMV (Klein and Manning, 2004) and improve performance by adding sparsity biases on dependency types. They assume a corpus annotated with POS tags. The aim of this bias is to limit unique head-dependent tag pairs, which is achieved by a constraint on model posteriors during the learning process.

The work of Marecek and Straka (2013)[2] is another enhancement of the DMV and is subsequently referred to as Unsupervised Dependency Parser (UDP). It additionally uses prior knowledge in the form of stop estimates that are computed on a large raw corpus using the reducibility principle: a sequence of words is considered as reducible if a word can be removed from the phrase and the remaining part appears another time in the corpus. The assumed property, that the first word of a reducible sequence does not have any left children and the last word of this sequence does not have any right children, is used for the calculation of such stop estimates. The authors show that estimates computed on a large corpus such as Wikipedia can be used for the parsing of new text.

Bisk and Hockenmaier (2013) use an EM approach to induce a Combinatory Categorial Grammar (CCG), based on very general linguistic assumptions. It creates a model that can be used to parse unseen data. The algorithm requires a corpus, previously assigned with POS tags, in order to be able to distinguish between word classes (mainly to find the verb), and employs general knowledge such as that sentences are headed by verbs. Further language-specific properties are induced from the training data.

Seginer (2007)[3] takes an incremental parsing and learning approach. It operates directly on the plain text without the need for POS tags, by using Common Cover Links (CCL), which can be directly converted to dependency arcs. This parser learns during parsing and can be used without a prior learning step. This should result in increased parsing quality towards later stages, which suggests several passes over the training data. The obtained model can then be reused to parse unseen data.

The approach of Søgaard (2012) is different from all other approaches discussed here: This algorithm does not require any training data and can operate with or without POS tags. For this reason, it can be applied to arbitrary amounts of data, since it operates sentence-wise without memorizing previous inputs, and produces non-projective dependency parses. The words of a phrase are ordered by centrality and a parse is determined by the ranking of a parsing algorithm, which uses general linguistic knowledge for grammar induction. This knowledge is inspired by the rules of Naseem et al. (2010), and the approach has been tuned (once and for all, for all languages) on development data from the Penn Treebank.

[1] http://code.google.com/p/pr-toolkit/
[2] http://ufal.mff.cuni.cz/udp/
[3] http://www.seggu.net/ccl/

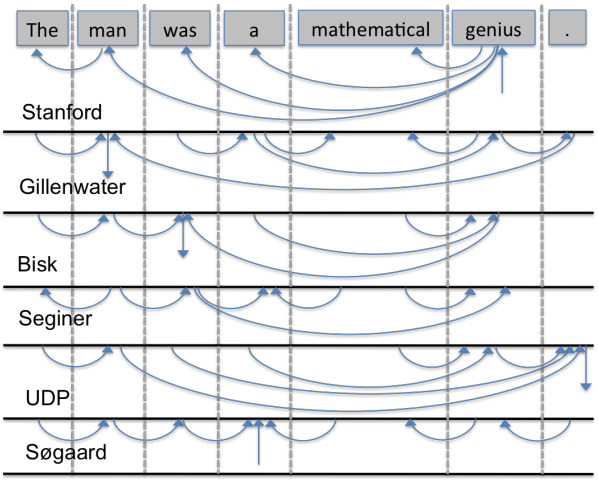


Figure 1: Parses for an example sentence for several parsers. Here, Bisk's parser looks most similar to the parses extracted from the Stanford parser. Gillenwater and UDP seem to have some problems with the full stop. Søgaards parser mostly connects neighbors.

	Baseline	Søgaard	Gillenwater	UDP	Bisk	Seginer	Seginer
English	53.2	59.9	64.4	55.4	70.3	55.6 (WSJ 40)	74.2 (WSJ 10)
German	33.7	57.6	35.7	52.4	68.4	38.2 (Negra 40)	48.0 (Negra 10)

Table 1: Unlabeled accuracy values of different unsupervised parsers based on the CoNLL-X shared task (Buchholz and Marsi, 2006). Seginers results show F-measure values for the Negra and the WSJ corpus, used with maximum sentence of lengths of 10 and 40.

An example sentence and the according parses coming from the 10M model, except for UDP, where the 1M model is used (cf. Table 2 in Section 4.3.1), are shown in Figure 1.

Table 1 reports the accuracy of four parsers for the English and the German treebanks from the CoNLL-X shared task (Buchholz and Marsi, 2006) predicting unlabeled dependency parses for sentences with length equal and smaller than 10 tokens. Seginer reports only F-scores for WSJ and Negra considering sentences with a maximum length of 10 and 40. The best baselines reported in Canisius et al. (2006) are a left branching method for English and a nearest neighbor branching method for German, which is a combination of left and right branching.

3.2 Computing Distributional Thesauri

The extraction of context features, used to calculate similarities between terms, is performed in accordance with the generic scheme proposed in (Biemann and Riedl, 2013): A (typed or untyped) parser arc is split into term and context feature, which consists of the edge direction and label (if any), and the connected term. Similarity between terms is subsequently computed on the overlap of their most salient context features. We represent the term t and the context feature c as a pair $< t, c >$ and extract a dependency triple (or dependency pair, as most unsupervised dependency parsers do not label the edges). For the dependency between *I* and *gave* `(nsub;gave;I)` in *I gave her the book*, terms and context features would look like `<gave,(nsub,I,@)>` and `<I,(nsub,@,gave)>`. In this example, the term *gave* is characterized by the context information that *I* is its nominal subject, and term *I* is characterized by being the subject of *gave*. We build distributional thesauri using the JoBimText[4] open-source framework. This framework scales to large data and has proven to outperform other methods, when using large data (Riedl and Biemann, 2013). The computation of the distributional thesaurus within this framework is following the MapReduce paradigm and scales to very large corpora. This is achieved by applying a significance measure between term and context feature, retaining only the most salient 1000 context features per term, and computing the cardinality of the set overlap between the respective context features

[4] `www.jobimtext.org`, (Biemann and Riedl, 2013)

per term, which defines the similarity between terms. Per term, the most similar terms are subsequently ranked, resulting in a distributional thesaurus as introduced by Lin (1997).

4 Evaluation

We report experimental results on German and English corpora. Both corpora are compiled from 10 million sentences (about 2 Gigawords) each from the Leipzig Corpora Collection[5], randomly sampled from online newspapers. The semantic similarity in English DTs is assessed using WordNet 3.1 as a lexical resource, as proposed by Riedl and Biemann (2013). For evaluating the German DTs, we replace WordNet by its German counterpart, GermaNet 8 (Hamp and Feldweg, 1997). We report results separately for frequent and infrequent targets and average the path scores for the most similar 10 words per entry. The evaluation of the English DTs is performed using 1000 frequent and 1000 infrequent nouns, as previously employed by Weeds et al. (2004). These nouns are randomly sampled from the British National Corpus (BNC) and all occur in WordNet. For the evaluation of German DTs, we randomly selected 1000 frequent and 1000 infrequent nouns from our German corpus that all occur in GermaNet.

4.1 Experimental Settings

The DTs are calculated using the dependencies from the unsupervised parsers, one at a time. To show the impact of corpus size, we down-sampled our corpora, and used 1 million (1M), 100,000 (100K) and 10,000 (10K) sentences (raw or automatically POS-tagged with the TreeTagger[6]) for training/inducing the parsers. Not all parsers were able to deal with the large training sets in feasible runtime, which might either be due to their computational complexity or their implementation. While it would be preferable to keep the corpus size for DT computations constant, this was not possible since some of our unsupervised parsers cannot be applied to unseen text. Hence, we decided to report DT quality results for two setups: Setup A uses the same data for training the parsers and the DT computation. Setup B uses the full corpus of 10M sentences for DT computation, parsed with unsupervised parsers induced on differently sized corpus samples. We feel that Setup B is better reflecting the possible utilization of unsupervised parsers for semantic similarity, since DT quality is known to increase with corpus size. However, we still wanted to assess the quality of parsers that cannot be operated on unseen text in their current state of development.

4.2 Four Baselines

We compare the results of unsupervised parsers against four baselines. As a lower-bound baseline, we use a random dependency parser that connects each word in a sentence with a randomly chosen other word. As a supervised upper-bound baseline, we use Stanford collapsed dependencies (Marneffe et al., 2006) for the English data and dependencies coming from the Mate tools (Bohnet, 2010) for the German corpus. Finally, to gauge whether the potential of unsupervised parsing to model long-range dependencies – as opposed to local n-gram contexts – lead to better distributional similarities, we use word bigrams and trigrams as n-gram-based systems. The bigram system simulates left- and right-branching. We characterize the word in the first and in the second position of two neighboring words, which results to the following term feature pairs according to the example in Section 3.2: `<I,(@,gave)>` and `<gave,(I,@)>`. Using the trigram, we characterize the word in the second position with the context feature formed by the pair of words in first and third position. The term-feature pair for *gave* would be `<gave, (I,@,her)>`.

While we expect the scores of any reasonable unsupervised parser to fall somewhere between the lower bound and the upper bound when compared in the same setting, the n-gram baselines serve as a measure for whether it is worth the trouble to induce and run the unsupervised parser for our evaluation application, as opposed to relying on an arguably simpler system for this purpose.

[5] `corpora.uni-leipzig.de`, (Richter et al., 2006)

[6] `www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`, (Schmid, 1997)

4.3 Results

4.3.1 Single Parser Results for English

We summarize the results for the English evaluation for Setup A and Setup B in Table 2. All unsupervised parsers beat the random baseline in all setups, with higher improvements observed using more training data, which somewhat validates their approaches. Also, more data for DT computation results in higher similarity scores, and rare words generally receive lower scores on average, which is expected and validates the DT computation framework.

		10k		100k		1M		10M	
	Parser	freq	rare	freq	rare	freq	rare	freq	rare
Setup A	Random	0.115	0.029	0.128	0.082	0.145	0.103	0.159	0.113
	Trigram	0.133	0.020	0.179	0.082	0.200	0.120	0.236	0.151
	Bigram	0.140	0.029	0.173	0.088	0.208	0.129	0.246	0.159
	Stanford	0.151	0.028	0.209	0.128	0.261	0.176	0.280	0.209
	Seginer	0.136	0.027	0.176	0.085	**0.211**	0.127	0.240	0.155
	Gillenwater	0.135	0.026	0.159	0.077	0.195	0.117	0.223	0.141
	Søgaard	0.120	0.027	0.147	0.083	0.185	0.117	0.227	0.144
	UDP	0.127	0.017	0.169	0.063	0.204	0.119	*	*
	Bisk	0.118	0.017	*	*	*	*	*	*
Setup B	Seginer	0.200	0.063	0.236	0.139	0.241	0.156	0.240	0.155
	Gillenwater	0.220	0.140	0.221	0.142	0.221	0.141	0.223	0.141
	Søgaard	0.227	0.144	0.227	0.144	0.227	0.144	0.227	0.144
	Bisk	0.220	0.139	*	*	*	*	*	*

Table 2: Setup A English: Parser induction and DT computation on the same corpus. Wordnet path scores averaged on top 10 similar words, for 1000 frequent and 1000 rare nouns. A * denotes that the evaluation failed because of computational constraints. Setup B English: Parser induction on different corpus sizes, and DT computation on 10M sentences.

In comparison to the n-gram baselines, only the parser by Seginer yields a higher score for frequent words and 1M sentences training in Setup A. However, the difference is very small and is confirmed on the 10M sentences only in comparison to the Trigram baseline. It seems that Seginer's training procedure saturates somewhere between 100K and 1M sentences, and shows even slightly worse performance on 10M sentences of training in Setup B. All parsers do not seem to be particularly useful as preprocessing steps for DT computation, since better similarity can consistently be reached by context features based on bigram statistics.

Comparing the unsupervised parsers, we note that Seginer's approach consistently scores highest in Setup A, while UDP comes in second for frequent words but not for rare words. While Gillenwater's approach reaches comparably high scores for small corpora in Setup A, it is beaten by Søgaard's no-training approach for larger corpora: It seems that Gillenwater's training procedure can hardly make use of additional training, which is shown in Setup B, where practically no differences are observed between 10K and 10M sentences of parser training. Differences in Setup A are thus solely due to increased corpus size for DT computation for the Gillenwater experiments.

UDP did not finish parsing 10M sentences after running for 157 days, and it is not trivial to disable its update procedure, which is why we could not include UDP in Setup B. Bisk's parser is a special case in this evaluation, since it only selects sentences shorter than 15 tokens for training, and hence was effectively trained on a 5400 sentence subset of the 10K corpus. While we did not manage to train it on larger corpora, we could apply this model on 10M sentences in Setup B, where it lands slightly below the no-training Søgaard parser, but clearly above Seginer's approach for 10K training.

4.3.2 Single Parser Results for German

A different picture is drawn for the German evaluation (see Setup A in Table 3). Comparing the results of the unsupervised parsers, Seginer's parser does not only outperform the trigram and bigram baseline for frequent nouns but also the supervised Mate parser for all corpus sizes. Yet, the improvements over

the Mate parser are not significant for all results using a paired t-test[7]. Also, Søgaards parser exceeds the trigram and bigram baseline for 10 million sentences. The remaining unsupervised parsers can beat the random baseline for frequent nouns but none of the n-gram baselines. Again we are not able to parse the 10 million sentences using UDP and also Gillenwater's parser failed, parsing this corpus. Comparing the baselines in Setup A (see Table 3), we observe a difference between the sophisticated baselines and the random baseline only for frequent words.

		10k		100k		1M		10M	
	Parser	freq	rare	freq	rare	freq	rare	freq	rare
Setup A	Random	0.097	0.002	0.108	0.010	0.123	0.051	0.143	0.077
	Trigram	0.102	0.002	0.130	0.014	0.159	0.067	0.179	0.086
	Bigram	0.112	0.003	0.130	0.009	0.163	0.053	0.192	0.082
	Mate	0.111	0.004	0.126	0.014	0.170	0.027	0.204	0.090
	Seginer	**0.113**†	0.002	**0.137**†	0.012	**0.171**	0.068	**0.208**	0.091
	Gillenwater	0.104	0.002	0.118	0.009	0.132	0.040	*	*
	Søgaard	0.104	0.002	0.123	0.010	0.161	0.054	0.193	0.077
	UDP	0.107	0.001	0.129	0.004	0.151	0.021	*	*
	Bisk	0.101	0.002	*	*	*	*	*	*
Setup B	Seginer	0.153	0.004	0.186	0.021	0.200	0.092	0.208	0.091
	Gillenwater	0.189	0.080	0.190	0.082	0.189	0.080	*	*
	Søgaard	0.193	0.077	0.193	0.077	0.193	0.077	0.193	0.077
	Bisk	0.185	0.069	*	*	*	*	*	*

Table 3: Setup A and B for German corpora.

Furthermore, we see that the supervised Mate parser results in worse scores for the frequent nouns using the 10k and 100k dataset in comparison to the bigram baseline. This could be attributed to the heavier tail in German's word frequency distribution, which results in sparser context features for small data[8]. For the 1M and 10M datasets, the supervised parser yields the best similarities for frequent nouns.

The results for Setup B for the German corpora are shown at the bottom in Table 3. We observe similar trends to the ones for the English data: using more data for the training does not seem to help the performance of Gillenwater's algorithm. Noticeable is the increase of Seginer's results for rare words as training data size increases. Seginer's algorithm even beats both n-gram baselines for the 10M corpus when trained only on 1 million sentences.

4.3.3 Combining Parsers for DT Quality Improvement

To clarify the best practice for building a DT of high quality, we combine different parsers: the two best-performing unsupervised parsers (Søgaard's and Seginer's), the baselines and the supervised parser. Additionally, these two parsers where the only ones which could be applied to the largest dataset for both languages.

For English (see Table 4), we observe a boost in performance when combining unsupervised parsers. Combining the supervised Stanford parser with the bigram and the trigram baselines also leads to a significant improvement ($p < 0.01$)[9] over the Stanford parser alone, which is about the same as combining the supervised parser with the two unsupervised parsers, and combining all five types of features for DT construction. Overall, a relative improvement of 3.5% on the average WordNet::Path measure for frequent words and a relative 4% improvement for rare words is obtained over the Stanford parser alone.

The results for German (see Table 5) show a similar trend. It is remarkable that merging the two unsupervised parsers already outperforms the supervised Mate parser significantly[9] with $p < 0.01$ (6.7% for frequent and 8% relative improvement for rare words). The combination of the supervised and unsupervised parsers again leads to further improvement, which is also significant over the supervised parser alone, and again, adding the bigram and trigram baselines to the three parsers does not help.

[7]Significant improvements ($p < 0.01$) against the Mate parser are marked with the symbol † in Table 3 for frequent nouns.

[8]Within the 10M sentences, there are 22 million word types in the German corpus and 10 million word types in the English corpus.

[9]We use a paired t-test to compare the DTs built using the supervised parser and the combinations.

Parser	frequent	rare
Stanford (supervised)	0.280	0.209
Seginer	0.240	0.155
Søgaard	0.227	0.144
Seginer & Søgaard	0.248	0.162
Stanford & Bigram & Trigram	**0.290**†	**0.217**†
Stanford & Seginer & Søgaard	**0.291**†	**0.217**†
Stanford & Seginer & Søgaard & Bigram & Trigram	**0.290**†	**0.218**†

Table 4: Combinations of different parsers for computing English thesauri. The cross (†) indicates significant improvements over the supervised parser.

Parser	frequent	rare
Mate (supervised)	0.204	0.090
Seginer	0.208	0.091
Søgaard	0.193	0.077
Seginer & Søgaard	0.218†	0.097†
Mate & Bigram & Trigram	0.204	0.091
Mate & Seginer & Søgaard	**0.222**†	**0.100**†
Mate & Seginer & Søgaard & Bigram & Trigram	**0.222**†	**0.100**†

Table 5: Combinations of different parsers for computing German thesauri

4.3.4 Discussion

Overall, it is surprising how well Søgaard's parser performs in comparison to others on this task, since it neither uses training nor relies on POS tags. This hints at either unsupervised parsing being simpler than commonly assumed or rather the immaturity of all unsupervised parsers tested. Further, we would have expected that trained unsupervised parsers, as most unsupervised methods, would exhibit a better performance when trained on larger corpora. This could not be confirmed for both systems that we trained on various corpus sizes, i.e. Seginer's and Gillenwater's approach. The findings are only moderately correlated with evaluations on treebanks, cf Table 1: Whereas Seginer's and Søgaard's parsers perform favorably in our evaluation, they are outperformed by Bisk's parser on treebanks, which currently does not scale to large data. Gillenwater's parser seems to be overly tuned to English treebanks, but cannot capitalize on this in our evaluation for English.

POS information does not seem beneficial for unsupervised parser induction in noun similarity evaluation, since the highest-scoring approach by Seginer does not use POS tags and a version of Søgaard's parser with POS tags scored slightly but consistently lower than the version without POS, as we found in further experiments. This is in line with the findings of Cramer (2007), who reports no benefit from manually corrected or unsupervised POS tags for a range of unsupervised parsers.

Comparing the results of previous intrinsic evaluations (see Table 1) and the results of our extrinsic evaluation (see Table 2 and 3), we observe that the ranking of parsers is only mildly correlated. Thus, our proposed evaluation covers different aspects than the adherence to (partially arbitrary) conventions of manually labeled dependency data. Also, our current evaluation disregards all arcs that do not involve nouns.

When combining parsers, we observe that we can improve the quality of DTs significantly. This leads us to conclude that unsupervised parsers should at least be used for generating features when computing distributional thesauri of high quality. In case no high-quality supervised parser is available for the language or domain of interest, it might suffice to use combinations of unsupervised parsers.

We also report the computation times of the different parsers, for the English dataset for Setup A (see Table 6). The results were computed on a server with 80 GB and 16 cores. Whereas all parsers require different amounts of memory, all parsers are single-threaded[10]. While Søgaard's parser is the fastest for small datasets, Seginer's runs faster on 10 million sentences. Whereas Gillenwater's and Seginer's

	10k	100k	1M	10M
Seginer	210	224	261	**508**
Gillenwater	3248	3248	3280	5546
Søgaard	**3**	**21**	**182**	975
UDP	183	1220	11316	-

Table 6: Computation time in minutes for parsing the data according to the English corpora used in Setup A, cf. Table 2

[10] As Søgaards algorithm parses sentence-wise without storing any information, it could be easily run multi-threaded.

algorithm require almost the same time for parsing 10k, 100k or 1M sentences, the runtime of the UDP and Søgaard's parser is linear in time with the number of sentences to be parsed. We cannot report the parsing times for the Bisk algorithm, as the parsing was not performed by us. Again it is noticeable that the best two parsers are also the two unsupervised parsers that run quickest.

5 Conclusion

The contribution of this paper is two-fold: First, we have proposed and conducted a comparative extrinsic evaluation procedure for unsupervised parsers based on noun similarity in DTs. Second, we have explored how to improve DT quality by combining features from several parsers. The transparency of this method with respect to the kind of induced structures (dependencies, constituent trees, combinatory categorial grammar) and with respect to labels of nodes and edges in the parse graph makes it possible to compare different unsupervised parsers without having to rely on treebanks. Since semantic similarity, especially for nouns, is a building block for many NLP applications, we feel that removing the dependency on high-quality supervised parsers can give rise to semantic technologies in many languages. We have conducted this evaluation with five different unsupervised parsers, and examined the influence of corpus size for parser training and for the similarity computation in a series of experiments. Using established methods for evaluating distributional similarity against lexical semantic resources, we were able to measure differences between parsers in this task that are not reflected by intrinsic evaluations that compare their induced structures to treebanks. These include the influence of corpus size on the training procedure and the consistency of parse fragments on "frequent versus rare words" as well as different languages. Further, we were able to pinpoint a crucial point in unsupervised parsers that has not received much attention: approaches that do not induce an actual parser that can be run on unseen sentences but merely produce syntactic annotations for a given fixed training corpus are hardly useful in applications.

Our evaluation results can be summarized as follows: For English, with its relatively fixed order, Seginer's parser achieves very scarce to no improvements compared to a simple n-gram baseline when used to compute distributional similarities. But for German, Seginer's parser outperforms all baselines including a state-of-the-art supervised parser, and Søgaard's simplistic approach compares favorably to the n-gram baselines. Furthermore, we demonstrate that the quality of noun similarity can be improved significantly when combining the features from supervised and unsupervised parsers.

While today's unsupervised parsers might not be ready for their utilization for semantic similarity for the English language, they can be applied to a large number of other languages where highly optimized supervised parsers are not available. Additionally, we feel that our proposed evaluation method exhibits enough sensitivity to be a meaningful test bed for future unsupervised parsers.

6 Outlook

Where do we go from here? We strongly argue that in times of availability of very large monolingual corpora from the web, we should strive at unsupervised parser induction systems that can make use of large training data, as opposed to focussing our efforts on complex models that scale poorly, and thus cannot elevate to the performance levels needed in order to make unsupervised parsing a building block in natural language processing applications.

For further work, we want to proceed in several ways: we would like to extend our evaluation framework from nouns to other parts of speech. Furthermore, we will explore whether unsupervised parsers can be tuned towards the task of computing a distributional thesaurus, e.g. by using only assignments with a certain confidence, type, or from sentences with limited length. Additionally, we would like to explore the interaction of unsupervised POS induction and grammar induction (Headden, III et al., 2008), in order to entirely remove language-dependent preprocessing for the purpose of semantic similarity computations, while at the same time being able to leverage the advantages of structured representations, cf. Erk and Padó (2008). Finally, we would like to test whether we can also detect a different ranking for different supervised parsers when comparing their scores in the normal treebank setting versus using them for building distributional thesauri.

Acknowledgments

This work has been supported by the German Federal Ministry of Education and Research (BMBF) within the context of the Software Campus project *LiCoRes* under grant No. 01IS12054, by IBM under a Shared University Research Grant and by DFG under the *SemSch* project grant.

References

Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity. *Journal of Language Modelling*, 1(1):55–95.

Yonatan Bisk and Julia Hockenmaier. 2013. An HDP Model for Inducing Combinatory Categorial Grammars. In *Transactions of the Association for Computational Linguistics*, pages 75–88, Atlanta, GA, USA.

Rens Bod. 2007. Is the end of supervised parsing in sight? In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 400–407, Prague, Czech Republic.

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 89–97, Beijing, China.

Sabine Buchholz and Erwin Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 149–164, New York City, New York.

Ekaterina Buyko and Udo Hahn. 2010. Evaluating the impact of alternative dependency graph encodings on solving event extraction tasks. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 982–992, Cambridge, Massachusetts.

Sander Canisius, Toine Bogers, Antal van den Bosch, Jeroen Geertzen, and Erik Tjong Kim Sang. 2006. Dependency parsing by inference over high-recall dependency predictions. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 176–180, New York City, New York.

Bart Cramer. 2007. Limitations of Current Grammar Induction Algorithms. In *Proceedings of the ACL 2007 Student Research Workshop*, pages 43–48, Prague, Czech Republic.

James R. Curran and Marc Moens. 2002. Improvements in automatic thesaurus extraction. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition - Volume 9*, ULA '02, pages 59–66, Philadelphia, Pennsylvania, USA.

James R. Curran. 2004. *From Distributional to Semantic Similarity*. Ph.D. thesis, University of Edinburgh.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 897–906, Honolulu, Hawaii.

Jennifer Gillenwater, Kuzman Ganchev, João Graça, Fernando Pereira, and Ben Taskar. 2010. Sparsity in dependency grammar induction. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 194–199, Uppsala, Sweden.

Amit Goyal and Hal Daumé, III. 2011. Generating semantic orientation lexicon using large data and thesaurus. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, WASSA '11, pages 37–43, Portland, Oregon, USA.

Birgit Hamp and Helmut Feldweg. 1997. GermaNet - a Lexical-Semantic Net for German. In *In Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15, Madrid, Spain.

William P. Headden, III, David McClosky, and Eugene Charniak. 2008. Evaluating unsupervised part-of-speech tagging for grammar induction. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 329–336, Manchester, United Kingdom.

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

Richard Johansson and Pierre Nugues. 2008. The effect of syntactic representation on semantic role labeling. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 393–400, Manchester, United Kingdom.

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

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

Dekang Lin. 1997. Using syntactic dependency as local context to resolve word sense ambiguity. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, ACL '98, pages 64–71, Madrid, Spain.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics - Volume 2*, COLING '98, pages 768–774, Montreal, Quebec, Canada.

David Marecek and Milan Straka. 2013. Stop-probability estimates computed on a large corpus improve Unsupervised Dependency Parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 281–290, Sofia, Bulgaria.

Marie-Catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2006*, pages 449–454, Genova, Italy.

George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38:39–41.

Yusuke Miyao, Rune Stre, Kenji Sagae, Takuya Matsuzaki, and Jun'ichi Tsujii. 2008. Task-oriented evaluation of syntactic parsers and their representations. In *Proceeding of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 46–54, Columbus, Ohio.

Yasaman Motazedi, Mark Dras, and François Lareau. 2012. Is bad structure better than no structure?: Unsupervised parsing for realisation ranking. In *Proceedings of the 24th International Conference on Computational Linguistics*, COLING '12, pages 1811–1830, Mumbai,India.

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

Joakim Nivre and Sandra Kübler. 2006. Dependency parsing. In *Tutorial at COLING-ACL*, Sydney, Australia.

Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 938–947, Singapore.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. WordNet::Similarity: measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, HLT-NAACL–Demonstrations '04, pages 38–41, Boston, Massachusetts, USA.

Matthias Richter, Uwe Quasthoff, Erla Hallsteinsdóttir, and Chris Biemann. 2006. Exploiting the Leipzig Corpora Collection. In *Proceedings of the IS-LTC 2006*, pages 68–73, Ljubljana, Slovenia.

Martin Riedl and Chris Biemann. 2013. Scaling to large^3 data: An efficient and effective method to compute distributional thesauri. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*, pages 884–890, Seattle, WA, USA.

Helmut Schmid. 1997. Probabilistic part-of-speech tagging using decision trees. In Daniel Jones and Harold Somers, editors, *New Methods in Language Processing*, Studies in Computational Linguistics, pages 154–164. UCL Press, London, GB.

Roy Schwartz, Omri Abend, Roi Reichart, and Ari Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *Proceedings of the 49nd Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 663–672, Portland, Oregon, USA.

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

Anders Søgaard. 2012. Unsupervised dependency parsing without training. *Natural Language Engineering*, 18(02):187–203.

Menno van Zaanen and University of Leeds. School of Computer Studies. 2001. *Building Treebanks Using a Grammar Induction System*. Research report series. University of Leeds, School of Computer Studies.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, pages 1015–1021, Geneva, Switzerland.

A Markovian approach to distributional semantics with application to semantic compositionality

Édouard Grave
EECS Department
UC Berkeley
grave@berkeley.edu

Guillaume Obozinski
LIGM – Université Paris-Est
École des Ponts – ParisTech
guillaume.obozinski
@imagine.enpc.fr

Francis Bach
Inria – Sierra project-team
École Normale Supérieure
francis.bach@ens.fr

Abstract

In this article, we describe a new approach to distributional semantics. This approach relies on a generative model of sentences with latent variables, which takes the syntax into account by using syntactic dependency trees. Words are then represented as posterior distributions over those latent classes, and the model allows to naturally obtain in-context and out-of-context word representations, which are comparable. We train our model on a large corpus and demonstrate the compositionality capabilities of our approach on different datasets.

1 Introduction

It is often considered that words appearing in similar contexts tend to have similar meaning (Harris, 1954). This idea, known as the *distributional hypothesis* was famously summarized by Firth (1957) as follow: "you shall know a word by the company it keeps." The distributional hypothesis has been applied in computational linguistics in order to automatically build word representations that capture their meaning. For example, simple distributional information about words, such as co-occurence counts, can be extracted from a large text corpus, and used to build a vectorial representation of words (Lund and Burgess, 1996; Landauer and Dumais, 1997). According to the distributional hypothesis, two words having similar vectorial representations must have similar meanings. It is thus possible and easy to compare words using their vectorial representations.

In natural languages, sentences are formed by the *composition* of simpler elements: words. It is thus reasonable to assume that the meaning of a sentence is determined by combining the meanings of its parts and the syntactic relations between them. This principle, often attributed to the German logician Frege, is known as *semantic compositionality*. Recently, researchers in computational linguistics started to investigate how the principle of compositionality could be applied to distributional models of semantics (Clark and Pulman, 2007; Mitchell and Lapata, 2008). Given the representations of individual words, such as *federal* and *agency*, is it possible to combine them in order to obtain a representation capturing the meaning of the noun phrase *federal agency*?

Most approaches to distributional semantics represent words as vectors in a high-dimensional space and use linear algebra operations to combine individual word representations in order to obtain representations for complex units. In this article, we propose a probabilistic approach to distributional semantics. This approach is based on the generative model of sentences with latent variables, which was introduced by Grave et al. (2013). We make the following contributions:

- Given the model introduced by Grave et al. (2013), we describe how in-context and ouf-of-context words can be represented by posterior distributions over latent variables (section 4).
- We evaluate out-of-context representations on human similarity judgements prediction tasks and determine what kind of semantic relations are favored by our approach (section 5).
- Finally, we evaluate in-context representations on two similarity tasks for short phrases (section 6).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1447–1456, Dublin, Ireland, August 23-29 2014.

2 Related work

Most approaches to distributional semantics are based on vector space models (VSM), in which words are represented as vectors in a high-dimensional space. These vectors are obtained from a large text corpus, by extracting distributional information about words such as the contexts in which they appear. A corpus is then represented as a word-by-context co-occurence matrix. Contexts can be defined as documents in which the target word appear (Deerwester et al., 1990; Landauer and Dumais, 1997) or as words that appear in the neighbourhood of the target word, for example in the same sentence or in a fixed-size window around the target word (Schutze, 1992; Lund and Burgess, 1996).

Next to vector space models, other approaches to distributional semantics are based on probabilistic models of documents, such as probabilistic latent semantic analysis (pLSA) introduced by Hofmann (1999) and which is inspired by latent semantic analysis, or latent Dirichlet allocation (LDA), introduced by Blei et al. (2003). In those models, each document is viewed as a mixture of k topics, where each topic is a distribution over the words of the vocabulary.

The previous models do not take into account the linguistic structure of the sentences used to build word representations. Several models have been proposed to address this limitation. In those models, the contexts are defined by using the syntactic relations between words (Lin, 1998; Curran and Moens, 2002; Turney, 2006; Padó and Lapata, 2007; Baroni and Lenci, 2010). For example, two words are considered in the same context if there exists a syntactic relation between them, or if there is a path between them in the dependency graph.

One of the first approaches to semantic compositionality using vector space models was proposed by Mitchell and Lapata (2008). In this study, individual word representations are combined using linear algebra operations such as addition, componentwise multiplication, tensor product or dilation. Those different composition operations are then used to disambiguate intransitive verbs given a subject (Mitchell and Lapata, 2008) or to compute similarity scores between pairs of small phrases (Mitchell and Lapata, 2010).

Another approach to semantic compositionality is to learn the function used to compose individual word representations. First, a semantic space containing representations for both individual words and phrases is built. For example, the words *federal*, *agency* and the phrase *federal agency* all have a vectorial representation. Then, a function mapping individual word representations to phrase representations can be learnt in a supervised way. Guevara (2010) proposed to use partial least square regression to learn this function. Similarly, Baroni and Zamparelli (2010) proposed to learn a matrix $\mathbf{A}$ for each adjective, such that the vectorial representation $\mathbf{p}$ of the adjective-noun phrase can be obtained from the vectorial representation $\mathbf{b}$ of the noun by the matrix-vector multiplication:

$$\mathbf{p} = \mathbf{A}\mathbf{b}.$$

Socher et al. (2012) later generalized this model by proposing to represent each node in a parse tree by a vector capturing the meaning and a matrix capturing the compositional effects. A composition function, inspired by artificial neural networks, is recursively applied in the tree to compute those representations.

Following the theoretical framework introduced by Coecke et al. (2010), Grefenstette and Sadrzadeh (2011) proposed to represent relational words (such as verbs) by tensors and theirs arguments (such as nouns) by vectors. Composing a relational word with its arguments is then performed by taking the pointwise product between the tensor and the Kronecker product of the vectors representing the arguments. Jenatton et al. (2012) and Van de Cruys et al. (2013) proposed two approaches to model subject-verb-object triples based on tensor factorization.

Finally, research in computation of word meaning in context is closely related to distributional semantic compositionality. Erk and Padó (2008) proposed a structured vector space model in which a word is represented by multiple vectors, capturing its meaning but also the selectional restrictions it has for the different arguments. Those different vectors can then be combined to compute a word representation in context. This model was later generalized by Thater et al. (2010). Dinu and Lapata (2010) introduced a probabilistic model for computing word representations in context. In their approach, words are represented as probability distributions over latent senses.

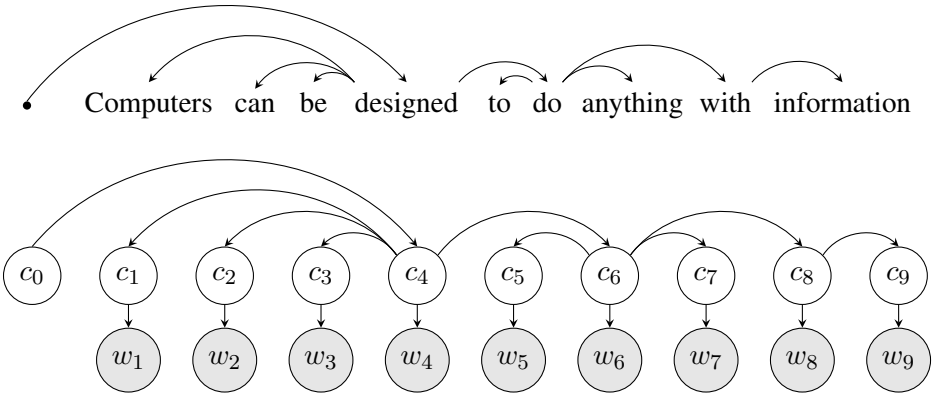


Figure 1: Example of a dependency tree and its corresponding graphical model.

3 Model of semantics

In this section we briefly review the generative model of sentences introduced by Grave et al. (2013), and which serves as the basis of our approach to distributional semantics.

3.1 Generative model of sentences

We denote the tokens of a sentence of length K by the K-uple $\mathbf{w} = (w_1, ..., w_K) \in \{1, ..., V\}^K$, where V is the size of the vocabulary and each integer represents a word. We suppose that each token w_k is associated to a corresponding semantic class $c_k \in \{1, ..., C\}$, where C is the number of semantic classes. Finally, the syntactic dependency tree corresponding to the sentence is represented by the function $\pi : \{1, ..., K\} \mapsto \{0, ..., K\}$, where $\pi(k)$ represents the parent of word k and 0 is the root of the tree (which is not associated to a word).

Given a tree π, the semantic classes and the words of a sentence are generated as follows. The semantic class of the root of the tree is set to a special start symbol, represented by the integer 0.[1] Then, the semantic classes corresponding to words are recursively generated down the tree: each semantic class c_k is drawn from a multinomial distribution $p_T(c_k \mid c_{\pi(k)})$, conditioned on the semantic class $c_{\pi(k)}$ of its parent in the tree. Finally, each word w_k is also drawn from a multinomial distribution $p_O(w_k \mid c_k)$, conditioned on its corresponding semantic class c_k. Thus, the joint probability distribution on words and semantic classes can be factorized as

$$p(\mathbf{w}, \mathbf{c}) = \prod_{k=1}^{K} p_T(c_k \mid c_{\pi(k)}) p_O(w_k \mid c_k),$$

where the variable $c_0 = 0$ represents the root of the tree. The initial class probability distribution $p_T(c_k \mid c_0 = 0)$ is parameterized by the probability vector $\mathbf{q}$, while the transition probability distribution between classes $p_T(c_k \mid c_{\pi(k)})$ and the emission probability distribution $p_O(w_k \mid c_k)$ are parameterized by the stochastic matrices $\mathbf{T}$ and $\mathbf{O}$ (*i.e.*, matrices with non-negative elements and unit-sum columns). This model is a hidden Markov model on a tree (instead of a chain). See Fig. 1 for an example of a sentence and its corresponding graphical model.

3.2 Corpus and learning

We train the generative model of sentences on the ukWac corpus (Baroni et al., 2009). This corpus, which contains approximately 1.9 billions tokens, was POS-tagged and lemmatized using TreeTagger (Schmid, 1994) and parsed using MaltParser (Nivre et al., 2007). Each word of our vocabulary is a pair of lemma and its part-of-speech. We perform smoothing by only keeping the V most frequent pairs, the infrequent ones being replaced by a common token. The parameters $\theta = (\mathbf{q}, \mathbf{T}, \mathbf{O})$ of the model are learned using the algorithm described by Grave et al. (2013). The number of latent states C and the number of lemma/POS pairs V were set using the development set of Bruni et al. (2012).

[1] We recall that the semantic classes corresponding to words are represented by integers between 1 and C.

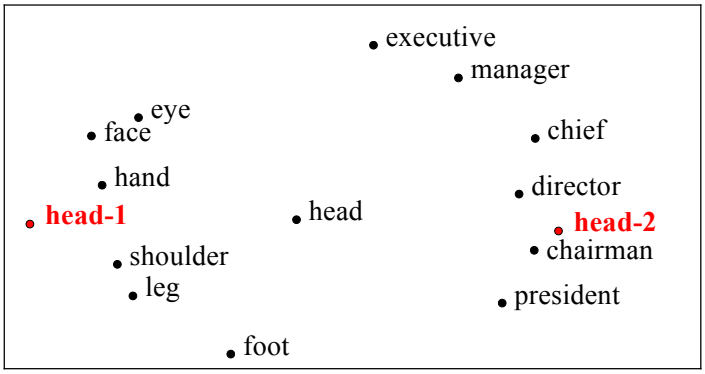


Figure 2: Comparison of out-of-context (black) and in-context (red) word representations. The two-dimensional visualization is obtained by using multidimensional scaling (Borg, 2005). See text for details.

4 Word representations

Given a trained hidden Markov model, we now describe how to obtain word representations, for both in-context and out-of-context words. In both cases, words will be represented as a probability distribution over the latent semantic classes.

In-context word representation. Obtaining a representation of a word in the context of a sentence is very natural using the model introduced in the previous section: we start by parsing the sentence in order to obtain the syntactic dependency tree. We then compute the posterior distribution of semantic classes c for that word, and use this probability distribution to represent the word. More formally, given a sentence $\mathbf{w} = (w_1, ..., w_K)$, the kth word of the sentence is represented by the vector $\mathbf{u}^k \in \mathbb{R}^C$ defined by

$$u_i^k = \mathbb{P}(C_k = i \mid W = \mathbf{w}).$$

The vector $\mathbf{u}^k$ is the posterior distribution of latent classes corresponding to the kth word of the sentence, and thus, sums to one. It is efficiently computed using the message passing algorithm (a.k.a. forward-backward algorithm for HMM).

Out-of-context representation. In order to obtain word representations that are independent of the context, we compute the previously introduced in-context representations on a very large corpus, and for each word type, we average all the in-context representations for all the occurrences of that word type in the corpus. More formally, given a large set of pairs of tokens and their in-context representations $(w_k, \mathbf{u}^k) \in \mathbb{N} \times \mathbb{R}^C$, the representation of the word type a is the vector $\mathbf{v}^a \in \mathbb{R}^C$, defined by

$$\mathbf{v}^a = \frac{1}{Z_a} \sum_{k \,:\, w_k = a} \mathbf{u}^k,$$

where Z_a is the number of occurrences of the word type a. The vector $\mathbf{v}^a$ is thus the posterior distribution of semantic classes averaged over all the occurrences of word type a.

Comparing in-context and out-of-context representations. Since in-context and out-of-context word representations are defined on the same space (the simplex of dimension C) it is possible to compare in-context and out-of-context representations easily. As an example, we have plotted in Figure 2 the out-of-context representation for the words *head, president, chief, chairman, director, executive, eye, face, shoulder, hand, leg, etc.* and the in-context representations for the word *head* in the context of the two following sentences:

1. *The nurse stuck her* head *in the room to announce that Dr. Reitz was on the phone.*
2. *A well-known Wall Street figure may join the Cabinet as* head *of the Treasury Department.*

Distance	RG65	WS353
Cosine	0.68	0.50
Kullback-Leibler	0.69	0.47
Jensen-Shannon	0.72	0.50
Hellinger	0.73	0.51
Agirre et al. (BoW)	0.81	0.65

Distance	SIM.	REL.
Cosine	0.68	0.34
Kullback-Leibler	0.64	0.31
Jensen-Shannon	0.69	0.33
Hellinger	0.70	0.34
Agirre et al. (BoW)	0.70	0.62

Table 1: Left: Spearman's rank correlation coefficient ρ between human and distributional similarity, on the RG65 and WORDSIM353 datasets. Right: Spearman's rank correlation coefficient ρ between human and distributional similarity on two subsets (similarity *v.s.* relatedness) of the WORDSIM353 dataset.

The two-dimensional visualization is obtained by using multidimensional scaling (Borg, 2005). First of all, we observe that the words are clustered in two groups, one containing words belonging to the *body part* class, the other containing words belonging to the *leader* class, and the word *head*, appears between those two groups. Second, we observe that the in-context representations are shifted toward the cluster corresponding to the disambiguated sense of the ambiguous word *head*.

5 Out-of-context evaluation

In this section, we evaluate out-of-context word representations on a similarity prediction task and determine what kind of semantic relations are favored by our approach.

5.1 Similarity judgements prediction

In word similarity prediction tasks, pairs of words are presented to human subjects who are asked to rate the relatedness between those two words. These human similarity scores are then compared to distributional similarity scores induced by our models, by computing the correlation between them.

Methodology. We use the RG65 dataset, introduced by Rubenstein and Goodenough (1965) and the WORDSIM353 dataset, collected by Finkelstein et al. (2001). These datasets comprise 65 and 353 word pairs respectively. Human subjects rated the relatedness of those word pairs. We use the Spearman's rank correlation coefficient ρ to compare human and distributional score distributions.

Comparison of similarity measures. Since words are represented by posterior distributions over latent semantic classes, we have considered distances (or divergences) that are adapted to probability distributions to compute the similarity between word representations: the symmetrised Kullback-Leibler divergence, the Jensen-Shannon divergence, and the Hellinger distance. We use the opposite of these dissimilarity measures in order to obtain similarity scores. We also included the cosine similarity measure as a baseline, as it is widely used in the field of distributional semantics.

We report results on both datasets in Table 1. Unsurprisingly, we observe that the dissimilarity measures giving the best results are the one tailored for probability distribution, namely the Jensen-Shannon divergence and the Hellinger distance. The Kullback-Leibler divergence is too sensitive to fluctuations of small probabilities and thus does not perform as well as other similarity measures between probability distributions. In the following, we will use the Hellinger distance. It should be noted that the results reported by Agirre et al. (2009) were obtained using a corpus containing 1.6 terawords, making it 1,000 times larger than ours. They also report results for various corpus sizes, and when using a corpus whose size is comparable to ours, their result on WORDSIM353 drops to 0.55.

Relatedness *v.s.* similarity. As noted by Agirre et al. (2009), words might be rated as related for different reasons since different kinds of semantic relations exist between word senses. Some words, such as *telephone* and *communication* might even be rated as related because they belong to the same semantic field. Thus, they proposed to split the WORDSIM353 dataset into two subsets: the first one comprising words that are *similar*, *i.e.*, synonyms, antonyms and hyperonym-hyponym and the second

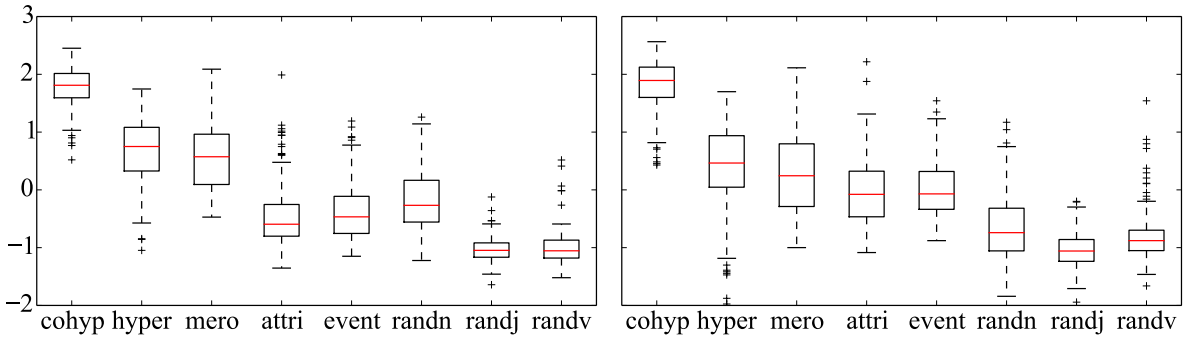


Figure 3: Similarity score distributions for various semantic relations on the BLESS dataset, without using the transition matrix (left) and with using the transition matrix (right) for comparing adjectives and verbs with nouns.

one comprising words that are *related*, *i.e.*, meronym-holonym and topically related words. We report results on these two subsets in Table 1. We observe that our model capture *similarity* ($\rho = 0.70$) much better than *relatedness* ($\rho = 0.34$). This is not very surprising since our model takes the syntax into account.

5.2 Semantic relations captured by our word representations

As we saw in the previous section, different semantic relations between words are not equally captured by our word representations. In this section, we thus investigate which kind of semantic relations are favored by our approach.

The BLESS dataset. The BLESS dataset (Baroni and Lenci, 2011) comprises 200 concrete concepts and eight relations. For each pair of concept-relation, a list of related words, referred to as relatum, is given. Five semantic relations are considered: *co-hyponymy*, *hypernymy*, *meronymy*, *attribute* and *event*. The *attribute* relation means that the relatum is an adjective expressing an attribute of the concept, while the *event* relation means that the relatum is a verb designing an activity or an event in which the concept is involved. The dataset also contains three *random* relations (*randn*, *randj* ans *randv*), obtained by the association of a random relatum, for different POS: noun, adjective and verb.

Methodology. We follow the evaluation proposed by the authors: for each pair of concept-relation, we keep the score of the most similar relatum associated to that pair of concept-relation. Thus, for each concept, we have eight scores, one for each relation. We normalize these eight scores (mean: 0, std: 1), in order to reduce concept-specific effects. We then report the score distributions for each relation as box plots in Figure 3 (left).

Results. We observe that the co-hyponymy relation is the best captured relation by a large margin. It is followed by the hypernymy and meronymy relations. The random noun relation is prefered over the attribute and the event relations. This happens because words with different part-of-speeches tend to appear in different semantic classes. It is thus impossible to compare words with different parts-of-speeches and thus to capture relation such as the event or the attribute relation as defined in the BLESS dataset. It is however possible to make a more principled use of the model to overcome this issue.

Comparing adjectives with nouns and nouns with verbs. In syntactic relations between nouns and adjectives, the noun is the head word and the adjective is the dependent. Similarly, in syntactic relations between nouns and verbs, most often the verb is the head and the noun is the dependent. Given a vector $\mathbf{v}_a$ representing an adjective and a vector $\mathbf{v}_n$ representing a noun, it is thus natural to left multiply them by the transition matrix of the model to obtain a vector $\mathbf{u}_a$ comparable to nouns and a vector $\mathbf{u}_n$ comparable to verbs:

$$\mathbf{u}_a = \mathbf{T}^\top \mathbf{v}_a \quad \text{and} \quad \mathbf{u}_n = \mathbf{T}^\top \mathbf{v}_n.$$

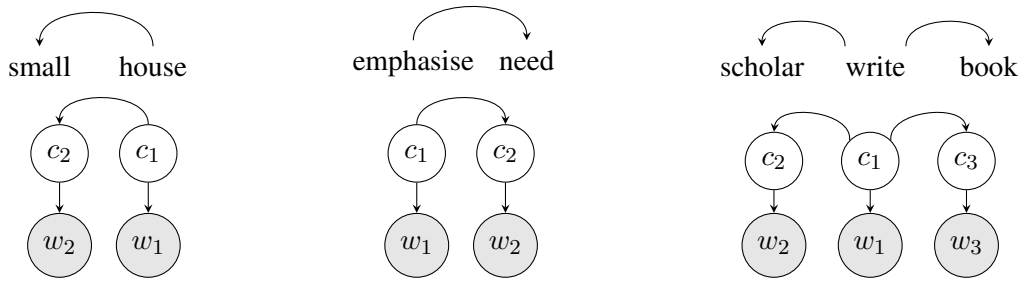


Figure 4: Graphical models used to compute in-context word representations for the compositional tasks.

We report in Figure 3 (right) the new score distributions obtained when adjective and noun representations are transformed before being compared to nouns and verbs. We observe that, when using these transformations, the attribute and event relations are better captured than the random relations. This demonstrates that the transition matrix $\mathbf{T}$ captures selectional preferences.

6 Compositional semantics

So far, we have only evaluated how well our representations are able to capture the meaning of words taken as individual and independent units. However, natural languages are highly compositional, and it is reasonable to assume that the meaning of a sentence or a phrase can be deduced from the meanings of its parts and the syntactic relations between them. This assumption is known as the principle of semantic compositionality.

In this section, we thus evaluate our representations on semantic composition tasks. More precisely, we determine if using in-context word representations helps to compute the similarity between short phrases such as adjective-noun, verb-object, compound-noun or subject-verb-object phrases. We use two datasets of human similarity scores, introduced respectively by Mitchell and Lapata (2010) and Grefenstette and Sadrzadeh (2011).

6.1 Methodology

We compare different ways to obtain a representation of a short phrase given our model. First, as a baseline, we represent a phrase by the out-of-context representation of its head word. In that case, there is no composition at all. Second, following Mitchell and Lapata (2008), we represent a phrase by the sum of the out-of-context representations of the words forming that phrase. Third, we represent a phrase by the in-context representation of its head word. Finally, we represent a phrase by the sum of the two in-context representations of the words forming that phrase. The graphical models used to compute in-context word representations are represented in Fig 4. The probability distribution $p(c_1)$ of the head's semantic class is set to the uniform distribution (and not to the initial class distribution $p_T(c_k \mid c_0 = 0)$).

6.2 Datasets

The first dataset we consider was introduced by Mitchell and Lapata (2010), and is composed of pairs of adjective-noun, compound-noun and verb-object phrases, whose similarities were evaluated by human subjects on a $1-7$ scale. We compare our results with the one reported by (Mitchell and Lapata, 2010). The second dataset we consider was introduced by Grefenstette and Sadrzadeh (2011). Each example of this dataset consists in a triple of subject-verb-object, forming a small transitive sentence, and a landmark verb. Human subjects were asked to evaluate the similarity between the verb and its landmark in the context of the small sentence. Following Van de Cruys et al. (2013), we compare the contextualized verb with the non-contextualized landmark, meaning that the landmark is always represented by its out-of-context representation. We do so because it is believed to better capture the compositional ability of our model and it works better in practice. We compare our results with the one reported by Van de Cruys et al. (2013).

	AN	NN	VN
head (out-of-context)	0.44	0.26	0.41
add (out-of-context)	0.50	0.45	0.42
head (in-context)	0.49	0.42	**0.43**
add (in-context)	**0.51**	0.46	0.41
M&L (vector space model)	0.46	**0.49**	0.38
Humans	0.52	0.49	0.55

	SVO
head (out-of-context)	0.25
add (out-of-context)	0.25
head (in-context)	**0.41**
add (in-context)	0.40
Van de Cruys et al.	0.37
Humans	0.62

Table 2: Spearman's rank correlation coefficients between human similarity judgements and similarity computed by our models on the Mitchell and Lapata (2010) dataset (left) and on the Grefenstette and Sadrzadeh (2011) dataset (right). AN stands for adjective-noun, NN stands for compoundnoun and VN stands for verb-object.

6.3 Discussion

Before discussing the results, it is interesting to note that our approach provides a way to evaluate the importance of disambiguation for compositional semantics. Indeed, the in-context representations proposed in this paper are a way to disambiguate their out-of-context equivalents. It was previously noted by Reddy et al. (2011) that disambiguating the vectorial representations of words improve the performance on compositional tasks.

Mitchell and Lapata (2010) dataset. We report results on the Mitchell and Lapata (2010) dataset in Table 2 (left). Overall, in-context representations achieves better performance than out-of-context ones. For the adjective-noun pairs and the verb-noun pairs, using only the in-context representation of the head word works almost as well (AN) or even better (VN) than adding the representations of the two words forming a pair. This means that for those particular tasks, disambiguation plays an important role. On the other hand, this is not the case for the noun-noun pairs. On that task, most improvement over the baseline comes from the *add* operation.

Grefenstette and Sadrzadeh (2011) dataset. We report results in Table 2 (right). First, we observe that in-context representations clearly outperform out-of-context ones. Second, we note that adding the subject, object and verb representations does not improve the result over only using the representation of the verb. These two conclusions are not really surprising since this task is mainly a disambiguation task, and disambiguation is achieved by using the in-context representations. We also note that our approach yields better results than those obtained by Van de Cruys et al. (2013), while their method was specifically designed to model subject-verb-object triples.

7 Conclusion and future work

In this article, we introduced a new approach to distributional semantics, based on a generative model of sentences. This model is somehow to latent Dirichlet allocation as structured vector space models are to latent semantic analysis. Indeed, our approach is based on a probabilistic model of sentences, which takes the syntax into account by using dependency trees. Similarly to LDA, our model can be viewed as a topic model, the main difference being that the topics are generated using a Markov process on a syntactic dependency tree instead of using a Dirichlet process.

The approach we propose seems quite competitive with other distributional models of semantics. In particular, we match or outperform state-of-the-art methods on semantic compositionality tasks. Thanks to its probabilistic nature, it is very easy to derive word representations for various tasks: the same model can be used to compute in-context word representations for adjective-noun phrases, subject-verb-object triples or even full sentences, which is not the case of the tensor based approach proposed by Van de Cruys et al. (2013).

Currently, the model of sentences does not use the dependency labels, which is the most significant limitation that we would like to address in future work. We also plan to explore spectral methods (Anandkumar et al., 2012) to provide better initialization for learning the parameters of the model. Indeed, we believe this could speed up learning and yields better results, since the expectation-maximization algorithm is quite sensitive to bad initialization. Finally, the code corresponding to this article will be available on the first author webpage.

Acknowledgments

Edouard Grave is supported by a grant from INRIA (Associated-team STATWEB). Francis Bach is partially supported by the European Research Council (SIERRA Project)

References

E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, and A. Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky. 2012. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*.

M. Baroni and A. Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

M. Baroni and A. Lenci. 2011. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10. Association for Computational Linguistics.

M. Baroni and R. Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.

D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*.

I. Borg. 2005. *Modern multidimensional scaling: Theory and applications*. Springer.

E. Bruni, G. Boleda, M. Baroni, and N. K. Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 136–145. Association for Computational Linguistics.

S. Clark and S. Pulman. 2007. Combining symbolic and distributional models of meaning. In *AAAI Spring Symposium: Quantum Interaction*, pages 52–55.

B. Coecke, M. Sadrzadeh, and S. Clark. 2010. Mathematical foundations for a compositional distributional model of meaning. *arXiv preprint arXiv:1003.4394*.

J. R. Curran and M. Moens. 2002. Scaling context space. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*.

G. Dinu and M. Lapata. 2010. Measuring distributional similarity in context. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

K. Erk and S. Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.

L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*.

J. R. Firth. 1957. *A synopsis of linguistic theory, 1930-1955*.

E. Grave, G. Obozinski, and F. Bach. 2013. Hidden Markov tree models for semantic class induction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*.

E. Grefenstette and M. Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

E. Guevara. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*.

Z. S. Harris. 1954. *Distributional structure*. Springer.

T. Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*.

R. Jenatton, N. Le Roux, A. Bordes, and G. Obozinski. 2012. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems 25*.

T. K Landauer and S. T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-volume 2*.

K. Lund and C. Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*.

J. Mitchell and M. Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*.

J. Mitchell and M. Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*.

J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*.

S. Padó and M. Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*.

S. Reddy, I. P. Klapaftis, D. McCarthy, and S. Manandhar. 2011. Dynamic and static prototype vectors for semantic composition. In *IJCNLP*, pages 705–713.

H. Rubenstein and J. B. Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

H. Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of international conference on new methods in language processing*.

H. Schutze. 1992. Dimensions of meaning. In *Supercomputing'92. Proceedings*. IEEE.

R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

S. Thater, H. Fürstenau, and M. Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

P. D. Turney. 2006. Similarity of semantic relations. *Computational Linguistics*.

T. Van de Cruys, T. Poibeau, and A. Korhonen. 2013. A tensor-based factorization model of semantic compositionality. In *Proceedings of NAACL-HLT*.

A Beam-Search Decoder for Disfluency Detection

Xuancong Wang[1,3] **Hwee Tou Ng**[1,2] **Khe Chai Sim**[2]
[1]NUS Graduate School for Integrative Sciences and Engineering
[2]Department of Computer Science, National University of Singapore
[3]Human Language Technology, Institute for Infocomm Research, Singapore
xuancong84@gmail.com, {nght, simkc}@comp.nus.edu.sg

Abstract

In this paper[1], we present a novel beam-search decoder for disfluency detection. We first propose node-weighted max-margin Markov networks (M3N) to boost the performance on words belonging to specific part-of-speech (POS) classes. Next, we show the importance of measuring the quality of cleaned-up sentences and performing multiple passes of disfluency detection. Finally, we propose using the beam-search decoder to combine multiple discriminative models such as M3N and multiple generative models such as language models (LM) and perform multiple passes of disfluency detection. The decoder iteratively generates new hypotheses from current hypotheses by making incremental corrections to the current sentence based on certain patterns as well as information provided by existing models. It then rescores each hypothesis based on features of lexical correctness and fluency. Our decoder achieves an edit-word F1 score higher than all previous published scores on the same data set, both with and without using external sources of information.

1 Introduction

Disfluency detection is a useful and important task in Natural Language Processing (NLP) because spontaneous speech contains a significant proportion of disfluency. The disfluencies in speech introduce noise in downstream tasks like machine translation and information extraction. Thus, the task of disfluency detection not only can help improve the readability of automatically transcribed speech, but also the performance of downstream NLP tasks.

There are mainly two types of disfluencies: filler words and edit words. Filler words include filled pauses (e.g., 'uh', 'um') and discourse markers (e.g., "I mean", "you know"). They are insertions in spontaneous speech that indicate pauses or mark boundaries in discourse. Thus, they do not convey useful content information. Edit words are words that are spoken wrongly and then corrected by the speaker. For example, consider the utterance:

Edit Filler Repair

I want a flight ~~*to Boston*~~ *uh I mean* to Denver

The phrase "to Boston" forms the edit region to be replaced by "to Denver". The words "uh I mean" are filler words that serve to cue the listener about the error and subsequent correction. So, the cleaned-up sentence would be "I want a flight to Denver", which is what the speaker originally intended to say. In general, edit words are more difficult to detect than filler words, and so edit word prediction accuracy is much lower. Thus, in this work, we mainly focus on edit word detection.

In Section 2, we briefly introduce previous work. In Section 3, we describe our improved baseline system that will be integrated into our beam-search decoder. Section 4 presents our beam-search decoder in detail. In Section 5, we describe our experiments and results. Section 6 gives the conclusion.

[1]The research reported in this paper was carried out as part of the PhD thesis research of Xuancong Wang at the NUS Graduate School for Integrated Sciences and Engineering.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1457–1467, Dublin, Ireland, August 23-29 2014.

2 Previous Work

Researchers have tried many models for disfluency detection. Johnson and Charniak (2004) proposed a TAG-based (Tree-Adjoining Grammar) noisy channel model, which showed great improvement over a boosting-based classifier (Charniak and Johnson, 2001). Maskey et al. (2006) proposed a phrase-level machine translation approach for this task. Liu et al. (2006) used conditional random fields (CRFs) (Lafferty et al., 2001) for sentence boundary and edit word detection. They showed that CRFs significantly outperformed maximum entropy models and hidden Markov models (HMM). Zwarts and Johnson (2011) extended this model using minimal expected F-loss oriented n-best reranking. Georgila (2009) presented a post-processing method during testing based on integer linear programming (ILP) to incorporate local and global constraints. In addition to textual information, prosodic features extracted from speech have been incorporated to detect edit words in some previous work (Kahn et al., 2005; Liu et al., 2006; Zhang et al., 2006). Zwarts and Johnson (2011) also trained extra language models on additional corpora, and compared the effects of adding scores from different language models as features during reranking. They reported that the language models gained approximately 3% in F1-score for edit word detection on the Switchboard development dataset. Qian and Liu (2013) proposed multi-step disfluency detection using weighted max-margin Markov networks (M3N) and achieved the highest F-score of 84.1% without using any external source of information. In this paper, we incorporate the M3N model into our beam-search decoder framework with some additional features to further improve the result.

3 The Improved Baseline System

Weighted max-margin Markov networks (M3N) (Taskar et al., 2003) have been shown to outperform CRF in (Qian and Liu, 2013), since it can balance precision and recall easily by assigning different loss penalty to different label misclassification pairs. In this work, we made use of M3N in expanding the search space and rescoring the hypotheses. To facilitate the integration of the M3N system into our decoder framework, we made several modifications that slightly improve the M3N baseline system. Our improved baseline system has two stages: the first stage is filler word prediction using M3N to detect words which can potentially be fillers, and the second stage is joint edit and filler word prediction using M3N. The output of the first stage is passed as features into the second stage. Both stages perform filler word prediction, since we found that joint edit and filler word detection performs better than edit word detection alone as edit words tend to co-occur with filler words, and the first-stage output can be fed into the second stage to extract additional features. We also augmented the M3N toolkit to support additional feature functions, allow weighting of individual nodes, and control the total number of model parameters.[2]

3.1 Node-Weighted and Label-Weighted Max-Margin Markov Networks (M3N)

A max-margin Markov network (M3N) (Taskar et al., 2003) is a sequence labeling model. It has the same structure as conditional random fields (CRF) (Lafferty et al., 2001) but with a different objective function. A CRF is trained to maximize the conditional probability of the true label sequence given the observed input sequence, while an M3N is trained to maximize the difference between the conditional probability of the true label sequence and the incorrectly predicted label sequence (i.e., maximizing the margin). Thus, we can regard M3N as a support vector machine (SVM) analogue of CRF (Suykens and Vandewalle, 1999).

The dual form of the training objective function of M3N is formulated as follows:

$$\begin{aligned} &\min_{\alpha} \frac{1}{2} C \| \sum_{\mathbf{x},\mathbf{y}} \alpha_{\mathbf{x},\mathbf{y}} \Delta \mathbf{f}(\mathbf{x},\mathbf{y}) \|_2^2 + \sum_{\mathbf{x},\mathbf{y}} \alpha_{\mathbf{x},\mathbf{y}} L(\mathbf{x},\mathbf{y}) \\ s.t. \quad &\sum_{\mathbf{y}} \alpha_{\mathbf{x},\mathbf{y}} = 1, \forall \mathbf{x} \qquad \text{and} \qquad \alpha_{\mathbf{x},\mathbf{y}} \geq 0, \forall \mathbf{x},\mathbf{y} \end{aligned} \tag{1}$$

[2]The source code of our augmented M3N toolkit can be downloaded at http://code.google.com/p/m3n-ext/

where $\mathbf{x}$ is the observed input sequence, $\mathbf{y} \in \mathcal{Y}$ is the output label sequence, $\alpha_{\mathbf{x},\mathbf{y}}$ are the dual variables to be optimized, and $C \geq 0$ is the regularization parameter to be tuned. $\Delta\mathbf{f}(\mathbf{x}, \mathbf{y}) = \mathbf{f}(\mathbf{x}, \widetilde{\mathbf{y}}) - \mathbf{f}(\mathbf{x}, \bar{\mathbf{y}})$ is the residual feature vector, where $\widetilde{\mathbf{y}}$ is the true label sequence, $\bar{\mathbf{y}}$ is the predicted label sequence given the model, and $\mathbf{f}(\mathbf{x}, \mathbf{y})$ is the feature vector. It is implemented as a sum over all nodes:

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = \sum_t \mathbf{f}(\mathbf{x}, \mathbf{y}, t) \tag{2}$$

where t is the position index of the node in the sequence. Each component of $\mathbf{f}(\mathbf{x}, \mathbf{y}, t)$ is a feature function, $f(\mathbf{x}, \mathbf{y}, t)$. For example, $f(w_0$='so', y_0='F', y_{-1}='O', $t)$ has a value of 1 only when the word at node t is 'so', the label at node t is 'F' (filler word), and the label at node $(t-1)$ is 'O' (outside edit/filler region, i.e., fluent). The maximum length of the y history (for this feature function, it is 2 since only y_0 and y_{-1} are covered) is called the *clique order* of the feature. $L(\mathbf{x}, \mathbf{y})$ is the loss function. A standard M3N uses an unweighted hamming loss, which is the number of incorrect nodes:

$$L(\mathbf{x}, \mathbf{y}) = \sum_t \delta(\widetilde{y}_t, \bar{y}_t) \qquad \text{where } \delta(a, b) = \begin{cases} 1, & \text{if } a = b \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Qian and Liu (2013) proposed using label-weighted M3N to balance precision and recall by adjusting the penalty on false positives and false negatives, i.e., $v(\widetilde{y}_t, \bar{y}_t)$ in Eqn. 4. In this work, we further extend this technique to individual nodes to train expert models, each specialized in a specific part-of-speech (POS) class. Our loss function is:

$$L(\mathbf{x}, \mathbf{y}) = \sum_t u_c(t) v(\widetilde{y}_t, \bar{y}_t) \delta(\widetilde{y}_t, \bar{y}_t) \qquad \text{where } u_c(t) = \begin{cases} B_c, & \text{if POS}(t) \in S_c \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

where B_c is a factor which controls the extent to which the model is biased to minimize errors on specific nodes, POS(t) is the POS tag of the word at node t, and S_c is the set of POS tags corresponding to that expert class c. We show that by integrating these expert models into our decoder framework, we can achieve further improvement.

3.2 Features

The feature templates for filler word and edit word prediction are listed in Table 1 and Table 2 respectively. w_i refers to the word at the i^{th} position relative to the current node; *window size* is the maximum number of words before and after the current word that the template covers, e.g., $w_{-1}w_0$ with a window size of 4 means $w_{-4}w_{-3}$, $w_{-3}w_{-2}$, ..., w_3w_4; p_i refers to the POS tag at the i^{th} position relative to the current node; $w_{i\sim j}$ refers to any word from the i^{th} position to the j^{th} position relative to the current node; $w_{i,\neq F}$ refers to the i^{th} word (w.r.t. the current node) not being a filler word; the multi-pair comparison function $I(a, b, c, ...)$ indicates whether each pair (a and b, b and c, and so on) are identical, for example, if $a = b \neq c = d$, it will output "101" ('1' for being equal, '0' for being unequal); and *ngram-score* features are the natural logarithm of the following probabilities: $P(w_{-3}, w_{-2}, w_{-1}, w_0)$, $P(w_0|w_{-3}, w_{-2}, w_{-1})$, $P(w_{-3}, w_{-2}, w_{-1})$, $P(\langle/s\rangle|w_{-3}, w_{-2}, w_{-1})$, $P(w_{-3})$, $P(w_{-2})$, $P(w_{-1})$, $P(w_0)$ ("$\langle/s\rangle$" denotes sentence-end). We use language models (LM) in two ways: individual n-gram scores as M3N features, and an overall sentence-level score for rescoring in our beam-search decoder. Our experiments show that this way of using LM gives the best performance.

We set the frequency pruning threshold to 5, so that the resulting model has about the same total number of parameters (7.6M) as (Qian and Liu, 2013). The clique order for each template is determined by considering the total number of features given that template. For example, for pause duration, there are 10 features (after cumulative binning), so we can set its clique order to 3 since there will be $10 \times 3^3 = 270$ weights; but for word 3-grams, there are 5M features, so setting its clique order to 3 or 2 will give rise to too many weights ($5\text{M} \times 3^3 = 135\text{M}$ for order 3; $5\text{M} \times 3^2 = 45\text{M}$ for order 2), thus we will reduce its clique order to 1. The same principle applies to other feature templates.

Feature Template	Window Size	Clique Order
w_0	4	1
$w_{-1}w_0$	4	2
$I(w_i, w_j)$	10	2
$I(w_i, w_j, w_{i+1}, w_{j+1})$	10	2
p_0	4	1
$p_{-1}p_0$	4	2
$p_{-2}p_{-1}p_0$	4	2
$I(p_i, p_j)$	10	2
$I(p_i, p_j, p_{i+1}, p_{j+1})$	10	2
transitions	0	3

Table 1: Feature templates for filler word prediction

Feature Template	Window Size	Clique Order
$w_{-2}w_{-1}w_0$	4	2
$I(w_i, w_j)(w_i$ if w_i=$w_j)$	10	2
$w_0w_{-6\sim-1}$, $w_0w_{1\sim6}$	0	1
$I(p_i, p_j)$	10	3
$I(p_i, p_j)(p_i$ if p_i=$p_j)$	10	3
$p_{-1}w_0$	2	2
$w_{-1}p_0$	2	2
$w_{-2,\neq F}w_{-1,\neq F}$	0	2
$w_{-3,\neq F}w_{-2,\neq F}w_{-1,\neq F}$	0	2
$p_{-2,\neq F}p_{-1,\neq F}$	0	2
$p_{-3,\neq F}p_{-2,\neq F}p_{-1,\neq F}$	0	2
ngram-score features	0	3
pause duration before w_0	0	3
pause duration after w_0	0	3
all features for filler word prediction	same	same

Table 2: Feature templates for edit word prediction

4 The Beam-Search Decoder Framework

4.1 Motivation

There are several limitations in the current M3N or CRF approach. Firstly, current models do not measure the quality of the cleaned-up sentences, i.e., the resulting sentence after removing all predicted filler and edit words. Secondly, one pass of disfluency detection may not be sufficient to detect all disfluencies. Qian and Liu (2013) showed that we can improve the performance significantly by running a second pass of edit detection. Our preliminary experiments also show that additional passes of edit detection further improve the performance. Lastly, we find that edit word detection accuracy differs significantly on words of different POS tags (Table 3). This is because words of different POS tags have different feature distributions. Thus, depending on the POS tag of the current word, the same feature may have different implications for disfluency. For example, consider the feature $I(p_0, p_2)$. When the current word is a determiner, the feature does not strongly suggest an edit word. In "You give me a book, a pen, a pencil, ...", the determiner 'a' gets repeated. However, when the current word is a verb, the feature strongly suggests it is an edit word. In "The hardest thing for us ~~*has been*~~ is to ...", both 'has' and 'is' are third-person singular verbs. Hence, it might be helpful if we train expert models each specialized in detecting edit words belonging to a specific POS and combine them dynamically according to the POS. Motivated by the beam-search decoder for grammatical error correction (Dahlmeier and Ng, 2012) and social media text normalization (Wang and Ng, 2013), we propose a novel beam-search decoder for

disfluency detection to overcome these limitations.

POS	Freq. (%)	Edit F1 (%)
PRP	25.5	92.33
DT	14.2	88.95
IN	10.4	84.45
VBP	8.3	86.88
RB	7.1	81.78
CC	4.6	86.76
BES	4.2	93.37
NN	3.4	52.30
VBD	3.1	86.51
VB	2.1	70.42
VBZ	1.9	79.70
...	...	...

Table 3: Baseline edit F1 scores for different POS tags

4.2 General Framework

The goal of the decoder is to find the best hypothesis for a given input sentence $\mathbf{w}$. A hypothesis $\mathbf{h}$ is a label sequence, one label for every word in the sentence. To find the best hypothesis, the decoder iteratively generates new hypotheses from current ones using a set of *hypothesis producers* and rescores each hypothesis using a set of *hypothesis evaluators*. For each hypothesis produced, the decoder cleans up the sentence by removing all the predicted filler words and edit words so that subsequent operations can act on the cleaned-up sentence $\bar{\mathbf{w}}$ if needed. Each hypothesis evaluator produces a score f which measures the quality of the current hypothesis based on certain aspects of fluency specific to that hypothesis evaluator. The overall score of a hypothesis is the weighted sum of the scores from all the hypothesis evaluators:

$$score(\mathbf{h}, \mathbf{w}) = \sum_i \lambda_i f_i(\mathbf{h}, \mathbf{w}) \tag{5}$$

The weights λ_is are tuned on the development set using minimum error rate training (MERT) (Och, 2003). The decoding algorithm is shown in Algorithm 1.

In our description, h_i denotes the hypothesized label at the i^{th} position; w_i denotes the word at the i^{th} position; $|\mathbf{h}|$ denotes the length of the label sequence; $f_{M3N}(h_i, \mathbf{w})$ denotes the M3N log-posterior probability of the label (at the i^{th} position of hypothesis $\mathbf{h}$) being the hypothesized label; $f_{M3N}(\mathbf{h}, \mathbf{w})$ denotes the normalized joint log probability of hypothesis $\mathbf{h}$ given the M3N model ('normalized' means divided by the length of the label sequence); $\bar{\mathbf{w}}$ denotes the cleaned-up sentence; $f_{LM}(\bar{\mathbf{w}}) = f_{LM}(\mathbf{h}, \mathbf{w})$ denotes the language model score of the sentence $\bar{\mathbf{w}}$ (cleaned up according to hypothesis $\mathbf{h}$) divided by sentence length; and $\bar{\mathbf{h}}$ denotes the sub-hypothesis obtained by running M3N on the cleaned-up sentence with updated features. Note that a sub-hypothesis will have a shorter label sequence if some words are labeled as filler word or edit word in the parent hypothesis. We can obtain $\mathbf{h}$ from $\bar{\mathbf{h}}$ by inserting all predicted filler and edit words from the parent hypothesis into the sub-hypothesis so that its label sequence has the same length as the original sentence.

4.3 Hypothesis Producers

The goal of hypothesis producers is to create a search space for rescoring using various hypothesis evaluators. Based on the information provided by the existing models and certain patterns where disfluencies may occur, we propose the following hypothesis producers for our beam-search decoder:

Confusable-phrase-dictionary: The motivation of using this hypothesis producer is to hypothesize labels for phrases which are commonly misclassified in the development data. We build a dictionary of

Algorithm 1
The beam-search decoding algorithm for a sentence. S: hypothesis stack; $\mathbf{h}$: hypothesis; $\mathbf{f}$: hypothesis evaluator score vector; $\mathbf{\Lambda}$: hypothesis evaluator weight vector

INPUT: a sentence $\mathbf{w}$ with N words
OUTPUT: a sequence of N labels, from {E, F, O}
initialize hypothesis $\mathbf{h}_0$, h_i='O'$\forall i \in [1, N]$
$S_A \leftarrow \{\mathbf{h}_0\}, S_B \leftarrow \emptyset$
for $iter$ = [1, $maxIter$] **do**
  for each $\mathbf{h}$ in S_A **do**
    for each *producer* in hypothesisProducers **do**
      for each $\mathbf{h}'$ in *producer*($\mathbf{h}$) **do**
        compute $\mathbf{f}(\mathbf{h}', \mathbf{w})$ from hypothesisEvaluators
        compute $score(\mathbf{h}', \mathbf{w}) = \mathbf{\Lambda}^\top \cdot \mathbf{f}(\mathbf{h}', \mathbf{w})$
        $S_B \leftarrow S_B + \{\mathbf{h}'\}$
  prune S_B according to $score$
  $S_A \leftarrow S_B, S_B = \emptyset$
return $argmax_{\mathbf{h}}\{score(\mathbf{h}, \mathbf{w})\}, \mathbf{h} \in S_A$

phrases (up to 5 words) and their corresponding true labels by considering the most frequent incorrectly predicted phrases in the development set. During decoding, whenever such a phrase occurs in a sentence and its label is not the same as that in the dictionary, it is changed to that in the dictionary and a new hypothesis is produced. For example, if the phrase "you know" has occurred 1144 times and has been misclassified 31 times, out of which 9 times it should be 'O', then an entry "you know O || 1144 31 9" will be added to the dictionary. If the original sentence contains "you know" and it is not labeled as O, a new hypothesis will be generated by labeling it as 'O'.

Repetition: Whenever the i^{th} word and the j^{th} word ($j > i$) are the same, all words from the i^{th} position (inclusive) to the j^{th} position (exclusive) are labeled as edit words. For example, in "I want to be able to um I just want it more for multi-tasking", three hypotheses are produced. The first hypothesis is produced by labeling every word in "I want to be able to um" as edit words since 'I' is repeated. The second hypothesis is produced by labeling every word in "want to be able to um I just" as edit words since 'want' is repeated. The third hypothesis is produced by labeling "to be able" as edit words since 'to' is repeated. The window size within which we search for repetitions is set to 12 (i.e., $j - i \leq 12$), since the longest edit region (due to repetition) in the development set is of that size. We introduce this hypothesis producer because the baseline system tends to miss long edit regions, especially when very few words in the region are repeated. However, sometimes a speaker does change what he intends to say by aborting a sentence so that only the beginning few words are repeated, as in the above sentence.

Filler-word-marker: We trained an M3N model for filler word detection. Multiple passes of filler word detection on cleaned-up sentences can sometimes detect filler words that are missed in earlier passes. This hypothesis producer runs before every iteration starts. It performs filler word prediction and modifies the feature table by setting the filler-indicator feature to true so that subsequent operations see the updated feature. However, it does not remove filler words during the clean up process because some words are defined as both filler word and edit word simultaneously.

Edit-word-marker: We run our baseline M3N (the second stage) on the cleaned-up sentence and obtain the N-best hypotheses, i.e., the top N hypotheses $\mathbf{h}$ with $max\{f_{M3N}(\bar{\mathbf{h}}, \bar{\mathbf{w}})\}$. This producer essentially performs multiple passes of disfluency detection.

4.4 Hypothesis Evaluators

Our decoder uses the following hypothesis evaluators to select the best hypothesis:

Fluent language model score: This is the normalized language model score of the cleaned-up sentence, i.e., $f_{fluentLM}(\bar{\mathbf{w}})$. A 4-gram language model is trained on the cleaned-up version of the training

texts (both filler words and edit words are removed). This score measures the fluency of the resulting cleaned-up sentence w.r.t. a fluent language model.

Disfluent language model score: This is the normalized language model score of the cleaned-up sentence, i.e., $f_{disfluentLM}(\bar{\mathbf{w}})$. A 4-gram language model is trained on the original training texts which contain disfluencies. This score measures the fluency of the resulting cleaned-up sentence w.r.t. a disfluent language model. These two LM scores provide contrastive measures because if a cleaned up sentence still contains disfluencies, the disfluent LM will be preferred over the fluent LM.

M3N disfluent score: This is the normalized joint log probability score of the current hypothesis $\mathbf{h}$, i.e., $f_{M3N}(\mathbf{h}, \mathbf{w})$. This score measures how much the baseline M3N model favors the disfluency label assignment of the current hypothesis.

M3N fluent score: This is the normalized joint log probability score of labeling the entire cleaned-up sentence as fluent, i.e.,

$$f_{M3N}(\bar{\mathbf{h}}\text{=}\mathbf{O}, \bar{\mathbf{w}}) = \frac{1}{|\bar{\mathbf{h}}|} \sum_{i=1}^{|\bar{\mathbf{h}}|} f_{M3N}(\bar{h}_i\text{='O'}, \bar{\mathbf{w}}) \tag{6}$$

This score measures how much the baseline M3N model favors the cleaned-up sentence of the current hypothesis. It acts as a discriminative LM in measuring the fluency of the cleaned-up sentence. If the cleaned-up sentence contains disfluencies, this evaluator function will tend to give a lower score.

Expert-POS-class c disfluent score: This is the normalized joint log probability score of the current hypothesis $\mathbf{h}$ under the expert M3N model for POS class c dynamically combined with the baseline M3N model, i.e.,

$$f_c(\mathbf{h}, \mathbf{w}) = \frac{1}{|\mathbf{h}|} \sum_{i=1}^{|\mathbf{h}|} g_c(h_i, \mathbf{w}), \qquad g_c(h_i, \mathbf{w}) = \begin{cases} f_{\text{M3N-}c}(h_i, \mathbf{w}) \text{ if POS}(w_i) \in S_c \\ f_{\text{M3N}}(h_i, \mathbf{w}) \text{ if POS}(w_i) \notin S_c \end{cases} \tag{7}$$

Training of the expert M3N models is described in Section 4.6.

4.5 Integrating M3N into the Decoder Framework

In most previous work such as (Liu et al., 2006) and (Qian and Liu, 2013) that performed filler and edit word detection using sequence models, the begin-inside-end-single (BIES) labeling scheme was adopted, i.e., for edit words (E), 4 labels are defined: E_B (beginning of an edit region), E_I (inside an edit region), E_E (end of an edit region), and E_S (single-word edit region). However, since our beam-search decoder needs to change the labels dynamically among filler words, edit words, and fluent words, it will be problematic if the label sequence has to conform to the BIES constraint especially when the posteriors are concerned. Thus, we use the minimal set of labels: E (Edit word), F (Filler word), O (Outside edit and filler region).

For the first-stage filler word detection, only 'F' and 'O' are used. To compensate for degradation in performance, we increase the clique order of features to 3. We found that increasing the clique order has a similar effect as using the BIES labeling scheme. For example, $f(w_i$='so', y_0='E_B', $t)$ means the previous word is not an edit, both the current word and the next word are edit words, i.e., the previous word, the current word, and the next word can be either O-E-E or F-E-E. So in our minimal labeling scheme, this feature will be decomposed into $f(w_i$='so', y_{-1}='O', y_0='E', y_{+1}='E', $t)$ and $f(w_i$='so', y_{-1}='F', y_0='E', y_{+1}='E', $t)$, both having a higher clique order.

Our preliminary experiments show that by increasing the clique order of features while reducing the number of labels (keeping about the same total number of parameters), we can maintain the same performance. However, training takes a longer time.

4.6 POS-Class Specific Expert Models

We trained 6 expert M3N models, each focusing on disfluency prediction of words belonging to the corresponding set of POS tags. The expert M3N models are trained in the same way as the baseline M3N model, except that we increase the loss weights (Eqn. 4) if the word of that node belongs to

the corresponding POS class. That is, M3N-Expert-POS-class-1 is trained to optimize performance on words belonging to POS-class-1. Nonetheless, it can still predict disfluency for words in other POS classes, except that the error rate may be higher because of the way training is biased.

Class	POS tags	Freq. (%)	F1 range
1	RBS POS PDT NNPS HVS PRP$ BES PRP	33.5	92.3 – 100
2	MD EX CC DT VBP WP WRB	32.2	86.0 – 90.8
3	RB IN	16.7	82.8 – 83.8
4	TO VBD WDT RP JJS	5.1	80.0 – 82.1
5	VBZ VB VBN JJ	6.1	69.3 – 78.1
6	VBG NN CD JJR UH NNS NNP XX RBR	3.2	42.1 – 64.2

Table 4: POS classes for expert M3N models and their baseline F1 scores

We split all POS tags into 6 classes, by first sorting all POS tags in descending order of their F1 scores. Next, for POS tags with higher F1 scores, we form larger classes (higher total proportion), and for POS tags with lower F1 scores, we form smaller classes. The POS classes are shown in Table 4. The algorithm dynamically selects posteriors from different M3N models, depending on the POS tag of the current word.

5 Experiments

5.1 Experimental Setup

We tested all the systems on the Switchboard Treebank corpus (LDC99T42), using the same train/develop/test split as previous work (Johnson and Charniak, 2004; Qian and Liu, 2013). We removed all partial words and punctuation symbols to simulate the condition when automatic speech recognition (ASR) output is used. Our training set contains 1.3M words in 174K sentences; our development set contains 86K words in 10K sentences; and our test set contains 66K words in 8K sentences. The original system has high precision but low recall, i.e., the system tends to miss out edit words. The imbalance can be solved by setting a larger penalty for mis-labeling edits as fluent, i.e., 2 instead of 1 for the weighted hamming loss. We used the loss matrix, $v(\widetilde{y}_t, \bar{y}_t)$, in Table 5 to balance precision and recall. We set the biasing factor B_c to 2, for every class c. We also added two pause duration features (pause duration before and after the current word) from the corresponding Switchboard speech corpus (LDC97S62). We trained our acoustic model on the Fisher corpus and used it to perform forced alignment on the Switchboard corpus to obtain the word boundary time information for calculating pause durations. For the *ngram-score* features, we used the small 4-gram language model trained on the training set with filler words and edit words removed. All continuous features are quantized into 10 discrete bins using cumulative binning (Liu et al., 2006). We set $maxIter$ to 4 in Algorithm 1. The regularization parameter C is set to 0.006, obtained by tuning on the development set.

Label	E	F	O
E	0	1	2
F	1	0	2
O	1	1	0

Table 5: Weighted hamming loss, $v(\widetilde{y}_t, \bar{y}_t)$ for M3N for both stages

5.2 Results

We use the standard F1 score as our evaluation metric, the same as (Qian and Liu, 2013). Performance comparison of the baseline model and expert models on subsets belonging to specific POS classes is shown in Table 6. It shows that by assigning larger loss weights to nodes belonging to a specific POS class, we can to various extent boost the performance on words in that POS class. However, doing so

will sacrifice the overall performance on the entire data set especially on POS classes with lower baseline scores (see Table 7). But since we have several expert models, if we combine them, they can complement each other's weakness and give an overall slightly better performance. The result also shows that the gain by training expert models decreases as the baseline performance on that POS class increases. For example, POS class 6 has the poorest baseline performance and the gain is 2.1%. This gain decreases gradually as we move up the table rows because the baseline performance gets better.

POS class	Expert-M3N F1	Baseline-M3N F1
1	92.5	92.2
2	87.1	86.9
3	84.9	84.7
4	85.3	84.1
5	71.8	70.4
6	57.3	55.2

Table 6: Edit detection F1 scores (%) of expert models on all words belonging to that POS class in the test set (expert-M3N column), and baseline model on all words belonging to that POS class in the test set (baseline-M3N column)

System	F1 (%)
Baseline-M3N	84.7
Expert-M3N(1)	84.6
Expert-M3N(2)	84.4
Expert-M3N(3)	84.3
Expert-M3N(4)	84.6
Expert-M3N(5)	84.0
Expert-M3N(6)	83.8

Table 7: Degradation of the overall performance by expert models compared to the baseline model

Table 8 shows the performance comparison of our baseline models and our beam-search decoder. Statistical significance tests show that our best decoder model incorporating all hypothesis evaluators gives a higher F1 score than the 3-label baseline M3N model (statistically significant at $p < 0.001$), and the 3-label baseline M3N model gives a higher F1 score than the M3N system of (Qian and Liu, 2013) (statistically significant at $p = 0.02$). Our three baseline models have about the same total number of parameters (7.6M). The BIES baseline M3N system uses the same feature templates as shown in Table 2 with reduced clique order. The 2-label baseline M3N system uses the same feature templates with the same clique order. Our results also show that joint filler and edit word prediction performs 0.4% better than edit word prediction alone. Direct combination of expert models is done by first running the general model and the expert models on each sentence to obtain all the label sequences (one for each model). Then for every word in the sentence, if its POS belongs to any one of those POS classes, we choose its label from the output of the corresponding expert model; otherwise, we choose its label from the output of the baseline model.

For the decoder, *M3N-disfluent-score* needs to be present in all cases (except when POS experts are present); otherwise, the F1 score is much worse because the entire sequence is not covered (i.e., just looking at the scores from the cleaned-up sentences is not sufficient in deciding how well filler and edit words have been removed). Adding *M3N-fluent-score*, *Fluent-LM*, or *Disfluent-LM* alone with *M3N-disfluent-score* gives about the same improvement; but when combined, higher improvement is achieved.

Similar to (Qian and Liu, 2013), our system does not make use of any external sources of information except for the last two rows in Table 8 where we added pause duration features. We found that adding pause duration features gave a small but consistent improvement in all experiments, about 0.3% absolute gain in F1 score. Our beam-search decoder (multi-threaded implementation) is about 4.5 times slower

System	F1 (%)
M3N system of (Qian and Liu, 2013)	84.1
Our baseline M3N (using BIES for E and F)	84.4
Our baseline M3N (using 2 labels: E,O)	84.3
Our baseline M3N (using 3 labels: E,F,O)	84.7
Direct combination of the 6 POS-expert-models according to each word's POS	85.2
Decoder: M3N-disfluent-score + M3N-fluent-score	85.1
Decoder: M3N-disfluent-score + Fluent-LM	85.2
Decoder: M3N-disfluent-score + Disfluent-LM	85.1
Decoder: M3N-disfluent-score + POS-experts	85.2
Decoder: M3N-disfluent-score + M3N-fluent-score + Fluent-LM + Disfluent-LM	85.6
Decoder: M3N-disfluent-score + M3N-fluent-score + Fluent-LM + Disfluent-LM + POS-experts	85.7
Decoder: M3N-disfluent-score + M3N-fluent-score + Fluent-LM + Disfluent-LM + PauseDur	85.9
Decoder: M3N-disfluent-score + M3N-fluent-score + Fluent-LM + Disfluent-LM + POS-experts + PauseDur	86.1

Table 8: Performance of the beam-search decoder with different combinations of components

than our baseline M3N model (single-threaded). Overall, it took about 0.4 seconds to detect disfluencies in one sentence with our proposed beam-search decoder approach.

To the best of our knowledge, the best published F1 score on the Switchboard Treebank corpus is 84.1% (Qian and Liu, 2013) without the use of external sources of information, and 85.7% (Zwarts and Johnson, 2011) with the use of external sources of information (large language models from additional corpora were used in (Zwarts and Johnson, 2011)). Without the use of external sources of information, our decoder approach achieves an F1 score of 85.7%, significantly higher than the best published F1 score of 84.1% of (Qian and Liu, 2013). Our decoder approach also achieves an F1 score of 86.1% after adding external sources of information (pause duration features), higher than the F1 score of 85.7% of (Zwarts and Johnson, 2011).

5.3 Discussion

We have manually analyzed the improvement of our decoder over the M3N baseline. For example, consider the sentence in Table 9. Both the baseline M3N system and the first pass output of the decoder will give the cleaned-up sentence "*are these do these programs ...*", which is still disfluent and has a relatively lower fluent LM score but a relatively higher disfluent LM score because of the erroneous n-gram "*are these do these*". The decoder makes use of the fluent LM and disfluent LM hypothesis evaluators during the beam search and performs additional passes of cleaning and eventually gives the correct output.

Sentence	*Um*	*and*	*uh*	*are*	*these*	*like*	*uh*	*do*	*these*	*programs*	...
Reference	F	F	F	E	E	E	F	O	O	O	...
M3N baseline	F	F	F	O	O	F	F	O	O	O	...
Decoder	F	F	F	E	E	E	F	O	O	O	...

Table 9: An example showing the effect of measuring the quality of the cleaned-up sentence.

Overall, our proposed decoder framework outperforms existing approaches. It also overcomes the limitations mentioned in Section 4.1. For example, hypothesis evaluators like *fluent language model score* and *M3N fluent score* achieve the purpose of measuring the quality of cleaned-up sentences. Repeatedly applying the *edit-word-marker* hypothesis producer on a sentence achieves the purpose of cleaning up

the sentence in multiple passes. Hypothesis evaluators corresponding to expert models achieve the purpose of combining POS class-specific expert models. All of these components extend the flexibility of the decoder framework in performing disfluency detection.

6 Conclusion

In conclusion, we have proposed a beam-search decoder approach for disfluency detection. Our beam-search decoder performs multiple passes of disfluency detection on cleaned-up sentences. It evaluates the quality of cleaned-up sentences and use it as a feature to rescore hypotheses. It also combines multiple expert models to deal with edit words belonging to a specific POS class. In addition, we also proposed a way (using node-weighted M3N in addition to label-weighted M3N) to train expert models each focusing on minimizing errors on words belonging to a specific POS class. Our experiments show that combining the outputs of the expert models directly according to POS tags can give rise to some improvement. Combining the expert model scores with language model scores in a weighted manner using our beam-search decoder achieves further improvement. To the best of our knowledge, our decoder has achieved the best published edit-word F1 score on the Switchboard Treebank corpus, both with and without using external sources of information.

7 Acknowledgments

This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

References

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proc. of NAACL.*

Daniel Dahlmeier and Hwee Tou Ng. 2012. A beam-search decoder for grammatical error correction. In *Proc. of EMNLP-CoNLL.*

Kallirroi Georgila. 2009. Using integer linear programming for detecting speech disfluencies. In *Proc. of NAACL.*

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy channel model of speech repairs. In *Proc. of ACL.*

Jeremy G. Kahn, Matthew Lease, Eugene Charniak, Mark Johnson, and Mari Ostendorf. 2005. Effective use of prosody in parsing conversational speech. In *Proc. of EMNLP.*

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001 Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML.*

Yang Liu, Elizabeth Shriberg, Andreas Stolcke, Dustin Hillard, Mari Ostendorf, and Mary Harper. 2006. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5).

Sameer Maskey, Bowen Zhou, and Yuqing Gao. 2006. A phrase-level machine translation approach for disfluency detection using weighted finite state transducers. In *Proc. of INTERSPEECH.*

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL.*

Xian Qian and Yang Liu. 2013. Disfluency detection using multi-step stacked learning. In *Proc. of NAACL.*

J.A.K. Suykens and J. Vandewalle. 1999 Least squares support vector machine classifiers. *Neural Processing Letters*, 9.

Ben Taskar, Carlos Guestrin, and Daphne Koller. 2003. Max-margin Markov networks. In *Proc. of NIPS.*

Pidong Wang and Hwee Tou Ng. 2013. A beam-search decoder for normalization of social media text with application to machine translation. In *Proc. of NAACL.*

Qi Zhang, Fuliang Weng, and Zhe Feng. 2006. A progressive feature selection algorithm for ultra large feature spaces. In *Proc. of ACL.*

Simon Zwarts and Mark Johnson. 2011. The impact of language models and loss functions on repair disfluency detection. In *Proc. of ACL.*

Single Document Keyphrase Extraction Using Label Information

Sumit Negi
IBM Research
Delhi, India
sumitneg@in.ibm.com

Abstract

Keyphrases have found wide ranging application in NLP and IR tasks such as document summarization, indexing, labeling, clustering and classification. In this paper we pose the problem of extracting *label specific* keyphrases from a document which has document level metadata associated with it namely *labels* or *tags* (i.e. multi-labeled document). Unlike other, supervised or unsupervised, methods for keyphrase extraction our proposed methods utilizes both the document's text and label information for the task of extracting *label specific* keyphrases. We propose two models for this purpose both of which model the problem of extracting label specific keyphrases as a random walk on the document's text graph. We evaluate and report the quality of the extracted keyphrases on a popular multi-label text corpus.

1 Introduction

The use of graphs to model and solve various problems arising in Natural Language Processing have lately become very popular. Graph theoretical methods or graph based approaches have been successfully applied for a varied set of NLP tasks such as Word Sense Disambiguation, Text Summarization, Topic detection etc. One of the earliest and most prominent work in this area has been the TextRank (Mihalcea and Tarau, 2004) method - an unsupervised graph-based ranking model for extracting keyphrases and "*key*" sentences from natural language text. This unsupervised method extracts prominent terms, phrases and sentences from text. The TextRank models the text as a graph where, depending on the end application, text units of various sizes and characteristics can be added as vertices e.g. open class words, collocations, sentences etc. Similarly, based on the application, connections can be drawn between these vertices e.g. lexical or semantic relation, contextual overlap etc. To identify "central" or "key" text units in this text graph, TextRank runs the *PageRank* algorithm on this constructed graph. The ranking over vertices (text units), which indicates their centrality and importance, is obtained by finding the stationary distribution of the random walk on the text graph.

In this paper, we consider the problem of extracting *label specific* keyphrases from a document which has document level metadata associated with it namely *labels* (i.e. multi-labeled document). To elaborate, consider a document as shown in Figure 1. This document has been assigned to two categories as indicated by the labels "*Air Pollution*" and "*Plant Physiology*". Running TextRank on this article yields top ranked key-phrases such as "*calibrated instrument*", "*polluting gases*", "*industrial development*" etc. These keyphrases, though central to the article, are not specific to any of the *labels* that have been assigned to the article. For instance, one would associate keyphrases such as "*carbon monoxide* ", "*air pollutants*" to be more relevant to the "*Air Pollution*" label and keyphrases such as "*stomatal movement*", "*cell defense*" to be more closely associated with the "*Plant Physiology*" label. The objective of this paper is to explore extensions to TextRank for extracting label-specific keyphrases from a multi-labeled document. Such label-specific keyphrases can be useful for a number of practical applications namely: highlighting such terms within the body of a document could provide a label-specific (topic-focussed) view of the document thus facilitating fast browsing and reading of the document, such key terms could also be useful for generating topic-driven or label-specific summaries and in multifaceted search.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1468–1476, Dublin, Ireland, August 23-29 2014.

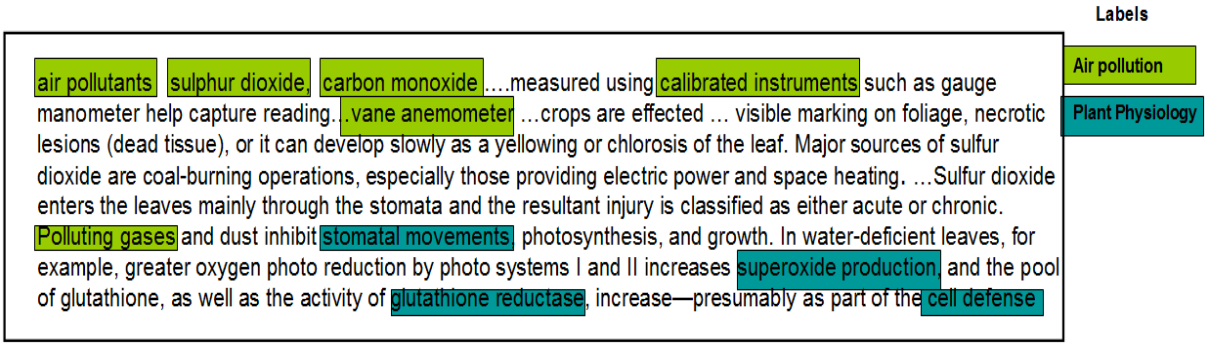


Figure 1: Label specific keyphrases (best viewed in color). Note that there could be keyphrases that are common to both labels. Due to space restrictions only a snippet of the document is shown.

The rest of the paper is organized as following. We discuss related work and provide an overview of our approach in Section 2. Details of the proposed method is discussed in Section 3 followed by evaluation in Section 4. Future work and conclusion is presented in Section 5.

2 Related Work

The methods for keyphrase (or keyword) extraction can be roughly categorized into either *unsupervised* or *supervised*. *Unsupervised methods* usually involve assigning a saliency score to each candidate phrase by considering various features. Popular work in this area include the use of point-wise KL-divergence between multiple language models for scoring both *phrase-ness* and *informativeness* of candidate phrases (Tomokiyo and Hurst, 2003), use of TF-IDF weighting (A. Hulth, 2003) etc. *Supervised machine learning algorithms* have been proposed to classify a candidate phrase into either keyphrase or not using features such as the frequency of occurrence, POS information, and location of the phrase in the document. All the above methods only make use the document text for generating keyphrases and cannot be used (as-is) for generating label-specific keyphrases.

One possible method for extracting label-specific keyphrases from a document could be based on post-processing the output of the TextRank algorithm in the following way (1) Identify a set of *label specific features* f_l^{cand} (unigram terms) that are strongly correlated with the *label*. This could be done by applying feature selection methods (Forman, 2003), (Forman, 2003) on a multi-label text corpus (we discuss this step in more detail in a later section). For instance, $f_{air\_pollution}^{cand}$={"*pollutant*","*gases*",...} (2) Run the TextRank algorithm on the document d to generate a list of keyphrases $keyphrase_d$ (3) *Filter* the resultant list $keyphrase_d$ based on lexical or semantic match with the label specific features f_l^{cand} to generate $keyphrase_d^l$ or label-l specific keyphrase for document d.

This approach suffers from the following limitations (a) The keyphrase list generated in Step (2) i.e. $keyphrase_d$ might be dominated by keyphrases which have little to do with label l. Post processing this list (Step 3) using f_l^{cand} might result in only very few keyphrases in $keyphrase_d^l$. (b) The label specific features f_l^{cand}, which are derived from corpus level statistics[1], might not be the best indicator of the *keyphrase-ness* of a term in the document. (c) Moreover, consider a scenario where a document is associated with more than one label. Consider the previous example where the document is associated with two labels "*Air Pollution*" and "*Plant Physiology*". When extracting keyphrases specific to the label/category "*Air Pollution*" from document d one would expect that the extracted keyphrases are *closer* to the *Air Pollution* label/category and *distant* from other labels associated with document d i.e. "*Plant Physiology*". It is not evident how this can be modeled in this approach. In this paper we propose an approach that models the problem of finding label-specific keyphrases in a document as a random walk on the document's text-graph. Two approaches are proposed namely *PTR: Personalized TextRank* and *TRDMS: TextRank using Ranking on Data Manifolds with Sinks*.

[1]Using feature selection methods

PTR: Personalized TextRank : In this setting the PageRank algorithm, which is the underpinning of the *TextRank* keyphrase extraction algorithm, is replaced with the personalized page rank (Haveliwala, 2002) algorithm. By using the label specific features f_l^{cand} as the *personalization vector* we are able to bias the walk on the underlying text graph towards terms relevant to the label. We discuss this approach in more detail in Section 3.3. Even though using a label specific *transport* or *personalization vector* helps bias the walk towards terms specific to that label, terms relevant to labels other than l continue to influence the walk. The *Personalized TextRank* method offers no elegant solution which would penalize terms unrelated to l while simultaneously preferring terms relevant to label l.

To achieve both these goals in one model we propose the *TRDMS: TextRank using Ranking on Data Manifolds with Sinks* approach. We model the problem of identifying label specific keyphrases in a given document as a random walk over the document's *weighted* text graph with *sink* and *query* nodes[2]. Ranking on data manifolds was first proposed by (Zhou et al., 2004) and has been used for multi-document summarization (Wan et al., 2007), image retrieval (He et al., 2004) etc. An intuitive description of the ranking algorithm is described as follows. A weighted network is constructed first, where nodes represent all the data and query points, and an edge is put between two nodes if they are "*close*". *Query* nodes are then initialized with a positive ranking score, while the nodes to be ranked are assigned a zero initial score. All the nodes, except the *sink* nodes, then propagate their ranking scores to their neighbor via the weighted network. The propagation process is repeated until a global state is achieved, and all the nodes except the query nodes are ranked according to their final scores. Manifold ranking gives high rank to nodes that are close to the query nodes on the manifold (*relevance*) and that have strong centrality (*importance*). Sink nodes, whose ranking is fixed to the minimum (zero) during the ranking process, do not spread any ranking score to their neighbors thus penalizing the nodes that are connected to them. To use this method for extracting label-(l) specific keyphrases , f_l^{cand} are modeled as *query* nodes while features associated with labels other than l are modeled as *sink* nodes. This approach is inspired by the work done by (Cheng et al., 2011) for query recommendation and update summarization. Section 3.4 discusses this method in more detail. To summarize, to the best of our knowledge we are the first to propose the problem of extracting *label* specific keyphrases from a multi-labeled document. Our modifications to TextRank for achieving this task are novel. Moreover, our idea of of using *Ranking on Data Manifolds* on the document-level text graphs for extracting label specific keyphrases is a new contribution.

3 Generating Label Specific Keyphrases

3.1 Notation

In this section we introduce notations which we use throughout the paper. Let D represent a multi-label document corpus and $\Im$ be the set of all possible labels which could be associated with documents in D. A document from this corpus is denoted by d and the set of labels associated with document d is denoted by ℓ, where $d \in D$ and $\ell \subseteq \Im$. The text graph for document d is denoted by G_d and M denotes the number of vertices in G_d . We describe how this text graph is constructed in Section 3.2. Features specific to label l, which are extracted from the corpus D, are represented as f_l^{cand}, where $l \in \Im$. Section 3.5 describes how these *label specific features* are extracted from a multi-label document corpus.

3.2 Building the Text Graph

For a given document d the text graph G_d is built in the following way. All open-class, unigram tokens occurring in d are treated as vertices. Two vertices are connected if their corresponding lexical units co-occur within a window of maximum N words, where N is set to 10 for all our experiments. As indicated by (Mihalcea and Tarau, 2004) co-occurrence links express relations between syntactic elements and represent cohesion indicators for a given text. Note that the methods described in Section 3.3 and Section 3.4 provide a score/rank for each vertex (unigram term) in the graph. To generate keyphrases (n-grams) from these candidate terms the following post-processing is performed on the top ranked terms. Vertices are sorted in reverse order of their score and the top K vertices in the ranking are retained

[2]Nodes correspond to terms in a text graph

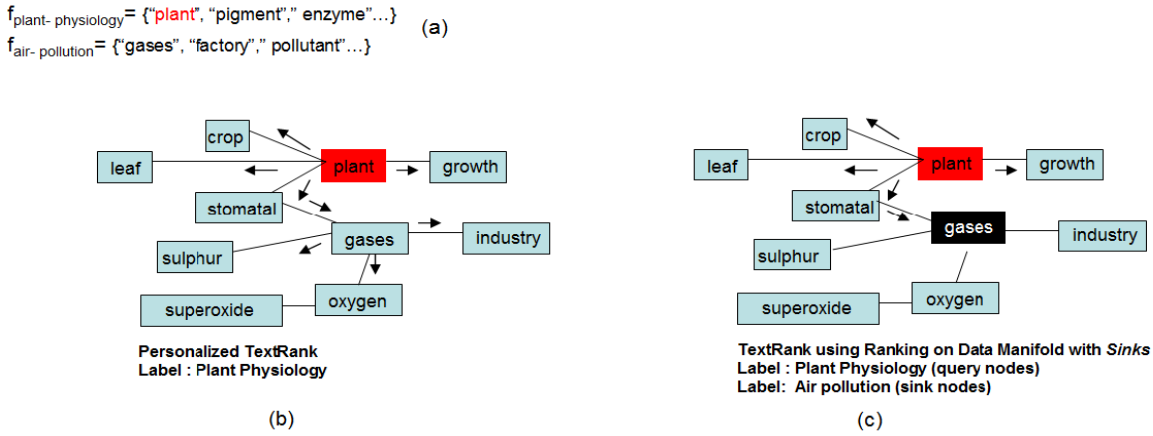


Figure 2: (a) Label specific features f_l^{cand} (b) *Personalized TextRank* - walk biased towards terms related to $f_{plant\_physiology}^{cand}$ (shown in red color). (c) *TextRank using Ranking on Data Manifold with Sinks*: walk biased towards terms related to $f_{plant\_physiology}^{cand}$, while simultaneously penalizing terms that are related to $f_{air\_pollution}^{cand}$. The sink points, which are shown in black color, are vertices whose ranking scores are fixed at the minimum score (zero in our case) during the ranking process. Hence, the sink points will never spread any ranking score to their neighbors. Arrows indicate diffusion of ranking scores (Figure best viewed in color)

for post-processing. Let this ranked list be represented as <T_K>. During post-processing, all terms selected as potential keywords are marked in the text, and sequence of adjacent keywords are collapsed into a multi-word keyphrase. For example, in the text *calibrated instruments are used to measure*, if the unigram terms *calibrated* and *instruments* are selected as potential/candidate terms by the PTR or TRDMS method, since they are adjacent they are collapsed into one single keyphrase "*calibrated instruments*". This heuristic is implemented as a function which is referred as $kphrase_{gen}$(<T_K>,d). This function takes as input the ranked term list <T_K> and the document text d and returns the collapsed set of keyphrases. A similar approach was adopted in the *TextRank* (Mihalcea and Tarau, 2004) work.

3.3 *PTR*: Personalized TextRank

For extracting label-l specific keyphrases from document d we modify the TextRank (Mihalcea and Tarau, 2004) algorithm. We replace the *PageRank* algorithm used in the *TextRank* method with the *Personalized Page Rank* (Haveliwala, 2002) algorithm. PageRank gives a stationary distribution of a random walk which, at each step, with a certain probability ϵ jumps to a random node, and with probability 1-ϵ follows a randomly chosen outgoing edge from the current node. More formally, let G_d denotes the text graph of document d with M vertices where d_i denotes the out degree of node w_i, then $p = \epsilon Lp + (1\text{-}\epsilon)v$. Where p is the page rank vector, L is a $M \times M$ transition probability matrix with L_{ji}=$\frac{1}{d_i}$. In the page rank equation v is a stochastic normalized vector whose element values are all $\frac{1}{M}$. This assigns equal probabilities to all nodes in the graph in case of random jumps. In the *personalized* page rank formulation the vector v can be non-uniform and can assign stronger probabilities to certain kind of nodes effectively biasing the PageRank vector. In the *PTR* approach v is modeled to capture the evidence that is available for label l in document d. Doing so biases the walk towards terms that are more specific to label l in the document. This is achieved by considering vertices (terms) that are common between the label l feature vector i.e. f_l^{cand} and the text graph for document d i.e. G_d. More precisely, for a label l associated with a document d, let V_d^l denote the intersection of the set V_d with f_l^{cand}, i.e. $V_d^l = V_d \cap f_l^{cand}$, where V_d denote the vertex set for the text graph G_d[3] and $l \in \ell$. In this way V_d^l indicates the *evidence* we have for label l in the text graph G_d. To illustrate this point consider Figure 2. The label specific features for label $Plant\ Physiology$ is shown in Figure 2 (a) denoted as $f_{plant-physiology}^{cand}$. The term colored in red

[3] G_d is the text graph built for document d using the method outlined in Section 3.2.

indicates the term that is common between $f^{cand}_{plant-physiology}$ and G_d i.e. $V^{plant-physiology}_d$

Having identified the nodes (V^l_d) which should be allocated stronger probabilities in v the next step is to devise a mechanism to determine these probabilities. We experiment with four approaches. In the first approach, referred to as $seed\_nodes\_only$, we allocate all the probability mass in v uniformly to the nodes in V^l_d, all other nodes i.e. nodes $\notin V^l_d$ are assigned zero probability. In the second approach, referred to as the $seed\_and\_eta$ approach, we keep aside a small fraction η of the probability mass, which is distributed uniformly to all the nodes $\notin V^l_d$, the rest of the probability mass i.e. 1-η[4] is uniformly distributed to all nodes $\in V^l_d$. The third approach, referred to as $non\_uniform\_seed\_only$, is similar to the $seed\_nodes\_only$ approach except that in this case the probability mass in v is not allocated uniformly to the nodes in V^l_d. Probability mass is allocated to the nodes in proportion to their importance, as indicated by the weights allocated to the feature in f^{cand}_l by the feature selection method used. As we discuss in Section 3.5 the feature selection methods, which are used for generating label specific feature f^{cand}_l, compute weights for individual features in f^{cand}_l. These weights (e.g mutual information score, t-score) indicate the strength of association between the feature and the label. In the $non\_uniform\_seed\_only$ approach we allocate probability mass to nodes in V^l_d in proportion to their *feature weights*. Finally, in the $non\_uniform\_eta$ approach we distribute the probability mass i.e. 1-η amongst the V^l_d in proportion to their *feature weights*. The left probability mass of η is distributed uniformly amongst other nodes $\notin V^l_d$. Performance of these different configurations are evaluated in Section 4.1.

One shortcoming of the *PTR* approach is that it does not provides a clean mechanism to integrate features from labels other than l which are associated with the document d. The motivation of doing so is to on one hand bias the walk on the text graph towards terms in f^{cand}_l while simultaneously penalizing terms which are in F_{cand} = $\cup_{k \neq l \text{ and } k \in \ell} f^{cand}_k$[5]. As shown in Figure 2 (b) not incorporating this information results in a leakage of scores (indicated using arrows) to nodes not relevant to label l (e.g. *gases*, *sulphur* etc) . In the next section we describe the ***TRDMS*** or ***TextRank using Ranking on Data Manifold with Sinks*** approach which allows us to simultaneously consider both f^{cand}_l and F_{cand} in the same model.

3.4 *TRDMS*: TextRank using Ranking on Data Manifold with Sinks

Algorithm 1: Algorithm for generating label-l specific keyphrases for document d

Data: Document d, label-l specific unigram features f^{cand}_l, unigram features for label categories other than l represented as F_{cand} = $\cup_{k \neq l and\ k \in \ell} f^{cand}_k$
Result: label-l specific keyphrases from document d
1. Build a Text Graph G_d for document d as discussed in Section 3.2. Let w_i indicate the vertices in G_d ;
2. Construct an *affinity matrix* A, where A_{ij} = *sim*(w_i,w_j) if there is an edge linking w_i, w_j in G_d. *sim*(w_i,w_j) indicates similarity between vertices w_i, w_j ;
3. Symmetrically, normalize A as S = $D^{-1/2}AD^{-1/2}$. D is a diagonal matrix matrix with its (*i*,*i*)-element equal to the sum of the *i*-th row A;
4. **while** *(!*converge*(p))* **do**
   Iterate $p(t+1)$ = $\alpha SIp(t)$ + (1-α)y ;
   /* where 0 <α <1 and I is an indicator diagonal matrix with it's (i,i)-element equal to 0 if $w_i \in V^{\neg l}_d$ and 1 otherwise.*/
end
5. Sort the vertices $w_q \in V_q$ in descending order of their scores $p[q]$. Let this ranked list be represented as <T_K>;
6. $kphrase^l_d$ = $kphrase_{gen}$(<T_K>,d) , where $kphrase^l_d$ is the label-l specific keyphrase list for document d;
7. return $kphrase^l_d$;

In this section we describe the *TextRank using Ranking on Data Manifold with Sinks* approach that allows us to simultaneously consider both f^{cand}_l and F_{cand} when extracting label l specific keyphrases from document's d text graph. For ease of exposition we repeat a few notations and introduce some new ones. Let V_d denote the vertex set for the text graph G_d. Vertices for the text graph G_d are represented by w_i where $i \in$ [1..M], M is the number of vertices i.e. M=$|V_d|$. As introduce earlier, V^l_d denotes the

[4]Please note v is a stochastic normalized vector whose elements sum to 1. In our experiments we set η=0.2
[5]Where ℓ indicates the label set associated with document d

intersection of the set V_d with f_l^{cand}, i.e. $V_d^l = V_d \cap f_l^{cand}$. V_d^l indicates the *evidence* we have for label l in the text graph G_d, where $l \in \ell$. These vertices are also referred to as *query nodes* in the ranking on data manifold literature. Let $V_d^{\neg l}$ denote the intersection of the set V_d with F_{cand}, where $F_{cand} = \cup_{k \neq l \text{ and } k \in \ell} f_k^{cand}$ i.e. all the unigram features associated with label categories other than l[6]. These vertices are also referred to as *sink nodes* in the ranking on data manifold literature. All other vertices are indicated by V_d^q, where $V_d^q = V_d \setminus (V_d^{\neg l} \cup V_d^l)$ denote the set of points to be ranked. Let $p{:}V \rightarrow \Re$ denote the ranking function which assigns a ranking score p_i to each vertex w_i in G_d. One can view p as a vector i.e. $p = [p_1,....,p_M]$. A binary vector $y = [y_1,....,y_M]$ is defined in which $y_i = 1$ if $w_i \in V_d^l$ otherwise $y_i = 0$.

Algorithm 1 gives a detailed outline of the *TRDMS* method. This algorithm is based on the algorithm proposed by (Cheng et al., 2011) for ranking on data manifold with sink points. To generate label-l specific keyphrase for document d the algorithm considers document d, label-l specific unigram features f_l^{cand}, and unigram features for labels other than l represented as F_{cand}. It begins by first building a text graph G_d. After this an affinity matrix A is constructed. This is shown in Step 2. The affinity matrix A, which captures the similarity between vertices (terms in the text graph) w_i and w_j, is built using WordNet. We use the popular WordNet::Similarity (Pedersen et al., 2004) package which measures the semantic similarity and relatedness between a pairs of concepts. After symmetrically normalizing A (Step 3) and initializing the *query* and *sink* nodes the scores are propagated till convergence (Step 4). The routine *converge(p)* checks for convergence by comparing the value of p between two consecutive iterations. If there is little or no change in p the routine return *true*. To generate n-gram keyphrases we follow the approach described in Section 3.2. In Step 6 of Algorithm 1 the $kphrase_{gen}$[7] routine is invoked. In order to choose top-k, label-l specific keyphrases for document d one can select the first k elements of the $kphrase_d^l$ list.

3.5 Generating label specific features from a multi-label corpus

As discussed in previous sections the label specific features f_l^{cand} play an important role in the overall ranking process. When searching for label-l specific keyphrases, the unigram features f_l^{cand} helps bias the walk on the document's text graph towards terms that are relevant and central to label l. We also saw that by considering F_{cand} i.e. unigram features belonging to label categories other than l[8] as *sink* nodes prevents *leakage* of the ranking score to terms not relevant or central to l. We show through experiments in Section 4 that this improves the quality of label-l specific keyphrases extracted from document d. In order to generate *label specific features* from a multi-label corpus D we adopt the *problem transformation* approach commonly used in multi-label learning. In this approach the multi-label corpus D is transformed into $\mid \Im \mid$ single-label data sets, where $\Im$ is the set of labels associated with corpus D. Post this transformation any single-*label* feature selection method can be used to extract label l specific features from these single-label data sets. For our setup we experiment with unigram features selected using mutual information and chi-squared based feature selection methods.

4 Experiment

In order to assess the quality of the label-specific keyphrases generated by our system we conduct a manual evaluation of the generated output. Details of this evaluation are provided in Section 4.1. For our experiments we use a subset of the multi-label corpus EUR-Lex[9]. The EUR-Lex text collection is a collection of documents about European Union law. It contains many different types of documents, including treaties, legislation, case-law and legislative proposals, which are labeled with EUROVOC descriptors. A document in this data-set could be associated with multiple EUROVOC descriptors[10]. The data set that was downloaded contained 16k documents documents and 3,993 EUROVOC descriptors.

[6] We do not assume that $f_l^{cand} \cap F_{cand} = \emptyset$

[7] Details of this routine are provided in Section 3.2

[8] In cases where the document is associated with more than one label or category

[9] http://www.ke.tu-darmstadt.de/resources/eurlex

[10] We treat these as labels

Method	*Precision*$^{\text{avg}}$	*Recall*$^{\text{avg}}$	*F-measure*$^{\text{avg}}$
TPP_{baseline}	0.163	0.194	0.177
$PTR_{\text{seed\_nodes\_only}}$	0.169	0.213	0.188
$PTR_{\text{seed\_and\_eta}}$	0.199	0.223	0.210
$PTR_{\text{non\_uniform\_seed\_only}}$	0.203	0.231	0.216
$PTR_{\text{non\_uniform\_eta}}$	0.237	0.257	0.247
TRDMS	0.397	0.387	0.392

Table 1: Keyphrase Extraction Results

We removed labels that were under represented[11] in this data set. We refer to this data set as the $EUR-Lex_{filtered}$ data set. We randomly selected 100 documents from the $EUR-Lex_{filtered}$ data set. Two criteria were considered when selecting these documents (a) Each document should be associated with at least 2 but not more than 3 labels (b) The size of the *evidence* set i.e. $|V_d^l|$ where $V_d^l = V_d \cap f_{cand}^l$ is at least 10% of $|V_d|$, where V_d represents the vertex set of the text graph associated with d. The resulting data set is referred to as the $EUR-Lex_{filtered}^{keyphrase}$ data set. The reason for enforcing these two criteria is the following. Ensuring that a document in $EUR-Lex_{filtered}^{keyphrase}$ has at least 2 labels allows us to experiment with *sink nodes* i.e. F_{cand}. As we discuss in Section 4.1 for each *label* associated with a document, a human evaluator was asked to generates a label specific list of keyphrases. For example, if a document is associated with 3 labels, three label specific keyphrase list had to be generated by the human evaluator. Allowing documents with more than 3 labels makes this process tedious. The reason for putting restriction (b) when building the $EUR-Lex_{filtered}^{keyphrase}$ is explained in Section 4.1.1. For generating label-l specific features we use the approach described in Section 3.5. For our experiments mutual information based feature selection method was used with a feature size of 250 i.e. $|f_l^{cand}| = 250$.

4.1 Label-specific Keyphrase Evaluation

Two graduate students were asked to manually extract label-specific keyphrases for each document in the $EUR-Lex_{filtered}^{keyphrase}$ data set. At most 10 keyphrases could be assigned to each document-label pair. This results in a total of 1721 keyphrases. The Kappa statistics for measuring inter-agreement among the annotation was 0.81. Any annotation conflicts between the two subjects was resolved by a third graduate student. For evaluation, the automatically extracted label-specific keyphrases for a given document were compared with the manually extracted/annotated keyphrases. Before comparing the keyphrase, the words in the keyphrase were converted to their corresponding base form using word stemming. We calculate three evaluation metrics namely Precision, Recall and F-measure for each document-label pair. Precision (P) = $\frac{count_{correct}}{count_{system}}$, Recall (R) = $\frac{count_{correct}}{count_{human}}$ and F-measure (F) = $\frac{2PR}{P+R}$, where $count_{correct}$ is the total number of correct keyphrases extracted by our method, $count_{system}$ is the total number of automatically extracted keyphrases and $count_{human}$ is the total number of keyphrases labeled by the human annotators. These metrics are calculated for each document-label pair in the $EUR-Lex_{filtered}^{keyphrase}$ data set and then averaged to obtain $Precision^{avg}$, $Recall^{avg}$ and $F-measure^{avg}$. These results are shown in Table 1

We compare the performance of our system against the *TextRank with Post-Processing:* $TPP_{baseline}$ baseline which was explained in Section 2. Briefly, in this setup to identify label-l specific keyphrases in document d, we run *TextRank* on document d and filter the generated keyphrase list based on f_l^{cand} i.e. label l specific features. In all setups the document text graph is built in the same fashion i.e. N = 10 and co-occurrence relationship is used to draw edges between nodes in the text graph. For generating the affinity matrix A, which is used in the *TRDMS* method, the *res* semantic similarity method is used[12]. To reiterate, when generating label-l specific keyphrases for document d the *PTR* method only uses f_l^{cand}, whereas the *TRDMS* method uses both f_l^{cand} (as *query* nodes) and $F_{cand} = \cup_{k \neq l \text{ and } k \in \ell} f_k^{cand}$ (where

[11] Any label which occurred less than 10% times in the data set was removed. The documents associated with these labels were also removed from the data set

[12] We experimented with other semantic similarity measures such as *lin* and *jcn*. The *res* measure gave us the best results

ℓ is the set of labels associated with document d) i.e. all the unigram features associated with label categories other than l (as *sink* nodes). One can observe from Table 1 that for *PTR* the *non_uniform_eta* configuration gives the best result. Overall the *TRDMS* approach significantly outperforms all *PTR* configurations and our baseline. This validates our belief that one can significantly improve the quality of extracted keyphrase by not only considering label-l specific features i.e. f_l^{cand} but also features associated with label categories other than l. When we analyzed the performance of *TRDMS* at the document level we observed that the keyphrase extraction metrics for documents which had *strongly correlated labels* e.g. "*tariff_quota*" and "*import_license*" was 9-11% lower than the reported average scores. On the contrary, keyphrase extraction metrics for documents which had labels that had no or weak correlation e.g. "*aid_contract*" and "*import_license*" was 3-5% higher than the reported average scores. One reason for this could be the substantial overlap between f_l^{cand} and F_{cand} for highly correlated labels. This large overlap results in the *query* nodes being considered as *sink* nodes which negatively impacts the score propagation in the underlying text graph.

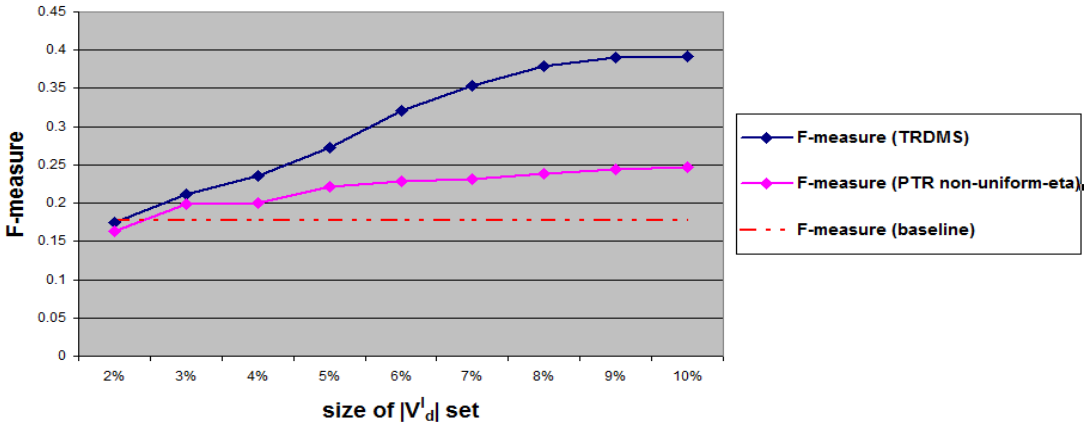


Figure 3: Impact of evidence set size on F-measure (best viewed in color)

4.1.1 Impact of evidence set size ($|V_d^l|$) on keyphrase generation results

To recap, elements in set V_d^l indicate the *evidence* we have for label l in the text graph of document d i.e. G_d. In order to investigate how the size of the evidence set i.e. $|V_d^l|$ impacts the performance of our system the following simulation was carried out. In different setups we randomly drop out elements from V_d^l so that the size of the resulting evidence set ranges from 2% to 10% of $|V_d|$, where $|V_d|$ represents the vertex set size of text graph G_d. We plot the impact this has on the F-measure in Figure 3. One observes that when the *evidence* set size is in the range 2-4% the gains over the $TPP_{baseline}$ baseline (0.177) are low to modest. As the evidence set size increases the gains over the baseline increases substantially.

5 Conclusion and Future Work

In this paper we presented the problem of extracting label specific keyphrases from a document. We pose the problem of extracting such keyphrases from a document as a random walk on a document's text graph. The methods proposed in this paper utilizes the *label specific features*, which are strongly associated with the label, to bias the walk towards terms that are more relevant to the label. We show through experiments that when generating label-l specific keyphrases it helps to consider both label-l specific features and features associated with labels other than l. As future work we would like to further assess the quality of the generated keyphrases by using these keyphrases for generating topic (or label) focused document summaries.

References

Takashi Tomokiyo and Matthew Hurst. 2003. *A language model approach to keyphrase extraction*. Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment.

A. Hulth. 2003. *Improved Automatic Keyword Extraction Given More Linguistic Knowledge*. Proceedings of the Conference on Empirical Methods in Natural Language Processing.

Yutaka Matsuo and Mitsuru Ishizuka. 2004. *Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information*. International Journal on Artificial Intelligence Tools.

Peter D. Turney. 2000. *Learning Algorithms for Keyphrase Extraction*. Information Retrieval.

Eibe Frank and W. Gordon Paynter and Ian H. Witten and Carl Gutwin and Craig G. Nevill-Manning. 1999. *Domain-Specific Keyphrase Extraction*. IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence.

Peter D. Turney. 2003. *Coherent Keyphrase Extraction via Web Mining*. IJCAI '03: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence.

Min Song and Il-Yeol Song and Xiaohua Hu. 2003. *KPSpotter: a flexible information gain-based keyphrase extraction system*. Fifth International Workshop on Web Information and Data Management.

Olena Medelyan and Ian H. Witten. 2006. *Thesaurus based automatic keyphrase indexing*. JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries.

George Forman. 2003. *An Extensive Empirical Study of Feature Selection Metrics for Text Classification*. The Journal of Machine Learning Research.

George Forman. 2004. *A pitfall and solution in multi-class feature selection for text classification*. International Conference on Machine Learning.

Rada Mihalcea and Paul Tarau. 2004. *TextRank: Bringing Order into Texts*. Proceedings of EMNLP-04 and the 2004 Conference on Empirical Methods in Natural Language Processing.

Rada Mihalcea, Paul Tarau and Elizabeth Figa. 2004. *PageRank on Semantic Networks, with Application to Word Sense Disambiguation*. COLING.

Rada Mihalcea, Paul Tarau and Elizabeth Figa. 2004. *PageRank on Semantic Networks, with Application to Word Sense Disambiguation*. COLING.

Dengyong Zhou and Jason Weston and Arthur Gretton and Olivier Bousquet and Bernhard SchÃűlkopf. 2004. *Ranking on Data Manifolds*. Advances in Neural Information Processing Systems.

Ted Pedersen and Siddharth Patwardhan and Jason Michelizzi. 2004. *WordNet::Similarity: Measuring the Relatedness of Concepts*. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2005. *A language independent algorithm for single and multiple document summarization*. In Proceedings of IJCNLPâĂŹ2005.

Xiaojun Wan. 2007. *TimedTextRank: adding the temporal dimension to multi-document summarization*. SIGIR.

Taher H. Haveliwala. 2002. *Topic-sensitive PageRank*. Proceedings of the Eleventh International World Wide Web Conference.

Xue-Qi Cheng and Pan Du and Jiafeng Guo and Xiaofei Zhu and Yixin Chen. 2011. *Ranking on Data Manifold with Sink Points*. IEEE Transactions on Knowledge and Data Engineering.

Jingrui He and Mingjing Li and Hong-Jiang Zhang and Hanghang Tong and Changshu Zhang. 2004. *Manifold-ranking Based Image Retrieval*. Proceedings of the 12th Annual ACM International Conference on Multimedia.

Xiaojun Wan and Jianwu Yang and Jianguo Xiao. 2007. *Manifold-Ranking Based Topic-Focused Multi-Document Summarization*. IJCAI.

Predicting Interesting Things in Text

Michael Gamon
Microsoft Corp.
One Microsoft Way
Redmond, WA 98052
mgamon@microsoft.com

Arjun Mukherjee
Department of Computer Science
University of Houston
Houston, TX 77004
arjun4787@gmail.com

Patrick Pantel
Microsoft Corp.
One Microsoft Way
Redmond, WA 98052
ppantel@microsoft.com

Abstract

While reading a document, a user may encounter concepts, entities, and topics that she is interested in exploring more. We propose models of "interestingness", which aim to predict the level of interest a user has in the various text spans in a document. We obtain naturally occurring interest signals by observing user browsing behavior in clicks from one page to another. We cast the problem of predicting interestingness as a discriminative learning problem over this data. We leverage features from two principal sources: textual context features and topic features that assess the semantics of the document transition. We learn our topic features without supervision via probabilistic inference over a graphical model that captures the latent joint topic space of the documents in the transition. We train and test our models on millions of real-world transitions between Wikipedia documents as observed from web browser session logs. On the task of predicting which spans are of most interest to users, we show significant improvement over various baselines and highlight the value of our latent semantic model.

1 Introduction

Reading inevitably leads people to discover interesting concepts, entities, and topics. Predicting what interests a user while reading a document has important applications ranging from augmenting the document with supplementary information, to ad placement, to content recommendation. We define the task of predicting **interesting things (ITs)** as ranking text spans in an unstructured document according to whether a user would want to know more about them. This desire to learn more serves as our proxy for interestingness.

There are many types of observable behavior that indicate user interest in a text span. The closest one to our problem definition is found in web browsing, where users click from one document to another via named anchors. The click process is generally governed by the user's interest (modulo erroneous clicks). As such, the anchor name can be viewed as a text span of interest for that user. Furthermore, the frequency with which users, in aggregate, click on an anchor serves as a good proxy for the level of interest[1].

What is perceived as *interesting* is influenced by many factors. The semantics of the document and candidate IT are important. For example, we find that when users read an article about a movie, they are more likely to browse to an article about an actor or character than to another movie or the director. Also, user profile and geo-temporal information are relevant. For example, interests can differ depending on the cultural and socio-economic background of a user as well as the time of the session (e.g., weekday versus weekend, daytime versus late night, etc.).

[1] Other naturally occurring expressions of user interest, albeit less fitting to our problem, are found in web search queries, social media engagement, and others.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1477–1488, Dublin, Ireland, August 23-29 2014.

Strictly speaking, human interestingness is a psychological and cognitive process (Varela et al., 1991). Clicks and long dwell times are salient observed behavioral signals of interestingness that have been well accepted in the information retrieval literature (Claypool et al., 2001; Mueller and Lockerd, 2001). In this paper, we utilize the observed user's browsing behavior as a supervision signal for modeling interestingness. Specifically, we cast the prediction of ITs as a discriminative learning task. We use a regression model to predict the likelihood of an anchor in a Wikipedia article to be clicked, which as we have seen above can serve as a proxy for interestingness. Based on an empirical study of a sample of our data, we use features in our model from the document context (such as the position of the anchor text, frequency of the anchor text in the current paragraph, etc.) as well as semantic features that aim to capture the latent topic space of the documents in the browsing transition. These semantic features are obtained in an unsupervised manner via a joint topic model of source and target documents in browsing transitions. We show empirical evidence that our discriminative model is effective in predicting ITs and we demonstrate that the automatically learned latent semantic features contribute significantly to the model performance. The main contributions of this paper are:

- We introduce the task of predicting interesting things as identifying what a user likely wants to learn more about while reading a document.
- We use browsing transitions as a proxy for interestingness and model our task using a discriminative training approach.
- We propose a semantic probabilistic model of interestingness, which captures the latent aspects that drive a user to be interested in browsing from one document to another. Features derived from this semantic model are used in our discriminative learner.
- We show empirical evidence of the effectiveness of our model on an application scenario.

2 Related Work

Salience: A notion that might at first glance be confused with interestingness is that of salience (Paranjpe 2009; Gamon et al. 2013). Salience can be described as the centrality of a term to the content of a document. Put another way, it represents what the document is about. Though salience and interestingness can interact, There are clear differences. For example, in a news article about President Obama's visit to Seattle, Obama is salient, yet the average user would probably not be interested in learning more about Obama while reading that article.

Click Prediction: Click prediction models are used pervasively by search engines. Query based click prediction aims at computing the probability that a given document in a search-result page is clicked on after a user enters some query (Joachims, 2002; Joachims et al., 2005; Agichtein et al., 2006; Guo et al., 2009a). Click prediction for online advertising is a core signal for estimating the relevance of an ad to a search result page or a document (Chatterjee et al., 2003; Broder et al., 2007; Craswell et al., 2008; Graepel et al., 2010). Also related are personalized click models, e.g., (Shen et al., 2012), which use user-specific click through rate (CTR). Although these applications and our task share the use of CTR as a supervision signal, there is a key difference: Whereas in web search CTR is used as a predictor/feature at runtime, our task specifically aims at predicting interestingness in the absence of web usage features: Our input is completely unstructured and there is no assumption of prior user interaction data.

Use of probabilistic models: Our semantic model is built over LDA (Blei et al., 2003) and has resemblances to Link-LDA models (Erosheva et al., 2004) and Comment-LDA models (Yano et al., 2009). However, these are tailored for blogs and associated comment discussions which is very different from our source to destination browsing transition logs. Guo et al., (2009b) used probabilistic models for discovering entity classes from query logs and in (Lin et al., 2012), latent intents in entity centric search were explored. Gao et al. (2011) employ statistical machine translation to connect two types of content, learning semantic translation of queries to document titles. None of the above models, however, are directly applicable to the joint topic mappings that are involved in source to destination browsing transitions which are the focus of our work.

Predicting Popular Content: Modeling interestingness is also related to predicting popular content in the Web and content recommenders (Lerman and Hogg, 2010; Szabo and Huberman, 2010; Bandari et al., 2012). In contrast to these tasks, we strive to predict what term a user is likely to be interested in when reading content. We do not rely on prior browsing history, since we aim to predict interestingness

in unstructured text with no interaction history. We show in our experiments that a popularity signal alone is not a sufficient predictor for interestingness.

3 The Interestingness Task

The process of identifying interesting things (ITs) on a page consists of two parts: (1) generating candidate things (e.g., entities, concepts, topics); and (2) scoring and ranking these according to interestingness. In this paper, we fix step 1 and focus our effort on step 2, i.e., the assignment of an interestingness score to a candidate. We believe that this scope is appropriate in order to understand the factors that enter into what is perceived as interesting by a user. Once we have gained an understanding of the interestingness scoring problem, however, there are opportunities in identifying candidates automatically, which we leave for future work.

In this section we begin by formally defining our task. We then introduce our data set of naturally occurring interest signals, followed by an investigation of the factors that influence them.

3.1 Formal Task Definition

We define our task as follows. Let U be the set of all documents and A be the set of all candidate text spans in all documents in U, generated by some candidate generator. Let $A_u \subset A$ be the set of candidates in $u \in U$. We formally define the interestingness task as learning the function below, where $\sigma(u, a)$ is the interestingness of candidate a in [2]:

$$\sigma: U \times A \to \mathbb{R} \quad (1)$$

3.2 Data Set

User browsing events on the web (i.e., a user clicking from one document to another) form a naturally occurring collection of interestingness signals. That is when a user clicks on an anchor in a document, we can postulate that the user is interested in learning more about it, modulo erroneous clicks.

We collect a large set of many millions of such user browsing events from session logs of a commercial web browser. Specifically, we collect from these logs each occurrence of a user click from one Wikipedia page to another during a one month period, from all users in all parts of the world. We refer to each such event as a *transition*. For each transition, our browser log provides metadata, including user profile information, geo-location information and session information (e.g., time of click, source/target dwell time, etc.) Our data set includes millions of transitions between Wikipedia pages.

For our task we require: (1) a mechanism for generating candidate things; (2) ample clicks to serve as a reliable signal of interestingness for training our models; and (3) accessible content. Our focus on Wikipedia satisfies all. First, Wikipedia pages tend to contain many anchors, which can serve as the set of candidate things to be ranked. Second, Wikipedia attracts enough traffic to obtain robust browsing transition data. Finally, Wikipedia provides full content[3] dumps. It is important here to note that our choice of Wikipedia as a test bed for our experiments does not restrict the general applicability of our approach: We propose a semantic model (Section 4.2) for mining latent features relevant to the phenomenon of interestingness which is general and can be applied to generic Web document collections.

Using uniform sampling, we split our data into three sets: a development set (20%), a training set (60%) and a test set (20%). We further subdivide the test set by assigning each transition as belonging to the HEAD, TORSO, or TAIL, which we compute using inverse CDF sampling on the test set. We do so by assigning the most frequently occurring transitions, accounting for 20% of the (source) traffic, to the HEAD. Similarly, the least frequently occurring transitions, accounting for 20% of the (source) traffic, are assigned to the TAIL. The remaining transitions are assigned to the TORSO. This three-way split reflects a common practice in the IR community and is based on the observation that web traffic frequencies show a very skewed distribution, with a small set of web pages attracting a large amount of traffic, and a very long tail of infrequently visited sites. Different regions in that distribution often show marked differences in behaviour, and models that are useful in one region are not necessarily as useful in another.

---

[2] We fix $\sigma(u, a) = 0$ for all $a \notin A_u$.

[3] We utilize the May 3, 2013 English Wikipedia dump from http://dumps.wikimedia.org, consisting of roughly 4.1 million articles.

3.3 What Factors Influence Interestingness?

We manually inspected 200 random transitions from our development set. Below, we summarize our observations.

Only few things on a page are interesting: The average number of anchors on a Wikipedia page is 79. Of these, only very few are actually clicked by users. For example, the Wikipedia article on the TV series "The Big Bang Theory" leads to clicks on anchors linking to the pages of the series' actors for 90% of transitions (while these anchors account for only a small fraction of all unique anchors on that page).

The semantics of source and destination pages is important: We manually determined the entity type of the Wikipedia articles in our sample, according to schema.org classes. 49% of all source urls in our data sample are of the `Creative Work` category, reflecting the strong popular interest in movies (37%), actors (22%), artists (18%), and television series (8%). The next three most prominent categories are `Organization` (12%), `Person` (11%) and `Place` (6%). We observed that transitions are influenced by these categories. For example, when the source article category is `Movie`, the most frequently clicked pages are of category `Actor` (63%) and `Character` (13%). For source articles of the `TVSeries` category, `Actor` destination articles account for 86% of clicks. `Actor` articles lead to clicks on `Movie` articles (45%) and other `Actor` articles (26%), whereas `Artist` articles lead to clicks on other `Artist` articles (29%), `Movie` articles (17%) and `MusicRecording` articles (18%).

The structure of the source page plays a role: It is well known that the position of a link on a page influences user click behavior: links that are higher on a page or in a more prominent position tend to attract more clicks. We noticed similar trends in our data.

The user plays a role: We hypothesized that users from different geographic and cultural backgrounds might exhibit different interests, or that interests are time-bound (e.g., interests on weekends differ from those on week days, daytime from nighttime, etc.) Initial experiments showed small effects of these factors, however, a more thorough analysis on a larger sample is necessary, which we leave for future work.

4 Modeling Interestingness

We cast the problem of learning the interestingness function σ (see Eq. 1) as a discriminative regression learning problem. Below, we first describe this model, and then we introduce our semantic topic model which serves to provide semantic features for the discriminative learner.

4.1 Discriminative Model

Although our task is to predict ITs from unstructured documents, we can leverage the user interactions in our data, described in Section 3.2 as our training signal.

Given a source document $s \in U$, and an anchor in s leading to destination document d, we use the aggregate click frequency of this anchor as a proxy for its interestingness, i.e.:

$$\sigma(s, d) = p(d|s) \qquad (2)$$

where $p(d|s)$ is the probability of a user clicking on the anchor to d when viewing s[4]. We use $p(d|s)$ as our regression target computed from our training data.

For our learning algorithm, we use boosted decision trees (Friedman, 1999). We tune our hyperparameters (i.e., number of iterations, learning rate, minimum instances in leaf nodes, and the maximum number of leaves) using cross-validation on the development set. Each transition in our training data is represented as a vector of features, where the features fall into three basic families:

1 Anchor features (**Anc**): position of the anchor in the document, frequency of the anchor, anchor density in the paragraph, and whether the anchor text matches the title of the destination page.
2 User session features (**Ses**): city, country, postal code, region, state and timezone of the user, as well as day of week, hour, and weekend vs. workday of the occurrence of the transition.

[4] For notational convenience, we use $\sigma(s, d)$ even though Eq. 1 defines its second argument as being a candidate text span. Here, it is implicit that *d* consists of both the target document and the anchor text (which serves as the candidate text span).

3 Semantic features: sourced in various experimental configurations from (1) Wikipedia page categories as assigned by Wikipedia editors (**Wiki**) or from (2) an unsupervised joint topic transition model (**JTT**) of source and destination pages (described in detail in the next section).

In some experimental configurations we use Wikipedia page categories as semantic features. We show in our experiments (see Section 5) that these are highly discriminative. It is important to note that editor-labeled category information is available in the Wikipedia domain but not in others. In other words, we can use this information to verify that semantics indeed is influential for interestingness, but we should design our models to not rely on this. We thus build an unsupervised semantic model of source and destination pages, which serves the purpose of providing semantic information without any domain-specific annotation.

4.2 The Semantics of Interestingness

As indicated in Section 3, the semantics of source and destination pages, s and d, influence the likelihood that a user is interested in d after viewing s. In this section we propose an unsupervised method for modeling the transition semantics between s and d. As outlined in the previous section, this model then serves to generate semantic features for our discriminative model of interestingness.

Referring to the notations in Table 1, we start by positing a distribution over the joint latent transition topics (in the higher level of semantic space), θ_t for each transition t. The corresponding source $t(s)$ and destination $t(d)$ articles of a given transition t are assumed to be admixtures of latent topics that are conditioned on the joint topic transition distribution, θ_t. For ease of reference, we will refer to this model as the Joint Transition Topic Model (**JTT**). The variable names and their descriptions are provided in Table 1. Figure 1 shows the plate notation of our model and the generative process:

1. For each topic k, draw $\varphi_k \sim Dir(\beta)$
2. For each transition t:
   a. Draw the joint topic transition distribution, $\theta_t \sim Dir(\alpha)$
   b. For each word token $j \in \{1 \dots N_S\}$:
      i. Draw $z^S_{t,j} \sim Mult(\theta_t)$
      ii. Emit $w^S_{t,j} \sim Mult(\varphi_k)$
   c. For each word token $j \in \{1 \dots N_D\}$:
      i. Draw $z^D_{t,j} \sim Mult(\theta_t)$
      ii. Emit $w^D_{t,j} \sim Mult(\varphi_k)$

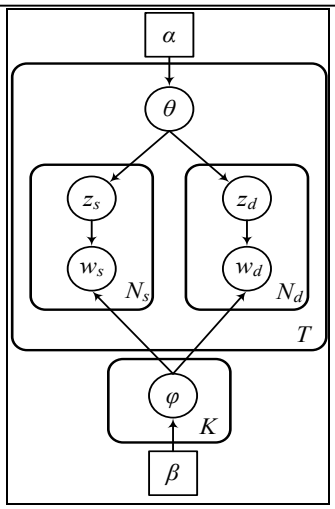


Figure 1. Generative Process and Plate Notation of JTT.

Variable	Description	Variable	Description
t	A transition t	Z^S, Z^D	Set of all topics in src, dest pages
$t(s)$, $t(d)$	The src and dest pages of t	W^S, W^D	Set of all word tokens in src, dest pages
$\theta_t \sim Dir(\alpha)$	Joint src/dest topic distribution	$\Theta = \{\theta_t\}$	Set of all latent joint transition topic distributions
z_s, z_d	Latent topics of $t(s)$, $t(d)$	$\Phi = \{\varphi_k\}$	Set of all latent topics
w_s, w_d	Observed word tokens of $t(s)$, $t(d)$	$\theta_{t,k}$	Contribution of topic k in transition t
$\varphi_k \sim Dir(\beta)$	Latent topic-word distributions for topic k	$w^S_{t,j}, w^D_{t,j}$	jth word of transition t in $t(s)$, $t(d)$
α, β	Dirichlet parameters for θ, φ	$z^S_{t,j}, z^D_{t,j}$	Latent topic of jth word of t in $t(s)$, $t(d)$
N_s, N_d	No. of terms in src and dest pgs of t	$n^S_{t(s),k}$	No. of words in $t(s)$ assigned to topic k
$T = \{t\}$	Set of all transitions, t	$n^D_{t(d),k}$	No. of words in $t(d)$ assigned to k
K	No. of topics	$n^S_{k,v}$	No. of times word v assigned to k in W^S
V	No. of unique terms in the vocab.	$n^D_{k,v}$	No. of times word v assigned to k in W^D

Table 1. List of notations.

Exact inference for JTT is intractable. Hence, we use Markov Chain Monte Carlo (MCMC) Gibbs sampling. Rao-Blackwellization (Bishop, 2006) is used to reduce sampling variance by collapsing latent variables θ and φ. Owing to space constraints, we omit the full derivation details. The full joint can be written succinctly as follows:

$$P(W^S, W^D, Z^S, Z^D) = \left(\prod_{t=1}^{T} \frac{B\left(n^S_{t(s),[\,]} + n^D_{t(d),[\,]} + \alpha\right)}{B(\alpha)}\right) = \left(\prod_{t=1}^{K} \frac{B(n^S_{k,[\,]} + n^D_{k,[\,]} + \beta)}{B(\beta)}\right) \quad (3)$$

Omission of a latter index in the count variables, denoted by [ ], corresponds to the row vector spanning over the latter index. The corresponding Gibbs conditional distributions for z^S and z^D are detailed below, where the subscript $(\neg(t,j))$ denotes the value of the expression excluding the counts of the term (t,j):

$$p(z^S_{t,j} = k| \ldots) \propto \frac{\left(n^S_{t(s),k}\right)_{\neg(t,j)} + n^D_{t(d),k} + \alpha}{\sum_{k=1}^{K}\left(\left(n^S_{t(s),k}\right)_{\neg(t,j)} + n^D_{t(d),k} + \alpha\right)} \times \frac{(n^S_{k,v})_{\neg(t,j)} + n^D_{k,v} + \beta}{\sum_{v=1}^{V}\left((n^S_{k,v})_{\neg(t,j)} + n^D_{k,v} + \beta\right)} \quad (4)$$

$$p(z^D_{t,j} = k| \ldots) \propto \frac{n^S_{t(s),k} + \left(n^D_{t(d),k}\right)_{\neg(t,j)} + \alpha}{\sum_{k=1}^{K}\left(n^S_{t(s),k} + \left(n^D_{t(d),k}\right)_{\neg(t,j)} + \alpha\right)} \times \frac{n^S_{k,v} + (n^D_{k,v})_{\neg(t,j)} + \beta}{\sum_{v=1}^{V}\left(n^S_{k,v} + (n^D_{k,v})_{\neg(t,j)} + \beta\right)} \quad (5)$$

We learn our joint topic model from a random traffic-weighted sample of 10,000 transitions, which are randomly sampled from the development set outlined in Section 3.2[5]. The decision to use this sample of 10,000 transitions is based on the observation that there were no statistically significant performance gains for models trained on more than 10k transitions. The Dirichlet hyperparameters are set to $\alpha = 50/K$ and $\beta = 0.1$ according to the values suggested in (Griffiths and Steyvers, 2004). The number of topics, K, is empirically set to 50. We also conducted pilot experiments with other hyperparameter settings, larger transition sets and more topics but we found no substantial difference in the end-to-end performance. Although increasing the number of topics and modeling more volume usually results in lowering perplexities and better fitting in topic models (Blei et al., 2003), it can also result in redundancy in topics which may not be very useful for downstream applications (Chen et al., 2013). For all reported experiments we use the posterior estimates of our joint model learned according to the above settings. In our discriminative interestingness model, we use three classes of features from JTT to capture the latent topic distributions of the source page, the destination page, and the joint topics for that transition. These correspond to source topic features (Z^S, labeled as JTTsrc in charts), destination topic features (Z^D, labeled as JTTdst), and transition topic features (Θ, labeled as JTTtrans). Each of these three sets comprises 50 features, for a total of 150.Θ is the distribution over joint src and dst topics that

[5] Note that we use the development set to train our semantic model since it is ultimately used to generate features for our discriminative learner of Section 4. Since the learner is trained using the training set, this strategy avoids overfitting our semantic model to the training set.

appear in a particular transition. Z^S and Z^D are the actual topic assignments for individual words in src and dst. Upon learning the JTT model, for each K topics, we get a probability of that topic appearing in the transition, in the src, and in the dst document (by taking the posterior point estimates for latent variables Θ, Z^S, Z^D respectively). The GBDT implementation we use for our discriminative model performs binning of these real-valued features over an ensemble of DTs.

5 Experiments

We evaluate our interestingness model on the task of proposing k anchors on a page that the user will find interesting (*highlighting* task). Recall the interestingness function σ from Eq. 1. In the highlighting task, a user is reading a document $s \in U$ and is interested in learning more about a set of anchors. Our goal in this task is to select k anchors that maximize the cumulative degree of interest of the user, i.e.:

$$\underset{A_s^k=(a_1,\ldots,a_k|a_i\in A_s)}{\operatorname{argmax}} \sum_{a_i\in A_s^k} \sigma(s,a_i) \qquad (6)$$

In other words, we consider the ideal selection to consist of the *k* most interesting anchors according to $\sigma(s,a)$.We compare the interestingness ranking of our models against a gold standard function, σ', computed from our test set. Recall that we use the aggregate click frequency of an anchor as a proxy for its interestingness. As such, the gold standard function for the test set is computed as:

$$\sigma'(s,a) = p(a|s) \qquad (7)$$

where $p(a|s)$ is the probability of a user clicking on the anchor a when viewing s.

Given a source document s, we measure the quality of a model's interestingness ranking against the ideal ranking defined above using the standard nDCG metric (Manning et al., 2008). We use the interestingness score of the gold standard as the relevance score.

Table 2 shows the nDCG results for two baselines and a range of different feature sets. The first high-level observation is that the task is difficult, given the low baseline results. Since there are many anchors on an average page, picking a random set of anchors yields very low nDCG scores. The nDCG numbers of our baselines increase as we move from HEAD to TORSO to TAIL, due to the fact that the average number of links per page (not unique) decreases in these sets from 170 to 94 to 41[6]. The second baseline illustrates that it is not sufficient to simply pick the top *n* anchors on a page.

Next, we see that using our set of anchor features (see Section 4.1) in the regression model greatly improves performance over the baselines, with the strongest numbers on the HEAD set and decreasing effectiveness in TORSO and TAIL. This shows that the distribution of interesting anchors on a page differs according to the popularity of the source content, possibly also with the length of the page. Our best performing model is the one using anchor features and all three sets of latent semantic features (Table 2, row 6; source, destination, and transition topics).

The biggest improvement is obtained on the HEAD data. This is not surprising given that the topic model is trained on a traffic weighted sample of Wikipedia articles and that HEAD pages tend to have more content, making the identification of topics more reliable. Regarding the individual contributions of the latent semantic features (Table 2, rows 4, 5), destination features alone hurt performance on the HEAD set. Latent semantic source features lead to a boost across the board, and the addition of latent semantic transition topic features produces the best model, with gains especially pronounced on the HEAD data. Figure 2 further shows the performance of our best configuration across ALL, HEAD, TORSO, and TAIL. Interestingly, the TAIL exhibits better performance of the model than the TORSO (with the exception of nDCG at rank 3 or higher). We hypothesize that this is because the average number of anchors in a TAIL page is less than half of that in a TORSO page.

[6] Wikipedia editors tend to spend more time on more frequently viewed documents, hence they tend contain more content and more anchors.

nDCG %	HEAD				TORSO				TAIL			
n	@1	@2	@5	@10	@1	@2	@5	@10	@1	@2	@5	@10
Baseline: random	4.07	4.90	6.24	8.10	3.56	4.83	7.66	10.92	6.20	11.74	19.50	25.82
Baseline: first *n* anchors	9.99	12.47	17.72	24.33	7.17	9.87	17.06	23.97	9.06	16.66	27.35	34.82
Anc	21.46	22.50	25.30	29.47	13.85	16.80	22.85	28.20	10.88	19.16	29.33	36.48
Anc+JTT_{dst}	13.97	16.33	19.69	23.78	11.37	14.17	19.67	24.66	11.62	19.69	29.69	36.35
Anc+JTT_{dst}+JTT_{src}	26.62	30.03	34.82	39.38	17.05	20.82	**27.15**	**32.48**	12.27	**21.56**	**31.88**	**38.85**
Anc+JTT_{dst}+JTT_{src}+JTT_{trans}	**34.49**	**35.21**	**38.01**	**41.80**	**18.32**	**21.69**	**28.03**	**33.22**	**13.06**	**21.68**	**32.13**	**38.01**

Table 2. Highlighting performance (% nDCG @ n) for different feature sets across HEAD, TORSO, and TAIL. Bold indicates statistically significant best systems (with 95% confidence).

Not shown in these results are the effects of using user session features. We consistently found that these features did not improve upon the configurations where anchor and JTT features are used. We do not, however, rule out the potential of such features on this task, especially in light of our data analysis observations from Section 3.3, which suggest an effect from these factors. We leave a more in-depth study of the potential contribution of these types of features for future research.

We now address the question how our unsupervised latent semantic features perform compared to the editor-assigned categories for Wikipedia pages, for two reasons. First, it is reasonable to consider the manually assigned Wikipedia categories as a (fine-grained) oracle for topic assignments. Second, outside of Wikipedia, we do not have the luxury of manually assigned categories/topics. As illustrated in Figure 3, we found that Wikipedia categories outperform the JTT topic features, but the latter can recover about two thirds of the nDCG gain compared to Wikipedia categories.

Finally, in the HEAD part of the data, we have enough historical clickthrough data that we could directly leverage for prediction. We conducted experiments where we used the prior probability $p(d|s)$ obtained from the development data (both smoothed and unsmoothed). Following this strategy we can achieve up to 65% nDCG@10 as shown in Figure 4 where the use of prior history (labeled "History: Target | Source Prior") is compared to our best model and to baselines. As stressed before, in most real-life applications, this is not a viable option since anchors or user-interaction logs are unavailable. Even in web browsing scenarios, the TORSO and TAIL have no or only very sparse histories. Furthermore, the information is not available in a "cold start" scenario involving new and unseen pages. We also examined whether the general popularity of a target page is sufficient to predict an anchor's interestingness, and we found that this signal performs better than the baselines, but significantly worse than our models. This series is labeled "History: Target Prior" in Figure 4.

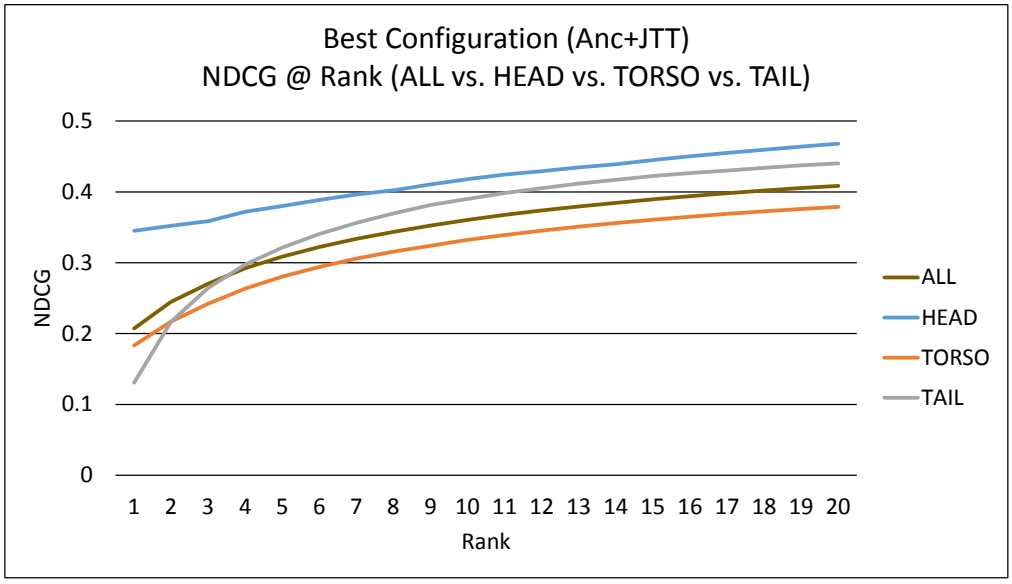


Figure 2. NDCG comparison across overall performance (ALL) versus HEAD, TORSO, and TAIL subsets, on the Highlighting task.

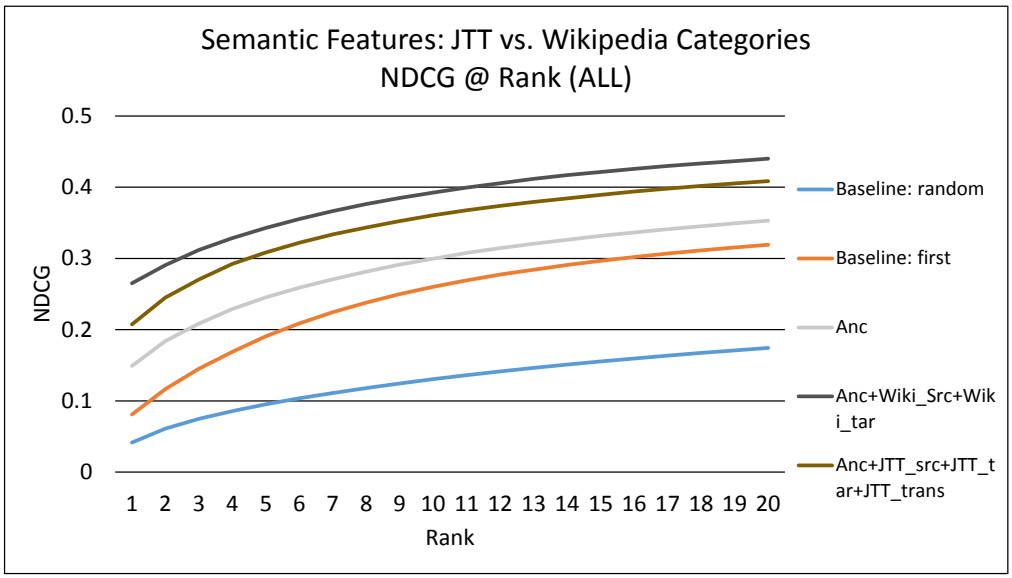


Figure 3. JTT features versus Wikipedia category features on Highlighting task.

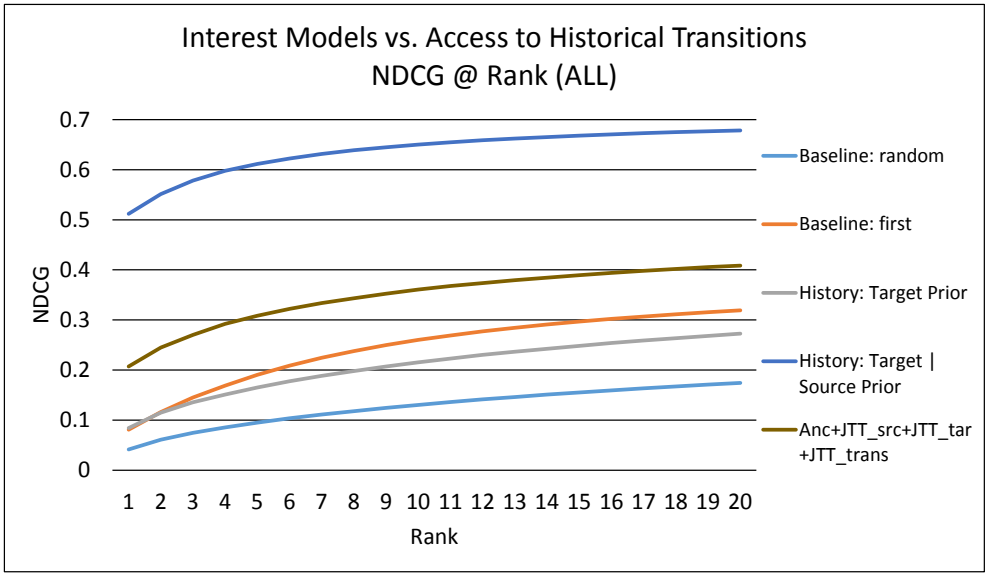


Figure 4. Highlighting task comparison between baselines, best configuration using JTT, and models with historical transitions.

Our highlights task reflects the main goal of our paper, i.e., to predict interestingness in the context of any document, whether it be a web page, an email, or a book. A natural extension of our work, especially in our experimental setting with Wikipedia transitions, is to predict the next click of a user, i.e., *click prediction*.

There is a subtle but important difference between the two tasks. Highlights aims to identify a set of interesting nuggets for a source document. A user may ultimately click on only a subset of the nuggets, and perhaps not in the order of most interest. Our experimental metric, nDCG, reflects this ranking task well. Click prediction is an inherently more difficult task, where we focus on predicting exactly the next click of a specific user. Unlike in the highlights task, there is no partial credit for retrieving other interesting anchors. Only the exact clicked anchor is considered a correct result. As such, we utilize a different metric than nDCG on this task. We measure our model's performance on the task of click prediction using cumulative precision. Given a unique transition event $\tau(s,a,d)$ by a particular user at a particular time, we present the transition, minus the gold anchor a and destination d, to our models, which in turn predict an ordered list of most likely anchors on which the user will click. The cumulative precision at k of a model, is 1 if any of the predicted anchors matched a, and 0 otherwise.

Table 3 outlines the results on this task and Figure 5 shows the corresponding chart for our best configuration. Note that in the click prediction task, the model performs best on the TAIL, followed by TORSO and HEAD. This seems to be a reflection of the fact that in this harder task, the total number of anchors per page is the most influential factor in model performance.

Cumulative Precision %	**HEAD**				**TORSO**				**TAIL**			
n	@1	@2	@5	@10	@1	@2	@5	@10	@1	@2	@5	@10
Baseline: random	1.07	2.08	5.29	10.55	1.94	3.91	9.71	19.00	5.97	11.66	26.43	44.94
Baseline: first *n* anchors	2.68	5.77	16.73	33.78	4.10	8.19	22.86	42.08	8.77	16.57	36.80	58.52
Anc	8.40	12.55	22.04	34.22	8.70	14.37	27.56	42.68	10.59	19.08	38.27	59.04
Anc+JTT_{dst}	5.48	9.19	17.77	29.14	6.93	12.07	23.90	38.00	11.23	19.59	38.46	57.87
Anc+JTT_{dst}+JTT_{src}	9.02	15.65	30.05	**44.72**	10.11	17.42	32.08	**47.07**	**11.95**	**21.47**	**40.96**	**61.24**
Anc+JTT_{dst}+JTT_{src}+JTT_{trans}	**11.53**	**18.43**	**31.93**	**45.36**	**10.86**	**18.19**	**32.96**	**47.66**	**12.64**	**21.58**	**41.27**	**61.28**

Table 3. Click prediction results for different feature sets across HEAD, TORSO, and TAIL. Bold indicates statistically significant best systems (with 95% confidence).

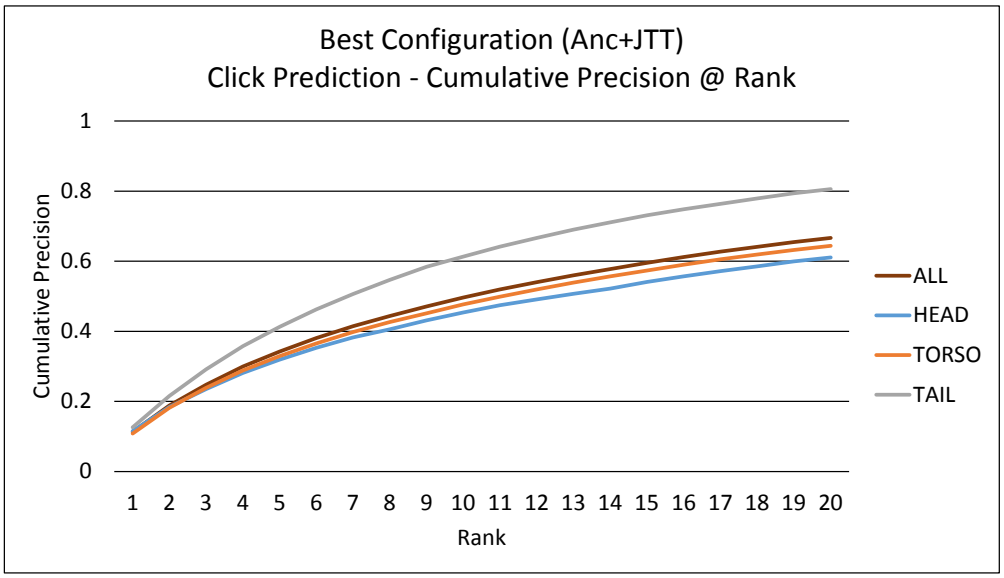


Figure 5. Overall performance (ALL) versus HEAD, TORSO, and TAIL subsets on click prediction.

6 Conclusion and Future Directions

We presented a notion of an IT on a page that is grounded in observable browsing behavior during content consumption. We implemented a model for prediction of interestingness that we trained and tested within the domain of Wikipedia. The model design is generic and not tied to our experimental choice of the Wikipedia domain and can be applied to other domains. Our model takes advantage of semantic features that we derive from a novel joint topic transition model. This semantic model takes into account the topic distributions for the source, destination, and transitions from source to destination. We demonstrated that the latent semantic features from our topic model contribute significantly to the performance of interestingness prediction, to the point where they perform nearly as well as using editor-assigned Wikipedia categories as features. We also showed that the transition topics improve results over just using source and destination semantic features alone.

A number of future directions immediately suggest themselves. First, for an application that marks interesting ITs on an arbitrary page, we would need a detector for IT candidates. A simple first approach would be to use a state-of-the-art Named Entity Recognition (NER) system to cover at least a subset of potential candidates. This does not solve the problem entirely, since we know that named entities are not the only interesting nuggets – general terms and concepts can also be of interest to a reader. On the other hand we do have reason to believe that entities play a very prominent role in web content consumption, based on the frequency with which entities are searched for (see, for example Lin et al. 2012 and the references cited therein). Using an NER system as a candidate generator would also allow us to

add another potentially useful feature to our interestingness prediction model: the type of the entity. One could also envision jointly modeling interestingness and candidate detection.

A second point concerns the observation from the previous section on the different regularities that seem to be at play according to the popularity and possibly the length of an article. More detailed experiments are needed to tease out this influence and possibly improve the predictive power of the model. User session features did not contribute to model performance when used in conjunction with other feature families, but closer investigation of these features is warranted for more personalized models of interestingness. Finally, a number of options regarding JTT could be explored further. Being trained on a traffic-weighted sample of articles, the topic model predominantly picks up on popular topics. This could be remedied by training on a non-weighted sample, or, more promisingly, on a larger non-weighted sample with a larger K, i.e. more permissible total topics.

References

Agichtein, E., Brill, E., and Dumais, S. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of SIGIR*. pp. 19-26.

Bandari, R., Asur, S., and Huberman, B. A. 2012. The Pulse of News in Social Media: Forecasting Popularity. In *Proceedings of ICWSM*.

Blei, D. M., Ng, A. Y., and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993-1022.

Broder, A., Fontoura, M., Josifovski, V., and Riedel, L. 2007. A semantic approach to contextual advertising. In *Proceedings of SIGIR*. pp. 559-566.

Bishop, C.M. 2006. Pattern Recognition and Machine Learning. Springer.

Chatterjee, P., Hoffman, D. L., and Novak, T. P. 2003. Modeling the clickstream: Implications for web-based advertising efforts. *Marketing Science*, *22*(4), 520-541.

Chen, Z., Mukherjee, A., Liu, B., Hsu, M., Castellanos, M. and Ghosh, R. 2013. Leveraging Multi-Domain Prior Knowledge in Topic Models. In *Proceedings of IJCAI*. pp. 2071-2077.

Claypool, M., Le, P., Wased, M., & Brown, D. 2001a. Implicit interest indicators. In *Proceedings of the 6th international conference on Intelligent user interfaces* (pp. 33-40). ACM.

Craswell, N., Zoeter, O., Taylor, M., and Ramsey, B. 2008. An experimental comparison of click position-bias models. In *Proceedings of WSDM*. pp. 87-94.

Erosheva, E., Fienberg, S., and Lafferty, J. 2004. Mixed membership models of scientific publications. In *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1). pp. 5220-5227.

Friedman, J. H. 1999. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189-1232, 1999.

Gamon, M., Yano, T., Song, X., Apacible, J. and Pantel, P. 2013. Identifying Salient Entities in Web Pages. In *Proceedings CIKM*. pp. 2375-2380.

Gao, J., Toutanova, K., and Yih, W. T. 2011. Clickthrough-based latent semantic models for web search. In *Proceedings of SIGIR*. pp. 675-684.

Graepel, T., Candela, J.Q., Borchert, T., and Herbrich, R. 2010. Web-scale Bayesian Click-Through Rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine. In *Proceedings of ICML*. pp. 13-20.

Griffiths, T.L and Steyvers, M. 2004. Finding Scientific Topics. *Proceedings of the National Academy of Science*, 101, suppl 1, 5228-5235.

Guo, F., Liu, C., Kannan, A., Minka, T., Taylor, M., Wang, Y.-M, and Faloutsos, C. 2009a. Click chain model in web search. In *Proceedings of WWW*. pp. 11-20.

Guo, J., Xu, G., Cheng, X., and Li, H. 2009b. Named entity recognition in query. In *Proceedings of SIGIR*. pp. 267-274.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *Proceedings of KDD*. pp. 133-142.

Joachims, T., Granka, L., Pan, B., Hembrooke, H. and Gay, G. 2005. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of SIGIR*. pp. 154-161.

Lerman, K., and Hogg, T. 2010. Using a model of social dynamics to predict popularity of news. In *Proceedings of WWW*. pp. 621-630.

Lin, T., Pantel, P., Gamon, M., Kannan, A., and Fuxman, A. 2012. Active objects: actions for entity-centric search. In *Proceedings of WWW*. pp. 589-598.

Manning, C. D., Raghavan, P., and Schutze, H. 2008. Introduction to Information Retrieval. Cambridge University Press.

Mueller, F., & Lockerd, A. 2001. Cheese: tracking mouse movement activity on websites, a tool for user modeling. In *CHI'01 extended abstracts on Human factors in computing systems* (pp. 279-280). ACM.

Paranjpe, D. 2009. Learning document aboutness from implicit user feedback and document structure. In *Proceedings of CIKM*. pp. 365-374.

Shen, S., Hu, B., Chen, W., and Yang, Q. 2012. Personalized click model through collaborative filtering. In *Proceedings of WSDM*. pp. 323-333.

Szabo, G., and Huberman, B. A. 2010. Predicting the popularity of online content. *Com-munications of the ACM*, *53*(8), 80-88.

Varela, F. J., Thompson, E. T., & Rosch, E. 1991. The embodied mind: Cognitive science and human experience. *The MIT Press*.

Yano, T., Cohen, W. W., & Smith, N. A. 2009. Predicting response to political blog posts with topic models. In *Proceedings of NAACL*. pp. 477-485.

Context Dependent Claim Detection

Ran Levy Yonatan Bilu Daniel Hershcovich Ehud Aharoni Noam Slonim
IBM Haifa Research Lab / Mount Carmel, Haifa, 31905, Israel
{ranl,yonatanb,danielh,aehud,noams}@il.ibm.com

Abstract

While discussing a concrete controversial topic, most humans will find it challenging to swiftly raise a diverse set of convincing and relevant claims that should set the basis of their arguments. Here, we formally define the challenging task of automatic claim detection in a given context and discuss its associated unique difficulties. Further, we outline a preliminary solution to this task, and assess its performance over annotated real world data, collected specifically for that purpose over hundreds of Wikipedia articles. We report promising results of a supervised learning approach, which is based on a cascade of classifiers designed to properly handle the skewed data which is inherent to the defined task. These results demonstrate the viability of the introduced task.

1 Introduction

The ability to argue in a persuasive manner is an important aspect of human interaction that naturally arises in various domains such as politics, marketing, law, and health-care. Furthermore, good decision making relies on the quality of the arguments being presented and the process by which they are resolved. Thus, it is not surprising that argumentation has long been a topic of interest in academic research, and different models have been proposed to capture the notion of an argument (Toulmin, 1958; Freeley and Steinberg, 2008). A fundamental component which is common to all these models is the concept of a *claim* (or *conclusion*). Specifically, at the heart of every argument lies a single claim, which is the assertion the argument aims to prove. Given a concrete topic, or context, most humans will find it challenging to swiftly raise a diverse set of convincing and relevant claims that should set the basis of their arguments. The purpose of this work is to formally define the challenging task of automatic claim detection *in a given context*, to outline a preliminary solution to this task, and to assess its performance over annotated real world data, collected specifically for this purpose.

In his classical argument model, Toulmin defined a claim as a conclusion whose merit must be established (Toulmin, 1958). Since we are interested not in detecting claims in general (Mochales Palau and Moens, 2009; Teufel, 1999), but rather in detecting claims that are specifically relevant to a pre-defined concrete context, we suggest a definition with a more functional flavor. In practice, we found this definition easy to convey to human labelers, and consequently feasible to capture by automatic detection methods. In particular, we define the following two concepts:

- **Topic** – a short phrase that frames the discussion.
- **Context Dependent Claim (CDC)** – a general, concise statement that directly supports or contests the given Topic.

Given these definitions, as well as a few more detailed criteria to reduce the variability in the manually labeled data, human labelers were asked to detect CDCs for a diverse set of Topics, in relevant Wikipedia articles. The collected data, that were used to train and assess the performance of the statistical models, are now freely available upon request for academic research.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1489–1500, Dublin, Ireland, August 23-29 2014.

The distinction between a CDC and other related texts can be quite subtle, as illustrated in Table 1. For example, automatically distinguishing a CDC like S1 from a statement that simply defines a relevant concept like S2, from a claim which is not relevant enough to the given Topic like S3, or from a statement like S4 that merely repeats the given Topic in different words, is clearly challenging. Further, CDCs can be of different flavors, ranging from factual assertions like S1 to statements that are more of a matter of opinion (Pang and Lee, 2008) like S5, adding to the complexity of the task. Finally, our data suggest that even if one focuses on Wikipedia articles that are highly relevant to the given Topic, only $\approx 2\%$ of their sentences include CDCs. Moreover, as illustrated in Table 2, detecting the exact CDC boundaries is far from trivial, as in a typical single Wikipedia sentence there are many optional boundaries to consider. Thus, we are faced with a large number of candidate CDCs, of which only a tiny fraction represents positive examples, that might be quite reminiscent of some of the negative examples. Nonetheless, as we demonstrate, a supervised learning approach – which is based on a cascade of classifiers, carefully designed to properly handle the exceptionally skewed data – can address these difficulties to attain promising results.

Topic: The sale of violent video games to minors should be banned		
S1	Violent video games can increase children's aggression	V
S2	Video game addiction is excessive or compulsive use of computer and video games that interferes with daily life	X
S3	Many TV programmers argue that their shows just mirror the violence that goes on in the real world	X
S4	Violent video games should not be sold to children	X
S5	Video game publishers unethically train children in the use of weapons	V

Table 1: Examples for CDCs and for statements that should not be considered as CDCs. The V and X indicate if the candidate is a CDC for the given Topic, or not, respectively.

Topic: The sale of violent video games to minors should be banned	
S1	Because violence in video games is interactive and not passive, critics such as Dave Grossman and Jack Thompson argue *that* **violence in games hardens children to unethical acts**, calling first-person shooter games "murder simulators", although no conclusive evidence has supported this belief.

Table 2: A CDC is often only a small part of a single Wikipedia sentence – e.g., the part marked in bold in this example. Detecting the exact CDC boundaries represents an additional challenge.

In summary, the key contribution of this work is three fold: we define the new task of Context Dependent Claim Detection; introduce a novel manually labeled benchmark dataset, collected specifically for this task; and outline an automatic solution for which we report first results over these data. These results are encouraging, demonstrating the viability of the introduced task.

2 Task Definition and Related Work

We assume that we are given a Topic and a relatively small set of relevant free-text articles, provided either manually or by automatic retrieval methods (Macdonald et al., 2010; Zhang et al., 2007). Our goal is to automatically pinpoint CDCs within these documents. We further require that the detected CDCs are reasonably well phrased, so that they can be instantly and naturally used in a discussion about the given Topic.This task, which we term *Context Dependent Claim Detection (CDCD)*, can be of great practical importance in decision support and persuasion enhancement, in various domains where relevant massive corpora are available for mining.

CDCD can be seen as a sub-task in the emerging wider field of argumentation mining that involves identifying argumentative structures within a document, as well as their potential relations

(Mochales Palau and Moens, 2009; Cabrio and Villata, 2012; Wyner et al., 2012). However, CDCD has several distinctive key features. Most importantly, as implied by its name, a CDC is defined with respect to a given context – the input Topic. Thus, identifying general characteristics of a claim-like statement as done in (Mochales Palau and Moens, 2009) is not sufficient, since one should further identify the relevance of the candidate claim to the Topic. In addition, we do not restrict ourselves to a particular domain nor to structured data (Mochales Palau and Moens, 2009), but rather consider free-text Wikipedia articles in a diverse range of subject matters. Moreover, in CDCD we require pinpointing the exact claim boundaries, which do not necessarily match a whole sentence or even a clause in the original text, thus adding a significant burden to the task, compared to classical tasks that are focused on sentence classifications (Guo et al., 2011).

CDCD also shares some relations with Argumentative Zoning (Teufel, 1999; Guo et al., 2011). There, the aim is to divide the text of a scientific article into "zones", each characterized by the rhetorical nature of its content. However, our work is not limited to scientific literature that often has a more objective and less persuasive style. Further, as mentioned, we go beyond sentence classification, aiming to detect the exact claim boundaries, and require detecting only claims relevant to a given Topic, rather than just any claim mentioned in a given article.

Finally, another important line of research is the Textual Entailment (TE) framework (Dagan et al., 2009). In this framework, a text fragment, T, is said to entail a textual hypothesis H if the truth of H can be most likely inferred from T. While TE can be an important underlying utility in CDCD, and perhaps vice versa, the tasks are quite different. For example, common instances of TE are rephrases or summarizations of a sentence; however these cannot serve to support or contest a given Topic, as they merely repeat it (Table 1, S4). Furthermore, TE focuses on factual assertions, which can be true or false, whilst CDC may represent a relevant opinion that perhaps does not have a strict truth value associated to it (Table 1, S5). More generally, TE is typically focused on declarative statements. However, persuasion and argumentation often have an emotional aspect and thus may involve additional sentence types. Correspondingly, in our framework it is quite natural that the Topic, or the associated CDCs, will correspond to imperative sentences, or even to exclamatory sentences.

3 Data

Our supervised learning approach relies on labeled data that were collected as described below. A detailed description of the labeling process is given in (Aharoni et al., 2014). Due to the high complexity of the labeling task, we worked with in-house labelers which were provided with detailed guidelines, and went through rigorous training.

At the core of the labeling guidelines, we outlined the definition of a CDC as *a general, concise statement that directly supports or contests the given Topic*. In practice, the labelers were asked to label a text fragment as a CDC if and only if it complies with *all* the following five criteria:

- **Strength** – Strong content that directly supports/contests the Topic.
- **Generality** – General content that deals with a relatively broad idea.
- **Phrasing** – The labeled fragment should make a grammatically correct and semantically coherent statement.
- **Keeping text spirit** – Keeps the spirit of the original text.
- **Topic unity** – Deals with one topic, or at most two related topics.

The guidelines further included concrete examples, taken from Wikipedia articles, to clarify these criteria. When in doubt, the labelers were naturally asked to make a judgment call. The labelers work was carefully monitored, and they were provided with detailed feedback as needed.

We selected at random 32 debate motions from *http://idebate.org/debatabase*, covering a wide variety of topics, from atheism to the US responsibility in the Mexican drug wars. Each motion served as a single Topic and went through a rigorous labeling process, consisted of three stages. First, given a Topic, 5 labelers searched Wikipedia independently for articles that they believe contain CDCs. Next, each of the articles identified in this search stage was read by 5 labelers, that worked independently to detect

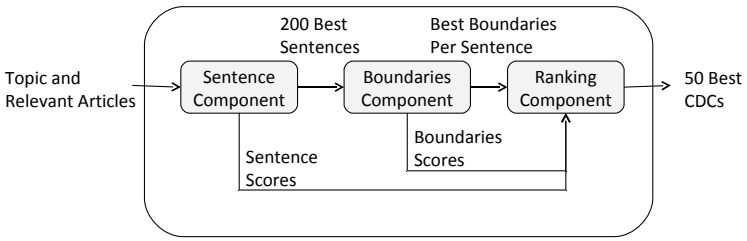
candidate CDCs. Finally, each of the candidate CDC proposed in the previous stage, was examined by 5 labelers that independently decided whether to confirm or reject the candidate. For the purposes of this work, we only considered candidate CDCs that were confirmed by a majority, i.e., by at least three labelers participating in the confirmation stage. The resulting labeled CDCs correspond to claims that can be naturally used in a discussion about the given Topic.

Through this process, for the 32 examined Topics, a total of 326 Wikipedia articles were labeled, yielding a total of 976 CDCs. Thus, even when considering articles that are presumably relevant to the given Topic, on average only 2 out of 100 sentences include a CDC. On the other hand, it should be noted that it was not clear to begin with that Wikipedia articles will contain CDCs that satisfy our relatively strict labeling guidelines. Nonetheless, on average, the labeling process yielded around 30 CDCs per Topic. Finally, the average Kappa agreement between pairs of labelers in the confirmation stage was 0.39, which is a relatively high agreement considering the complexity of the labeling task and the inherent elusiveness of the involved concepts.

4 Technical Approach

Our CDCD approach is designed as a cascade, or funnel, of three components (depicted in Figure 1), which receives as input a Topic along with relevant articles and should output the CDCs contained therein. The purpose of the funnel is to gradually focus on smaller and smaller CDC-containing text segments, while filtering out irrelevant text. Thus, the cascade divides the high level CDCD problem into smaller and more tangible problems – given an article, detect sentences that include CDCs; given a sentence, detect the exact CDC boundaries; given a set of CDC candidates, rank them so that true candidates are on top.

Figure 1: High level design of our CDCD approach. The indicated numbers are the ones used in our experiments, and in general should be determined based on the data and use case.



To appreciate the need for this cascade, let us first consider the scale of this detection problem. In our labeled data, per Topic we have an average of 10 relevant Wikipedia articles that contain at least 1 CDC. Each article contains an average of 155 sentences, each sentence spans on average 23 words, i.e., ≈ 200 sub-sentences, each of which may represent a candidate CDC. Thus, in principle, for each Topic we consider around 300,000 candidate CDCs, of which typically only 30 represent positive examples. By breaking the problem into independent sub-problems, at each stage the skew between positive examples and negative examples is less daunting, thus easier to handle by classical machine learning techniques. In addition, since much surplus text is filtered along the cascade, "downstream" components typically examine much smaller amounts of text, and thus can plausibly make use of more demanding algorithms. Finally, this conceptual separation naturally allows to develop features tailored individually to each task; for example, the grammatical correctness of a text fragment is clearly relevant for boundaries detection, while being irrelevant when classifying whole sentences.

In general, each component was developed independently within the classical supervised learning paradigm. Namely, numeric features are extracted from binary-labeled text segments, and are used to train a classifier. Next, this classifier is used to assign a score to each incoming test candidate and high-scoring candidates are passed on to the next component. In addition, rule-based filters might be used to discard some of the candidates. Note, while developing a "downstream" component we implicitly assumed that the previous "upstream" components have worked perfectly. Hence, for example, the *train-*

ing data for the boundary-detection component comprised only of sentences that truly contain CDCs. In what follows, we discuss each component in greater detail.

4.1 Sentence Component

The **Sentence Component** is responsible for detecting CDC-sentences, that is, to determine whether a candidate sentence contains a CDC or not. Some sentences contain more than one CDC but this is not very common. Hence, we consider this as a binary classification problem. The component receives an average of 1500 sentences per Topic and passes the top scoring 200 sentences to the next component. Specifically, we used Logistic Regression (LR) classifier due to its efficiency and its model interpretability, and focused our efforts on developing highly discriminative features for this classifier.

Since our focus is on detecting claims that are relevant to a given Topic, we naturally developed two main types of features – **Context features**, which examine the relation between the candidate sentence and the Topic; and **Context-free features**, which rely solely on the content of the candidate sentence, aiming to capture the probability it includes a "claim like" statement. For computing the context-features, we use the topic as it appears in debatabase (see Section 3). Specifically, the most dominant features we identified included:

MatchAtSubject: Cosine similarity between the Topic and the subjects of the candidate sentence – namely, all tokens that are marked as the subject of some sub-tree in the sentence ESG parse tree (McCord et al., 2012).

ExpandedCosineSimilarity: Cosine similarity between the Topic and semantic expansions of the candidate sentence. We use WordNet (Miller, 1995) to expand nouns into synonyms, hypernyms and hyponyms.

ESGFeatures: Binary features obtained from the ESG parser (McCord et al., 2012). The most prominent among them is an indicator of whether the sentence contains the token "that" that is assigned the "conjugate" feature by the ESG – see, for example, the emphasized "that" in the example in Table 2. Other features include: verb in present tense, infinitive verb, year, and named location.

SubjectivityScore: A classifier-based score that captures the degree of subjectivity in the sentence (Raykar et al., 2014).

Sentiment: Ratio of sentiment words in the sentence, based on a list of sentiment words from (Hu and Liu, 2004).

In addition to these Context features and Context-free features, we also developed a feature that represents a mix of these two types, that was proven essential to our performance, and relied on an extension of the Sequential Pattern Matching (SPM) algorithm (Srikant and Agrawal, 1996). Specifically, for this **SequentialPatternMatch** feature, each sentence token was encoded as a tuple describing several attributes for that token – e.g., the token's text, the token's POS tag, and various binary indicators, indicating if the token is a sentiment word, if it is mentioned in the given Topic, if it is included in an automatically learned lexicon of "claim words", and if it is identified by a NER utility (Finkel et al., 2005). A variant of the SPM algorithm (Srikant and Agrawal, 1996) was then used to detect patterns that characterize CDC-sentences, and these patterns were added to the features examined by the LR classifier. Specifically, each of these feature values was set to 1 if a candidate sentence had a match with the relevant pattern, and to 0 otherwise. For example, in Table 2, the word "that" is encoded as [that,IN,CDC] implying it is included in the "claim words" lexicon with POS tag IN; the word "games" is encoded as [games,NNS,Topic] implying it is mentioned in the Topic with POS tag NNS; and the word "unethical" is encoded as [unethical,JJ,Sentiment] implying it is a sentiment word with POS tag JJ. Correspondingly, in this sentence there is a match to the sequential pattern: [that,IN], [Topic], [Sentiment], which is one of the patterns detected automatically by our algorithm, as characterizing CDC-sentences. A more detailed description of this extended SPM approach will be given elsewhere.

It is worthwhile mentioning that one can envision this component as being broken up into two: One component that detects general claim-sentences, regardless of whether or not they relate to the Topic, based on the context-free features; Another component will detect relation to the Topic, regardless of whether or not the sentence is a claim. (Or some variation of this setting.)

The problem with this approach, as we see it, is that it greatly complicates the annotation guidelines and the associated annotation work. That is, without a topic, it is less clear how to define what a claim is, and deciding when a sentence is related to the topic is bound to be highly subjective. Furthermore, taking this approach would require adding additional detection and confirmation stages, lowering the amount of collected annotated data. For these reasons we have adopted the combined approach, even though it makes error analysis more difficult - without manual analysis it is not clear whether the errors are sentences which do not contain claims or which are unrelated to the topic, or both.

4.2 Boundaries Component

The **Boundaries Component** is responsible for detecting the exact CDC boundaries within CDC-sentences. Notice, that our definition of a CDC and the associated labeling guidelines – that gave rise to our ground-truth data – imply that in free text articles a CDC often do not correspond to an easily identified sub-tree in the sentence parse tree. For example, since we are interested in detecting focused claims the labelers are often led to mark a concise claim rather than a compound claim as in the following sentence – *"The argument of deprivation states that* **abortion is morally wrong** *because it deprives the fetus of a valuable future"*. Note that choosing the boundaries from *"abortion"* to *"future"* would have included two distinct claims. Similarly, since the labelers are guided to prefer more general versions of the CDC, as long as the original text spirit is kept, determining where the CDC should start could be quite a subtle decision. Thus, the exact CDC boundaries often rely on the semantics of the text, and not just on its grammatical structure. Correspondingly, identifying the exact CDC boundaries is far from trivial.

Based on similar considerations to those mentioned above, we divide this component into two sub-components.

Boundaries Coarse Filter: This sub-component is based on a Maximum Likelihood probabilistic model that given a sentence, selects the 10 sub-sentences whose boundaries most probably correspond to a CDC. Specifically, given a sentence, for each of its sub-sentences[1] we consider the token preceding it; the token with which it starts; the token with which it ends; and the token following it, where a token here can be a word or a punctuation mark. Given these four tokens, the algorithm estimates the probability that this sub-sentence represents a CDC. For practical purposes, the probability is estimated naively, by assuming that each token is independent of the others. In addition, the Boundaries Coarse Filter employs simple rules to filter out trivial cases such as sub-sentences that do not contain a verb and a noun, or sub-sentences for which the parse root is tagged as a sentence-fragment.

Boundaries Fine-Grained Filter: This sub-component is based on a LR classifier that selects one sub-sentence out of the 10 provided by the Boundaries Coarse Filter. Here as well we considered Context-free features and Context features, where the former type were typically weighted as more dominant by the LR classifier. Importantly, though, the Context-free features examined by this sub-component relied on the division of the entire sentence, as implied by the examined boundaries. Specifically, the candidate boundaries induce a division of the containing sentence into three parts: prefix, candidate body, and suffix, where the prefix and/or suffix might be empty. The features are then calculated for each of these three parts independently. Thus, for example, the presence of the word "that" in the prefix as opposed to its presence in the candidate body, will increase or decrease the confidence of the examined boundaries, respectively. In addition, the LR classifier considered features derived from the probabilistic model defined by the Boundaries Coarse Filter, that also aim to assess the probability that the examined boundaries yield a CDC.

Next, we elaborate on some of the dominant features examined by the Boundaries Fine-Grained Filter.

CDC-Probability features: These features indicate the conditional probability that the examined boundaries define a CDC, given the tokens around and within these boundaries. For example, the **Word-Before-Word-After** numeric feature, denoted $P(ta, tb)$, is defined as follows. Let $\{t_1, \ldots, t_n\}$ represent the list of tokens in a sentence, where a token is a word or a punctuation mark, then $P(ta, tb)$ is the probability that the sub-sentence $\{t_i, \ldots, t_j\}$ represents a CDC, given that $t_{i-1} = t_a,\ t_{j+1} = t_b$, as

[1] Here, a "sub-sentence" is any consecutive sequence of three tokens or more, that is included in the examined sentence.

estimated from our training data. Similarly, the **Word-Before-First-PoS** feature is based on the estimated conditional probability that the candidate defined by the examined boundaries is a CDC, given the token before the boundaries, t_{i-1}, and the POS-tag of the first token within the boundaries, t_i. Other features of this type include the conditional probability based on the presence of single tokens within the boundaries, and the initial score assigned to the examined boundaries by the Boundaries Coarse Filter.

Sentence-Probability features: These features aim to indicate the probability that the examined boundaries induce a grammatically correct sentence. For this purpose we examine a set of 100,000 presumably grammatically correct sentences, taken from a separate set of Wikipedia articles, and estimate the probability of each word to appear in a given position in a valid sentence. Next, given the examined boundaries, we ask for each of its first three tokens and each of its last three tokens, what is the probability of having a grammatically correct sentence, given that the observed token is in its observed position.

ModifierSeparation: The ESG parser (McCord et al., 2012) describes the modifiers of its parsed tokens, such as the object of a verb. Typically, a token and its modifier should either be jointly included in the CDC, or not included in it. This notion gave rise to several corresponding features.

Parse Sentence Match: These are binary features that indicate whether the examined boundaries correspond to a sub-tree whose root is labeled "S" (sentence) by the Stanford parser (Socher et al., 2013) or by the ESG parser (McCord et al., 2012), while parsing the entire surrounding sentence.

"that-conj" matches CDC: A binary feature indicating whether in the ESG parsing we have a subordinator "that" token, whose corresponding covered text matches the examined boundaries.

DigitCount: Counts the number of digits appearing in the sentence – before, within, and after the examined boundaries.

UnbalancedQuotesOrParenthesis: Binary features, indicating whether there is an odd number of quote marks, or unbalanced parenthesis, within the examined boundaries.

4.3 Ranking Component

The **Ranking Component** is responsible for the final scoring of the CDC candidates. It is also based on a LR classifier, that considers the scores of all previous components, as well as additional features described below. A simpler alternative could have been to rely solely on the initial sentence component ranking. However, since CDCs often correspond to much smaller parts of their surrounding sentence, considering the scores of all previous components is more effective. In contrast to the components described above, for which the training set is fully defined by the labeled data, the Ranking Component needs be trained also on the output of its "upstream" components, since it relies on the scores produced by these components.

In addition, the Ranking Component is using the following features:

CandidateComplexity, a score based on counting punctuation marks, conjunction adverbs (e.g., "likewise", "therefore"), sentiment shifters (e.g., "can not", "undermine") and references, included in the candidate CDC.

Sentiment, **ExpandedCosineSimilarity** and **MatchAtSubject**, as in the Sentence Component above, estimated specifically for the CDC candidate.

5 Experiments

We describe the results of running the cascade of aforementioned components, in the designed order, in a Leave-One-Out (LOO) fashion, over 32 Topics. In each LOO fold, the training data consisted of the labeled data for 31 Topics, while the test data consisted of articles that included at least one CDC for the designated test Topic.

The **Sentence Component** was run with the goal of selecting 200 sentences for the test Topic, and sorting them so that CDC-containing sentences are ranked as high as possible. As shown in Table 3, the mean precision and recall of this component, averaged across all 32 folds, were 0.09 and 0.73, respectively. When looking at the top scoring 50 sentences per Topic, the mean precision and recall are 0.18 and 0.4, respectively. As evident by the last row in Table 3, these results are way beyond a trivial

random selection of sentences, indicating that the Sentence Component is capturing a strong statistical signal associated with CDCs.

	Precision	Recall	Precision @ 50	Recall @ 50
mean	0.09	0.73	0.18	0.4
std	0.05	0.19	0.10	0.21
min	0.01	0.27	0.02	0.10
max	0.18	1.00	0.40	1.00
rand	0.02	0.13	0.02	0.03

Table 3: Sentence component. Last line indicates the expected values had selection been made at random.

Next, we employed the two sub-components of the **Boundaries Component**. First, the Boundaries Coarse Filter selected 10 sub-sentences for each candidate sentence. Recall that for sentences which actually contain CDCs, the aim is to have this CDC kept among the selected 10 sub-sentences. As shown in Table 4, this happens for 98% of the CDC-containing sentences (see Table 4). In other words, if the examined sentence included a CDC, the Boundaries Coarse Filter almost always included this CDC as one of the top 10 sub-sentences it suggested for that sentence. In the second step, for each candidate sentence, the Boundaries Fine-Grained Filter sorted the 10 sub-sentences proposed by the Boundaries Coarse Filter, aiming to have the CDC – if one exists – at the top. As indicated in Table 4, if indeed a CDC was present amongst the 10 sub-sentences sorted by this component, then it was ranked first in 50% of the cases. These results as well are clearly way beyond what is expected by random sorting, indicating a strong statistical signal that was properly captured by this component.

	Boundaries Coarse Filter Recall	Boundaries Fine-Grained Filter Recall
mean	0.98	0.50
std	0.39	0.16
min	0.67	0.25
max	1.00	1.00
rand	0.04	0.004

Table 4: Boundaries component - The left column relates to the fraction of sentences where the labeled CDC is among the top 10 candidates ranked by the Coarse Filter. The right column relates to to the fraction of sentences where the labeled CDC is identified correctly by the Fine-Grained Filter. The last row indicates the expected values had selection been made at random.

Finally, the **Ranking Component** combines the scores generated in the previous steps, as well as additional features, to set the final order of CDC candidates. The goal of this component – similar to that of the entire CDCD task – is to select 50 CDC candidates with high precision. Note, that on average, there are around 30 labeled CDCs per Topic. Thus, on average, the maximal precision at 50 should be around 0.6. As indicated in Table 5, our final precision at 50, averaged across all 32 folds, was 0.12, which is again way beyond random performance. Focusing at our top predicions naturally results with even higher precision – for example, the precision of our top 5 predictions was on average 0.23.

It should be noted that the analysis presented here is fairly strict. A predicted CDC is considered as True Positive if and only if it precisely matches a labeled CDC, that was confirmed as such by at least three labelers. Thus, for example, if a predicted CDC was confirmed by only two out of five annotators, it will be considered as an error in the analysis above. Furthermore, if the predicted CDC has a significant overlap with a labeled CDC, it will still be considered as an error, even if it represents a grammatically correct variant of the labeled CDC, that was simply less preferred by the labelers due to relatively minor considerations. Thus, although we still need to quantify the frequency of these "weak" errors, it is clear that for most practical scenarios, the performance of our system are above the strict numbers described here.

	Precision @ 5	Precision @ 10	Precision @ 20	Precision @ 50
mean	0.23	0.20	0.16	0.12
std	0.21	0.20	0.11	0.07
min	0.00	0.00	0.00	0.00
max	0.80	0.60	0.50	0.32
rand	0.00008	0.00008	0.00008	0.00008

Table 5: Ranking component

Category	Number of candidates
Accept	27
Accept with corrections	3
Generality failed	9
Strength failed	145
Text Spirit failed	1
Multiple candidates	5
Repeats topic	5
Incoherent	37
Not a sentence	17

Table 6: Number of ”false” claims in each rejection category

6 Error Analysis

We present an analysis of the errors (using a slightly earlier version of the system). The analysis covered the same 32 Topics described above, where for each Topic we analyzed the errors among the top 10 predictions. In total there were 249 sentences which did not exactly match the annotated data. Each of these seemingly-erroneous CDC candidates was then given to 5 annotators, who had to confirm or reject it and select a rejection reason. The goal of the analysis is to understand the types of errors the system makes as well as to obtain feedback on text spans that were not originally detected by the labelers (possible misses). Specifically, the labelers were instructed to choose one of the options in the following list:

Accept - The candidate should be accepted as is.

Accept with correction - The candidate should be accepted with minor corrections.

Generality failed - The candidate is too specific.

Multiple Candidates - The candidate contains more than one claim.

Repeats Topic - The candidate simply reiterates the topic (or its negation) rather than claim something about it.

Strength failed - The candidate does not directly and explicitly supports or contests the topic.

Text Spirit failed - The candidate does not keep the spirit of text in which it appeared.

Incoherent - The candidate is not a coherent claim.

Not a sentence - The candidate is not grammatical.

A majority vote was used to obtain the final answer. Table 6 gives the number of candidates in each category. As can be seen, about 10% of the candidates were actually accepted in this round. Most of the errors were attributed to ”Strength Failed”, which is a fairly wide category. In future analysis we plan to break it down into more specific sub-categories. Table 7 gives some examples of candidates generated by the system (which do not exactly match the annotated data) and their corresponding categories.

7 Discussion and Future Work

We introduced the CDCD task which is scientifically challenging, and moreover, potentially invaluable for various novel applications. We outlined a machine learning approach to address this task, which is designed based on a cascade of classifiers for handling the special difficulties of this task, and in

Category	Topic	Candidate claim
Accept	The sale of violent video games to minors should be banned	Some researchers believe that while **playing violent video games leads to violent actions**, there are also biological influences that impact a person's choices.
Accept with corrections	Democratic governments should require voters to present photo identification at the polling station	Proponents of a similar law proposed for Texas In March 2009 also argued that **photo identification was necessary to prevent widespread voter fraud**.
Generality failed	Parents should be allowed to genetically screen foetuses for heritable diseases	While psychological stress experienced during a cycle might not influence an IVF outcome, it is possible that **the experience of IVF can result in stress that leads to depression**.
Strength failed	Physical eduaction should be compulsory	**Physical education trends have developed recently to incorporate a greater variety of activities**.
Strength failed	Parents should be allowed to genetically screen a for heritable diseases	However, **the trade-off between risk of birth defect and risk of complications from invasive testing is relative and subjective**; some parents may decide that even a 1:1000 risk of birth defects warrants an invasive test while others wouldn't opt for an invasive test even if they had a 1:10 risk score.
Strength failed	Parents should be allowed to genetically screen foetuses for heritable diseases	This has made international news, and had led to accusations that **many doctors are willing to seriously endanger the health and even life of women in order to gain money**.
Multiple candidates	Wind power should be a primary focus of future energy supply	**The use of wind power reduces the necessity for importing electricity from abroad and strengthens the regional economy**.
Repeats topic	Affirmative action should be used	More recently, a Quinnipiac poll from June 2009 finds that 55% of Americans feel that **affirmative action should be abolished**, yet 55% support affirmative action for disabled people.
Incoherent	Bribery is sometimes acceptable	**The difference with bribery is that this is a tri-lateral relation**.
Incoherent	Parents should be allowed to genetically screen foetuses for heritable diseases	Having this information in advance of the birth means that **healthcare staff as well as parents can better prepare themselves for the delivery of a child with a health problem**.
Not a sentence	A mandatory retirement age should be instituted	**Mandatory retirement is the age** at which persons who hold certain jobs or offices are required by industry custom or by law to leave their employment, or retire.

Table 7: Example sentences for each rejection category

particular the inherently skewed ratio between positive examples and negative examples. We assessed the performance of the proposed approach over a novel benchmark dataset, carefully developed for this task. Our results verify the soundness of our definitions, and the validity of the introduced CDCD task.

In future work we intend to expand the collected labeled data and to generate new versions of this benchmark, that will be further released for academic research. In parallel, we intend to explore various ways to improve the accuracy of our predictions. One intriguing direction, highlighted by examining our data, is the possibility of defining different CDC types. For example, it might be that developing separate classifiers for factual CDCs – like S1 in Table 1, and other classifiers designed to detect more subjective CDCs – like S5 in Table 1, will yield better performance, assuming that each of these two types has a distinguished statistical signature. Similarly, it might be that developing domain-orineted statistical models will further enhance the quality of the CDC predictions.

In this work we analyzed labeled data in which for a given Topic, the relevant articles were manually identified. Combining a CDCD solution with automatic opinion retrieval techniques (Macdonald et al., 2010; Zhang et al., 2007) would be a natural next step towards developing an even more powerful CDCD system. Moreover, while compelling arguments start with high quality and relevant claims, they must include reliable evidence to support the validity of the introduced claims. Thus, combining a CDCD system with a system that automatically detects such supportive evidence , may give rise to a new generation of automatic argumentation methods. In principles, such methods may detect relevant CDCs in some articles, and support these CDCs with evidence detected within other articles, or even within entirely different corpora, ending up with automatically generated arguments, that were never explicitly proposed before in this form by humans. Developing successful solutions for the CDCD task is a fundamental step in pursuing this vision.

References

Ehud Aharoni, Anatoly Polnarov, Tamar Lavee, Daniel Hershcovich, Ran Levy, Ruty Rinott, Dan Gutfreund, and Noam Slonim. 2014. A benchmark dataset for automatic detection of claims and evidence in the context of controversial topics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, First Workshop on Argumentation Mining*. Association for Computational Linguistics, June.

Elena Cabrio and Serena Villata. 2012. Combining textual entailment and argumentation theory for supporting online debates interactions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 208–212, Jeju Island, Korea, July. Association for Computational Linguistics.

I. Dagan, B. Dolan, B. Magnini, and D. Roth. 2009. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 15(04).

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

A. Freeley and D. Steinberg. 2008. *Argumentation and Debate*. Cengage Learning.

Yufan Guo, Anna Korhonen, and Thierry Poibeau. 2011. A weakly-supervised approach to argumentative zoning of scientific documents. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 273–283, Stroudsburg, PA, USA. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Knowledge Discovery and Data Mining*, pages 168–177.

Craig Macdonald, Rodrygo L.T. Santos, Iadh Ounis, and Ian Soboroff. 2010. Blog track research at trec. *SIGIR Forum*, 44(1):58–75, August.

M. C. McCord, J. W. Murdock, and B. K. Boguraev. 2012. Deep parsing in watson. *IBM J. Res. Dev.*, 56(3):264–278, May.

George A. Miller. 1995. Wordnet: A lexical database for english. *COMMUNICATIONS OF THE ACM*, 38:39–41.

Raquel Mochales Palau and Marie-Francine Moens. 2009. Argumentation mining: the detection, classification and structure of arguments in text. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Law (ICAIL 2009),*, pages 98–109. ACM.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Vikas Raykar, Mitesh Khapra, Amrita Saha, Priyanka Agrawal, and Shantanu Godbole. 2014. Subjectivity detection. work in progress.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing With Compositional Vector Grammars. In *ACL*.

Ramakrishnan Srikant and Rakesh Agrawal. 1996. Mining sequential patterns: Generalizations and performance improvements. pages 3–17.

Simone Teufel. 1999. Argumentative zoning: Information extraction from scientific text. Technical report.

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

Adam Wyner, Jodi Schneider, Katie Atkinson, and Trevor J. M. Bench-Capon. 2012. Semi-automated argumentative analysis of online product reviews. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *COMMA*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 43–50. IOS Press.

Wei Zhang, Clement Yu, and Weiyi Meng. 2007. Opinion retrieval from blogs. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 831–840, New York, NY, USA. ACM.

Annotating Argument Components and Relations in Persuasive Essays

Christian Stab[‡] **and Iryna Gurevych**[†‡]
[‡]Ubiquitous Knowledge Processing Lab (UKP-TUDA),
Department of Computer Science, Technische Universität Darmstadt
[†]Ubiquitous Knowledge Processing Lab (UKP-DIPF),
German Institute for Educational Research
http://www.ukp.tu-darmstadt.de

Abstract

In this paper, we present a novel approach to model arguments, their components and relations in persuasive essays in English. We propose an annotation scheme that includes the annotation of claims and premises as well as support and attack relations for capturing the structure of argumentative discourse. We further conduct a manual annotation study with three annotators on 90 persuasive essays. The obtained inter-rater agreement of $\alpha_U = 0.72$ for argument components and $\alpha = 0.81$ for argumentative relations indicates that the proposed annotation scheme successfully guides annotators to substantial agreement. The final corpus and the annotation guidelines are freely available to encourage future research in argument recognition.

1 Introduction

The ability of formulating persuasive arguments is a crucial aspect in writing skills acquisition. On the one hand, well-defined arguments are the foundation for convincing an audience of novel ideas. On the other hand, good argumentation skills are essential for analyzing different stances in general decision making. By automatically recognizing arguments in text documents, students will be able to inspect their texts for plausibility as well as revise the discourse structure for improving argumentation quality. This assumption is supported by recent findings in psychology, which confirm that even general tutorials effectively improve the quality of written arguments (Butler and Britt, 2011). In addition, *argumentative writing support systems* will enable tailored feedback by incorporating argument recognition. Therefore, it could be expected that they provide appropriate guidance for improving argumentation quality as well as the student's writing skills.

An argument consists of several components (i.e. claims and premises) and exhibits a certain structure constituted by argumentative relations between components (Peldszus and Stede, 2013). Hence, recognizing arguments in textual documents includes several subtasks: (1) separating argumentative from non-argumentative text units, (2) identifying claims and premises, and (3) identifying relations between argument components.

There exist a great demand for reliably annotated corpora including argument components as well as argumentative relations (Reed et al., 2008; Feng and Hirst, 2011) since they are required for supervised machine learning approaches for extracting arguments. Previous argument annotated corpora are limited to specific domains including legal documents (Mochales-Palau and Moens, 2008), newspapers and court cases (Reed et al., 2008), product reviews (Villalba and Saint-Dizier, 2012) and online debates (Cabrio and Villata, 2012). To the best of our knowledge, no work has been carried out to annotate argument components and argumentative relations in persuasive essays (section 2). In addition, the reliability of the corpora is unknown, since only few of these works provide holistic inter-rater agreement scores and none a detailed analysis and discussion of inter-rater agreement.

In this work, we introduce a new argument annotation scheme and a corpus of persuasive essays annotated with argument components and argumentative relations. Our primary motivation is to create

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1501–1510, Dublin, Ireland, August 23-29 2014.

a corpus for argumentative writing support and to achieve a better understanding of how arguments are represented in texts. In particular, the contributions of this paper are the following: First, we introduce a novel annotation scheme for modeling arguments in texts. Second, we present the findings of a pre-study and show how the findings influenced the definition of the annotation guidelines. Third, we show that the proposed annotation scheme and guidelines lead to substantial agreement in an annotation study with three annotators. Fourth, we provide the annotated corpus as freely available resource to encourage future research.[1]

2 Related Work

2.1 Previous Argument Annotated Corpora

Currently, there exist only a few corpora that include argument annotations. The work most similar to ours with respect to the annotation scheme is Araucaria (Reed et al., 2008) since it also includes structural information of arguments. It is based on the *Argumentation Markup Language* (AML) that models argument components in a XML-based tree structure. Thus, it is possible to derive argumentative relations between components though they are not explicitly included. In contrast to our work, the corpus consists of several text genres including newspaper editorials, parliamentary records, judicial summaries and discussion boards. In addition, the reliability of the annotations is unknown. Nevertheless, researchers use the corpus for different computational tasks, e.g. separating argumentative from non-argumentative sentences (Mochales-Palau and Moens, 2011), identifying argument components (Rooney et al., 2012) and classifying argumentation schemes (Feng and Hirst, 2011).

Mochales-Palau and Moens (2008) conduct an argument annotation study in legal cases of the *European Court of Human Rights* (ECHR). They experiment with a small corpus of 10 documents and obtain an inter-rater agreemnt of $\kappa = 0.58$. In a subsequent study, they elaborated their guidelines and obtain an inter-rater agreement of $\kappa = 0.75$ on a corpus of 47 documents (Mochales-Palau and Moens, 2011). Unfortunately, the annotation scheme is not described in detail, but it can be seen from the examples that it includes annotations for claims and supporting or refuting premises. Unlike our work, the annotation scheme does not include argumentative relations.

Cabrio and Villata (2012) annotate argumentative relations in debates gathered from *Debatepedia*. Instead of identifying argument components, they are interested in relations between arguments to identify which are the ones accepted by the community. They apply textual entailment for identifying support and attack relations between arguments and utilize the resulting structure for identifying accepted arguments. Therefore, they annotate a pair of arguments as either entailment or not. In contrast to our work, the approach models relationships between pairs of arguments and does not consider components of individual arguments. In addition, the work does not include an evaluation of the annotation's reliability.

Villalba and Saint-Dizier (2012) study argumentation annotation in a corpus of French and English product reviews. Their goal is to identify arguments related to opinion expressions for recognizing reasons of customer opinions. Their annotation scheme is limited to eight types of support (e.g. justification, elaboration, contrast). Compared to our annotation scheme, the work distinguishes between different premise types. However, the approach is tailored to product reviews, and the work does not provide an inter-rater agreement study.

In contrast to previous work, our annotation scheme includes argument components and argumentative relations. Both are crucial for argument recognition (Sergeant, 2013) and argumentative writing support. First, argumentative relations are essential for evaluating the quality of claims, since it is not possible to examine how well a claim is justified without knowing which premises belong to a claim (Sampson and Clark, 2006). Second, methods that recognize if a statement supports or attacks a claim enable the collection of additional evidence from other resources to recommend argument improvement. In addition, we provide a detailed analysis of the inter-rater agreement and an analysis of disagreements.

[1] http://www.ukp.tu-darmstadt.de/data/argumentation-mining

2.2 Persuasive Essays

Persuasive essays are extensively studied in the context of *automated essay grading* (Shermis and Burstein, 2013), which aims at automatically assigning a grade to a student's essay by means of several features. Since the argument structure is crucial for evaluating essay quality, Burstein et al. (1998) propose an approach for identifying the argumentative discourse structure by means of discourse marking. They utilize a surface cue word and phrase lexicon to identify the boundaries of arguments at the sentence level in order to evaluate the content of individual arguments and to enrich their feature set for determining precise grades. Although the identification of argument boundaries is important for argument recognition, our work allows a more fine-grained analysis of arguments since it also includes argument components and argumentative relations.

Madnani et al. (2012) studied persuasive essays for separating organizational elements from content. They argue that the detection of organizational elements is a step towards argument recognition and inferring the structure of persuasive discourse. Further, they refer to organizational elements as claim and premise indicating word sequences which they call *shell expressions*. They annotate 200 essays and estimate an inter-rater agreement of $\kappa = 0.699$ and $F_1 = 0.726$ on a subset of 50 essays annotated by two annotators. However, their annotation scheme is limited to shell expressions and compared to our work it does not include argument components or argumentative relations.

Additional annotation studies on persuasive essays focus on identifying style criteria (Burstein and Wolska, 2003), factual information (Beigman Klebanov and Higgins, 2012), holistic scores for argumentation quality (Attali et al., 2013) or metaphors (Beigman Klebanov and Flor, 2013). We are not aware of an annotation study including argument components and argumentative relations in persuasive essays.

3 Annotation Scheme

The goal of our proposed annotation scheme is to model argument components as well as argumentative relations that constitute the argumentative discourse structure in persuasive essays. We propose an annotation scheme including three argument components and two argumentative relations (figure 1).

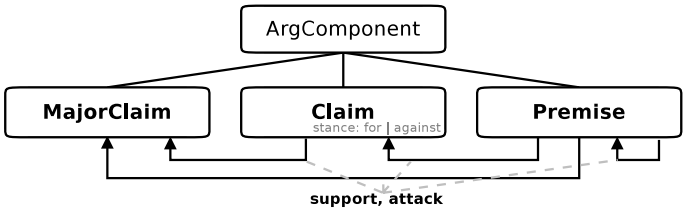


Figure 1: Argument annotation scheme including argument components and argumentative relations indicated by arrows below the components.

3.1 Argument Components

Persuasive essays exhibit a common structure. Usually, the introduction includes a *major claim* that expresses the author's stance with respect to the topic. The major claim is supported or attacked by arguments covering certain aspects in subsequent paragraphs. Sentences (1–3) illustrate three examples of major claims (the major claim is in bold face).[2]

(1) "*I believe that* ***we should attach more importance to cooperation during education***."
(2) "*From my viewpoint,* ***people should perceive the value of museums in enhancing their own knowledge***."
(3) "*Whatever the definition is,* ***camping is an experience that should be tried by everyone***."

In the first example, the author explicitly states her stance towards cooperation during education. The major claims in the second and third example are taken from essays about museums and camping

[2]We use examples from our corpus (5.1) without correcting grammatical or spelling errors.

respectively. In (1) and (2) a *stance indicating expression* (*"I believe"* and *"From my viewpoint"*) denotes the presence of the major claim. Although, these indicators are frequent in persuasive essays, not every essay contains an expression that denotes the major claim. In those cases, the annotators are asked to select the expression that is most representative with respect to the topic and author's stance (cf. (3)).

The paragraphs between introduction and conclusion of persuasive essays contain the actual arguments which either support or attack the major claim.[3] Since argumentation has been a subject in philosophy and logic for a long time, there is a vast amount of argumentation theories which provide detailed definitions of argument components (Toulmin, 1958; Walton et al., 2008; Freemen, 2011).[4] All these theories generally agree that an *argument* consists of several components and that it includes a *claim* that is supported or attacked by at least one *premise*. Examples (4) and (5) illustrate two arguments containing a claim (in bold face) and a premise (underlined).

(4) "***It is more convenient to learn about historical or art items online***. *With Internet, people do not need to travel long distance to have a real look at a painting or a sculpture, which probably takes a lot of time and travel fee.*"
(5) "***Locker checks should be made mandatory and done frequently*** *because they assure security in schools, makes students healthy, and will make students obey school policies.*"

The claim is the central component of an argument. It is a controversial statement that is either true or false and should not be accepted by readers without additional support. The premise underpins the validity of the claim. It is a reason given by an author for persuading readers of the claim. For instance, in (4) the author underpins his claim that Internet usage is convenient for exploring cultural items because of time and travel fee savings. In this example, both components cover a complete sentence. However, a sentence can also contain several argument components like in example (5). Therefore, we do not predefine the boundaries of the expression to be annotated (markable) in advance and annotate each argument as a *statement*, which is a sequence of words that constitutes a grammatically correct sentence.

To indicate if an argument supports or attacks a major claim, we add a *stance attribute* to the claim that denotes the polarity of an argument with respect to the author's stance. This attribute can take the values *for* or *against*. For example, the argument given in (4) refutes the major claim in example (2). Thus, the stance attribute of the claim in (4) is set to *against* in this example.

3.2 Argumentative Relations

Argumentative relations model the discourse structure of arguments in persuasive essays. They indicate which premises belong to a claim and constitute the structure of arguments. We follow the approach proposed by Peldszus and Stede (2013) and define two directed relations between argument components: *support* and *attack*.[5] Both relations can hold between a premise and another premise, a premise and a (major-) claim, or a claim and a major claim (figure 1). For instance, in example (4) the premise in the second sentence is a reason or justification for the claim in the first sentence and the claim in (4) attacks the major claim of example (2). Thus, an argumentative relation between two components indicates that the source component is a reason or a refutation for the target component. The following example illustrates a more complex argument including one claim and three premises.

(6) "***Living and studying overseas is an irreplaceable experience when it comes to learn standing on your own feet***. *One who is living overseas will of course struggle with loneliness, living away from family and friends*$_1$ *but those difficulties will turn into valuable experiences in the following steps of life*$_2$. *Moreover, the one will learn living without depending on anyone else*$_3$."

Figure 2 illustrates the structure of this argument. The claim is attacked by premise$_1$, whereas premise$_2$ is a refutation of premise$_1$. The third premise is another reason that underpins the claim in this paragraph.

[3] In some cases, the introduction or conclusion contains arguments as well, those are also annotated in the annotation study.
[4] A review of argumentation theory is beyond the scope of this paper but a survey can be found in (Bentahar et al., 2010)
[5] Peldszus and Stede also define a *counter-attacking relation* that is omitted in our scheme, since it can also be represented as a chain of attacking premises.

This shows that it is not necessary to explicitly distinguish between supporting and attacking premises, since the relational structure and the type of argumentative relations implicitly denote the role of argument components. Additionally, argumentative relations enable the modeling of relationships between pairs of arguments on the macro level, e.g., by linking claims to the major claim.

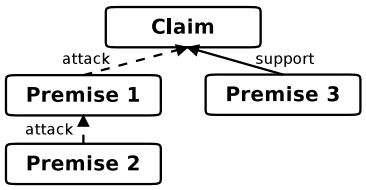


Figure 2: Argumentation structure of example (6)

4 Pre-Study

We conduct a preliminary study to define the annotation guidelines on a corpus of 14 short text snippets (1–2 sentences) that are either gathered from example essays or written by one of the authors. We ask five non-trained annotators to classify each text as argumentative or non-argumentative. If a text is classified as argumentative, the annotators are asked to identify the claim and the premise. In the first task, we obtain an inter-rater agreement of 58.6% and multi-$\pi = 0.171$ (Fleiss, 1971)[6]. We identified the markables for measuring the inter-rater agreement of the second task by manually determining the statements in each of the 14 text snippets. In total, we determined 32 statements and obtained an inter-rater agreement of 55.9% and multi-$\pi = 0.291$. These results indicate a low reliability of the annotations. In addition, they emphasize the demand for a precisely defined argument annotation strategy. In subsequent discussions, we discovered that the primary source of uncertainty is due to the missing context. Since the text snippets are provided without any information about the topic, the annotators found it difficult to decide if a snippet includes an argument or not. In addition, the annotators report that the author's stance might facilitate the separation of argumentative from non-argumentative text and to determine the components of arguments.

According to these findings, we define a new *top-down annotation process* starting with the major claim and drill-down to the claims and premises. Therefore, the annotators are aware of the author's stance after identifying the major claim. In addition, we ask the annotators to read the entire essay in order to identify the topic before starting with the actual annotation task. Although, this approach is more time-consuming than a direct identification of argument components, we show in our annotation study (section 5) that it yields reliably annotated data. In particular, the annotation guidelines consist of the following steps:

1. *Topic and stance identification*: Before starting with the annotation process, annotators identify the topic and the author's stance by reading the entire essay.

2. *Annotation of argument components*: In this step, the major claim is identified either in the introduction or in the conclusion of an essay. Subsequently, annotators identify the claims and premises in each paragraph. We instruct the annotators to annotate each argument component as a statement covering an entire sentence or less. We consolidate the annotations of all annotators before continuing with the next step (section 5.4).

3. *Annotation of argumentative relations*: Finally, the claims and premises are linked within each paragraph, and the claims are linked to the major claim either with a support or attack relation.

[6]Although the coefficient was introduced by Fleiss as a generalization of Cohen's κ (Cohen, 1960), it is actually a generalization of Scott's π (Scott, 1955), since it assumes a cumulative distribution of annotations by all annotators (Artstein and Poesio, 2008). We follow the naming proposed by Artstein and Poesio and refer to the measure as multi-π.

5 Annotation Study

Three annotators participate in the study and annotate the essays independently using our described annotation scheme. We conduct several training sessions after each annotator has read the annotation guidelines. In these sessions, annotators collaboratively annotate 8 example essays for resolving disagreements and obtaining a common understanding of the annotation guidelines. For the actual annotation task, we used the *brat annotation tool* that is freely available.[7] It allows the annotation of text units with arbitrary boundaries as well as the linking of annotations for modeling argumentative discourse structures.

5.1 Data

Our corpus consists of 90 persuasive essays in English, which we selected from *essayforum*[8]. This forum is an active community that provides writing feedback for different kinds of texts. For instance, students post their essays for retrieving feedback about their writing skills while preparing themselves for standardized tests. We randomly selected the essays from the *writing feedback* section of the forum and manually reviewed each essay. Due to the non-argumentative writing style and significant language flaws, we replaced 4 of them during a manual revision of the corpus. The final corpus includes 1,673 sentences with 34,917 tokens. On average, each essay has 19 sentences and 388 tokens.

5.2 Inter-rater Agreement

We evaluate the reliability of the argument component annotations using two strategies. Since there are no predefined markables in our study, annotators have to identify the boundaries of argument components. We evaluate the annotations using Krippendorff's α_U (Krippendorff, 2004). It considers the differences in the markable boundaries of several annotators and thus allows for assessing the reliability of our annotated corpus. In addition, we evaluate if a sentence contains an argument component of a particular category using percentage agreement and two chance-corrected measures: multi-π (Fleiss, 1971) and Krippendorff's α (Krippendorff, 1980). Since only 5.6% of the sentences contain several annotations of different argument components, evaluating the reliability at the sentence-level provides a good approximation of the inter-rater agreement. In addition, it enables comparability with future argument annotation studies that are conducted at the sentence-level. The annotations yield the following class distribution at the token-level: 3.5% major claim, 18.2% claim, 48.1% premise and 30.2% are not annotated. At the sentence-level 5.4% contain a major claim, 26.4% a claim, 61.1% a premise and 19.3% none annotation. Thus, 12.2% of the sentences contain several annotations.

	%	π	α	α_U
MajorClaim	.9827	.8334	.8365	.7726
Claim	.8690	.6590	.6655	.6033
Premise	.8618	.7075	.7131	.7594

Table 1: Inter-rater agreement of argument component annotations

We obtain the highest inter-rater agreement for the annotations of the major claim (table 1). The inter-rater agreement of 98% and multi-$\pi = 0.833$ indicates that the major claim can be reliably annotated in persuasive essays. In addition, there are few differences regarding the boundaries of major claims ($\alpha_U = 0.773$). Thus, annotators identify the sentence containing the major claim as well as the boundaries reliably. We obtain an inter-rater agreement of multi-$\pi = 0.708$ for premise annotations and multi-$\pi = 0.66$ for claims. This is only slightly below the "*tentative conclusion boundary*" proposed by Carletta (1996) and Krippendorff (1980). The unitized α of the major claim and the claim are lower than the sentence-level agreements (table 1). Only the unitized α of the premise annotations is higher compared to the sentence-level agreement. Thus, the boundaries of premises are more precisely identified. The joint unitized measure for all categories is $\alpha_U = 0.724$. Hence, we tentatively conclude that the annotation of argument components in persuasive essays is reliably possible.

[7] http://brat.nlplab.org

[8] http://www.essayforum.com

The agreement of the stance attribute is computed for each sentence. We follow the same methodology as for the computation of the argument component agreement, but treat each sentence containing a claim as either for or against according to the stance attribute (sentences not containing a claim are treated as not annotated, but are included in the markables). Thus, the upper boundary for the stance agreement constitutes the agreement of the claim annotations. The agreement of the stance attribute is only slightly below the agreement of the claim (86%; multi-$\pi = 0.643$; $\alpha = 0.65$). Hence, the identification of either attacking or rebutting claims is feasible with high agreement.

We determine the markables for evaluating the reliability of argumentative relations as the set of all pairs between argument components according to our annotation scheme. So, the markables correspond to all relations that were possible during the annotation task. In total, the markables include $5{,}137$ pairs of which 25.5% are annotated as support relation and 3.1% as attack relations. We obtain an inter-rater agreement above 0.8 for both support and attack relations (table 2) that is considered by Krippendorff (1980) as good reliability. Therefore, we conclude that argumentative relations can be reliably annotated in persuasive essays.

	$\%$	π	α
support	.9267	.8105	.8120
attack	.9883	.8052	.8066

Table 2: Inter-rater agreement of argumentative relation annotations

5.3 Error Analysis

To study the disagreements encountered during the annotation study, we created *confusion probability matrices* (CPM) (Cinková et al., 2012) for argument components and argumentative relations. A CPM contains the conditional probabilities that an annotator assigns a certain category (column) given that another annotator has chosen the category in the row for a specific item. In contrast to traditional confusion matrices, a CPM also enables the evaluation of confusions if more than two annotators are involved in an annotation study.

	Major Claim	Claim	Premise	None
Major Claim	.675	.132	.148	.045
Claim	.025	.552	.338	.086
Premise	.014	.163	.754	.069
None	.012	.123	.204	.660

Table 3: Confusion probability matrix for argument component annotations (Category 'None' indicates argument components that are not identified by an annotator.)

The major disagreement is between claims and premises (table 3). This could be expected since a claim can also serve as premise for another claim, and it is difficult to distinguish these two concepts in the presence of reasoning chains. For instance, examples (7–9) constitute a reasoning chain in which (7) is supported by (8) and (8) is supported by (9):

(7) "*Random locker checks should be made obligatory*."
(8) "*Locker checks help students stay both physically and mentally healthy*."
(9) "*It discourages students from bringing firearms and especially drugs*."

Considering this structure, (7) can be classified as claim. However, if (7) is omitted, (8) becomes a claim that is supported by (9). Thus, the distinction between claims and premises depends not only on the context and the intention of the author but also on the structure of a specific argument. Interestingly, the distinction between major claims and claims is less critical. Apparently, the identification of the major claim is easier since it is directly related to the author's stance in contrast to more general claims that cover a certain aspect with respect to the overall topic of the essay.

The CPM for relations (table 4) reveals that the highest confusion is between support/attack relations and none classified relations. This could be due to the fact that it is difficult to identify the correct target of a relation, especially in the presence of multiple claims or reasoning chains in a paragraph. For instance,

	support	attack	none
support	.750	.013	.238
attack	.104	.691	.205
none	.092	.001	.898

Table 4: Confusion probability matrix for argumentative relation annotations

in the previous example an annotator could also link (9) directly to (7) or even to (7) and (8). In both cases, the argument would be still meaningful. The distinction between support and attack relations does not reveal high disagreements. To sum up, the error analysis reveals that the annotation of argumentative relations yields more reliable results than that of argument components. This could be due to the fact that in our studies, argument components are known before annotating the relations and thus the task is easier. Nevertheless, it could be interesting to annotate relations before classifying the types of argument components and to investigate if it positively influences the reliability of annotations.

5.4 Creation of the Final Corpus

The creation of the final corpus consists of two independent tasks. First, we consolidate the argument components before the annotation of argumentative relations. So each annotator works on the same argumentative components when annotating the relations. Second, we consolidate the argumentative relations to obtain the final corpus. We follow a majority voting in both steps. Thus, an annotation is adopted in the final corpus if at least two annotators agree on the category as well as on the boundaries. In applying this strategy, we observed seven cases for argument components and ten cases for argumentative relations that could not be solved by majority voting. Those cases were discussed in the group of all annotators to reach an agreement. Table 5 shows an overview of the final corpus. It includes 90 major

	ALL	avg. per essay	standard deviation
Sentence	1,673	19	7
Tokens	34,917	388	124
MajorClaim	90	1	0
Claim	429	5	2
Claim (for)	365	4	2
Claim (against)	64	1	1
Premises	1,033	11	6
support	1,312	15	7
attack	161	2	2

Table 5: Statistics of the final corpus

claims (each essay contains exactly one), 429 claims and 1,033 premises. This proportion between claims and premises is common in argumentation and confirms the findings of Mochales-Palau and Moens (2011, p. 10) that claims are usually supported by several premises for "*ensuring a complete and stable standpoint*".

6 Conclusion & Future Work

We presented an annotation study of argument components and argumentative relations in persuasive essays. Previous argument annotation studies suffer from several limitations: Either they do not follow a systematic methodology and do not provide detailed inter-rater agreement studies or they do not include annotations of argumentative relations. Our annotation study is the first step towards computational argument analysis in educational applications that provides both annotations of argumentative relations and a comprehensive evaluation of the inter-rater agreement. The results of our study indicate that the annotation guidelines yield substantial agreement. The resulting corpus and the annotation guidelines are freely available to encourage future research in argument recognition.

In future work, we plan to utilize the created corpus as training data for supervised machine learning methods in order to automatically identify argument components as well as argumentative relations. In addition, there is a demand to scale the proposed annotation scheme to other genres e.g. scientific articles or newspapers and to create larger corpora.

Acknowledgements

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806. We thank Piyush Paliwal and Krish Perumal for their valuable contributions and we thank the anonymous reviewers for their helpful comments.

References

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Yigal Attali, Will Lewis, and Michael Steier. 2013. Scoring with the computer: Alternative procedures for improving the reliability of holistic essay scoring. *Language Testing*, 30(1):125–141.

Beata Beigman Klebanov and Michael Flor. 2013. Argumentation-Relevant Metaphors in Test-Taker Essays. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 11–20, Atlanta, GA, USA.

Beata Beigman Klebanov and Derrick Higgins. 2012. Measuring the use of factual information in test-taker essays. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 63–72, Montreal, Quebec, Canada.

Jamal Bentahar, Bernard Moulin, and Micheline Bélanger. 2010. A taxonomy of argumentation models used for knowledge representation. *Artificial Intelligence Review*, 33(3):211–259.

Jill Burstein and Magdalena Wolska. 2003. Toward evaluation of writing style: finding overly repetitive word use in student essays. In *Proceedings of the tenth conference of European chapter of the Association for Computational Linguistics*, EACL '03, pages 35–42, Budapest, Hungary.

Jill Burstein, Karen Kukich, Susanne Wolff, Ji Lu, and Martin Chodorow. 1998. Enriching Automated Essay Scoring Using Discourse Marking. In *Proceedings of the Workshop on Discourse Relations and Discourse Markers*, pages 15–21, Montreal, Quebec, Canada.

Jodie A. Butler and M. Anne Britt. 2011. Investigating Instruction for Improving Revision of Argumentative Essays. *Written Communication*, 28(1):70–96.

Elena Cabrio and Serena Villata. 2012. Natural language arguments: A combined approach. In *Proceedings of the 20th European Conference on Artificial Intelligence*, ECAI '12, pages 205–210, Montpellier, France.

Jean Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254.

Silvie Cinková, Martin Holub, and Vincent Kríž. 2012. Managing uncertainty in semantic tagging. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 840–850, Avignon, France.

Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Vanessa Wei Feng and Graeme Hirst. 2011. Classifying arguments by scheme. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 987–996, Portland, OR, USA.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

James B. Freemen. 2011. *Argument Structure: Representation and Theory*, volume 18 of *Argumentation Library*. Springer.

Klaus Krippendorff. 1980. *Content Analysis: An Introduction to its Methodology*. Sage.

Klaus Krippendorff. 2004. Measuring the Reliability of Qualitative Text Analysis Data. *Quality & Quantity*, 38(6):787–800.

Nitin Madnani, Michael Heilman, Joel Tetrault, and Martin Chodorow. 2012. Identifying High-Level Organizational Elements in Argumentative Discourse. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 20–28, Montreal, Quebec, Canada.

Raquel Mochales-Palau and Marie-Francine Moens. 2008. Study on the Structure of Argumentation in Case Law. In *JURIX the twenty-first annual conference on legal knowledge and information systems*, pages 11–20, Florence, Italy.

Raquel Mochales-Palau and Marie-Francine Moens. 2011. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22.

Andreas Peldszus and Manfred Stede. 2013. From Argument Diagrams to Argumentation Mining in Texts: A Survey. *International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)*, 7(1):1–31.

Chris Reed, Raquel Mochales-Palau, Glenn Rowe, and Marie-Francine Moens. 2008. Language resources for studying argument. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation*, LREC '08, pages 2613–2618, Marrakech, Morocco.

Niall Rooney, Hui Wang, and Fiona Browne. 2012. Applying kernel methods to argumentation mining. In *Proceedings of the Twenty-Fifth International Florida Artificial Intelligence Research Society Conference*, FLAIRS '12, pages 272–275, Marco Island, FL, USA.

Victor D. Sampson and Douglas B. Clark. 2006. Assessment of argument in science education: A critical review of the literature. In *Proceedings of the 7th International Conference on Learning Sciences*, ICLS '06, pages 655–661, Bloomington, IN, USA.

William A. Scott. 1955. Reliability of Content Analysis: The Case of Nominal Scale Coding. *Public Opinion Quarterly*, 19(3):321–325.

Alan Sergeant. 2013. Automatic argumentation extraction. In *Proceedings of the 10th European Semantic Web Conference*, ESWC '13, pages 656–660, Montpellier, France.

Mark D. Shermis and Jill Burstein. 2013. *Handbook of Automated Essay Evaluation: Current Applications and New Directions*. Routledge Chapman & Hall.

Stephen E. Toulmin. 1958. *The uses of Argument*. Cambridge University Press.

Maria Paz Garcia Villalba and Patrick Saint-Dizier. 2012. Some facets of argument mining for opinion analysis. In *Proceeding of the 2012 conference on Computational Models of Argument*, COMMA '12, pages 23–34, Vienna, Austria.

Douglas Walton, Chris Reed, and Fabrizio Macagno. 2008. *Argumentation Schemes*. Cambridge University Press.

Building a Hierarchically Aligned Chinese-English Parallel Treebank

Dun Deng and **Nianwen Xue**

Computer Science Department, Brandeis University
415 South Street, Waltham MA, USA
ddeng@brandeis.edu, xuen@brandeis.edu

Abstract

We construct a hierarchically aligned Chinese-English parallel treebank by manually doing word alignments and phrase alignments simultaneously on parallel phrase-based parse trees. The main innovation of our approach is that we leave words without a translation counterpart (which are mostly language-particular function words) unaligned on the word level, and locate and align the appropriate phrases which encapsulate them. In doing so, we harmonize word-level and phrase-level alignments. We show that this type of annotation can be performed with high inter-annotator consistency and have both linguistic and engineering potentials.

1 Introduction

The value of human annotated syntactic structures for Statistical Machine Translation has been clearly demonstrated in string-to-tree (Galley et al., 2004; Galley et al., 2006; Huang et al., 2006), tree-to-string (Liu et al., 2006; Liu and Gildea, 2008), and tree-to-tree (Eisner, 2003; Liu et al., 2009; Chiang, 2010) models. One recurring issue which hampers the utility of syntactic structures is the incompatibility between word alignments and syntactic structures (Denero and Klein, 2007; Fossum et al., 2008; Pauls et al., 2010). The incompatibility arises because word alignments and syntactic structures are established independently of each other. In the case of tree-to-tree models, there is also the issue of incompatible parallel tree structures resulting from divergent syntactic annotation standards that have been independently conceived based on monolingual corpora (Chiang, 2010). In this paper, we report an effort in building a Hierarchically Aligned Chinese-English Parallel Treebank (HACEPT) where we manually do word-level and phrase-level alignments simultaneously on parallel phrase-based parse trees. In this process, we attempt to establish an annotation standard that harmonizes word-level and phrase-level alignments. We also analyze a common incompatibility issue between Chinese-English parallel parse trees exposed in the annotation process, with the goal of solving the issue by semi-automatically revising the trees.

In the rest of this paper, we describe how we construct the HACEPT and discuss issues arising in the construction process. In Section 2, we discuss the problems of word alignment done without considering its interaction with syntactic structures. In Section 3, we describe our annotation procedure where we perform word-level and phrase-level alignments simultaneously in a coordinated manner, and show how our approach is free of the problems discussed in Section 2. In Section 4, we report a common incompatibility issue between parse trees and propose a solution. We also compare the issue with translation divergence (Dorr, 1994) and show that they are different in nature and occurrence frequency. In Section 5, we present the results of two experiments we have done on our annotation to show the intuitiveness of our approach and the linguistic and engineering potentials of our corpus. We then describe related work in Section 6 and conclude our paper in Section 7.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1511–1520, Dublin, Ireland, August 23-29 2014.

2 Incompatibilities between word alignments and syntactic structures

All the existing word alignment practice we know of treats word alignment as a stand-alone task without systematically considering its interaction with the syntactic structure of a sentence. The inevitable consequence of the practice is that both redundancies and incompatibilities between word alignments and syntactic structures will arise in many places. In this section, we illustrate the issues through language-particular function words, where the problems are most frequently found. Due to language-particular idiosyncrasy and lack of lexical content, these function words usually do not have a translation counterpart, which presents a great challenge to alignment annotation. There are two logical possibilities of dealing with these words, both of which are represented in existing annotation practice. The first is to leave them unaligned or link them to a fictitious NULL word (Ahrenberg, 2007; Brown et al., 1990), and the second, which also seems to be the more common pratice, is to attach these function words to a word that has a translation counterpart, and then align the function word and its host with the counterpart of the host (Melamed, 1998; Li et al., 2009). For ease of discussion, below we will refer to the latter pratice as the "glue-to-a-host" strategy (GTAHS). Both approaches are less than desirable: the former leaves the function words unaccounted for, and the latter leads to incompatibility issues we discuss in detail below.

First note that, by attaching language-particular function words to a host, the GTAHS creates redundancies between word alignments and syntactic structures since many of these function words have already been associated with a host within a constituent in the parse tree (e.g., the English determiner *the* is placed inside the projection of its host, namely an NP). A more serious issue is that the GTAHS creates spurious ambiguities. Lexical ambiguity is inevitable in translation. For instance, the English noun *bank* has more than one lexical meaning and each of the meanings corresponds to a different Chinese word. That fact aside, the GTAHS creates spurious ambiguities, which, in our view, would be harmful to Machine Translation (MT) if extracted as translation rules. Consider the following example, where the Chinese noun 苹果 is aligned to six English strings (aligned elements are underlined):

(1) a. eat apples <> 吃 苹果
b. eat an apple <> 吃 苹果
c. eat the apple <> 吃 苹果
d. fond of apples <> 喜欢 苹果
e. talk about apples <> 谈论 苹果
f. provide them with apples <> 给 他们 苹果

The English *apple* and the Chinese 苹果 match in meaning and are both unambiguous. In cases where the English noun is used with a determiner as in (1b) and (1c), since Chinese has no determiners and the bare noun 苹果 can be the appropriate translation for either *an apple* or *the apple* given a context, the GTAHS attaches the determiner to *apple* and the whole string is aligned with 苹果. In other similar cases where an English element such as a preposition is absent in Chinese as in (1d), (1e) and (1f), the GTAHS glues the preposition to *apple* and the whole PP is aligned with 苹果. With the GTAHS, the unambiguous Chinese 苹果 ends up being aligned with more than one English string. This kind of spurious ambiguity is very common given the GTAHS.

The second issue is that, by attaching function words to a host, the GTAHS effectively creates rudimentary syntactic structures, which are often incompatible with the syntactic structures annotated based on existing treebanking annotation standards. For example, all the aligned multi-word strings underlined in (2) do not correspond to a constituent in a Penn TreeBank (Marcus et al., 1993) or Chinese TreeBank (Xue et al., 2005) parse tree:

(2) a. If I were him <> 如果 我 是 他 的话
b. He is visiting Beijing <> 他 正 访问 北京

c. the beginning of the new year <> 新 年 伊始

d. to quickly and efficiently solve the problem <> 迅速 有效地 解决 问题

Given the incompatibilities between existing word alignments and syntactic structures, in the next section we describe an approach where we perform word-level and phrase-level alignments simultaneously on parallel phrase-based parse trees, attempting to construct a hierarchically aligned corpus where word alignments are harmonized with syntactic structures.

3 Annotation specification and procedure

The data we annotate is the Chinese-English portion of the Parallel Aligned Treebank (PAT) described in (Li et al., 2012). Our data consists of two batches, one of which is weblogs and the other of which is postings from online discussion forums. The English sentences in the data set are annotated based on the original Penn TreeBank (PTB) annotation stylebook (Bies et al., 1995) as well as its extensions (Warner et al., 2004), while the Chinese sentences in the data set are annotated based on the Chinese TreeBank (CTB) annotation guidelines (Xue and Xia, 2000) and its extensions (Zhang and Xue, 2012). The PAT only has word alignments, which are done under the GTAHS, and no phrase alignments.

The main departure of our approach is that we loosen the requirement that every word in a sentence pair needs to be word-aligned. On the word level, we only align words that have an equivalent in terms of lexical meaning and grammatical function. For words that do not have a translation counterpart, we leave them unaligned and locate the appropriate phrases in which they appear to be aligned. This way, we eliminate both the redundancies and spurious ambiguities discussed in Section 2. Since phrase alignment is done between syntactic nodes on parallel parse trees, we also eliminate the incompatibilities between word alignments and syntactic structures. See the discussion of the concrete example in Figure 1 below to see the points made here.

Next we discuss our annotation procedure in detail. Our annotators are presented with sentence pairs that come with parallel parse trees. The task of the annotator is to decide, first on the word level and then on the phrase level, if a word or phrase needs to be aligned at all, and if so, to which word or phrase it should be aligned. The decisions about word alignment and phrase alignment are not independent, and must obey well-formedness constraints as outlined in (Tinsley et al., 2007):

a. A non-terminal node can only be aligned once.
b. if Node n_c is aligned to Node n_e, then the descendants of n_c can only be aligned to descendants of n_e.
c. if Node n_c is aligned to Node n_e, then the ancestors of n_c can only be aligned to ancestors of n_e.

This means that once a word alignment is in place, it puts constraints on phrase alignments. A pair of non-terminal nodes (n_c, n_e) cannot be aligned if a word that is a descendant of n_c is aligned to a word that is not a descendant of n_e on the word level.

Let us use the concrete example in Figure 1 to illustrate the annotation process, which is guided by a set of detailed annotation guidelines. On the word level, only those words that are connected with a dashed line are aligned since they have equivalents. Note that the Chinese words 把 (a function word used to prepose the object to the left of the verb), 这样 (an adverb meaning "this way"), 可 (a modal meaning "can") and the English discourse connective *so that*, the auxiliary verb *is* and the preposition *from* are all left unaligned on the word level. Aligning these function words will generate artificial ambiguous cases and create incompatibilities between word alignments and parse trees that have already been illustrated and discussed in Section 2. For instance, if 把 is to be word-aligned, it would be glued to the noun 重力 and the whole string 把 重力 will be aligned to the English *gravity*. Note that both 重力 and *gravity* are unambiguous and form a one-to-one correspondence. With the word alignment between 把 重力 and *gravity*, we make the unambiguous *gravity* correspond to both 重力 and 把 重力 (and possibly

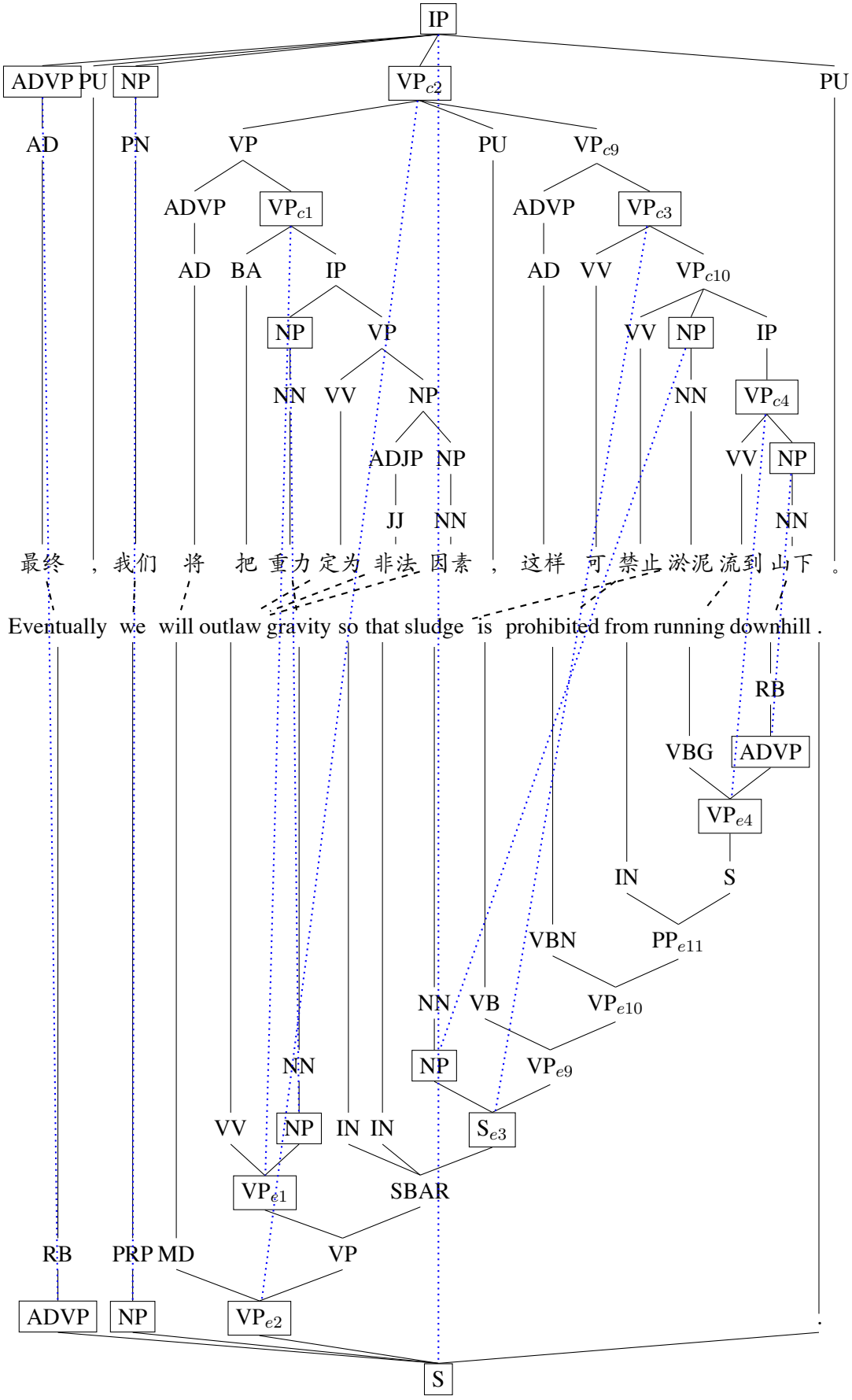


Figure 1: A hierarchically aligned sentence pair

more strings), thus creating a spurious ambiguity. Also note that the string 把 重力 does not form a constituent in the Chinese parse tree, so the word alignment is incompatible with the syntactic structure of the sentence. By leaving 把 unaligned, we avoid both the spurious ambiguity and the incompatibility.

With word alignments in place, next the annotator needs to perform phrase alignments. Note that word alignments place restrictions on phrase alignments. For instance, e_9 and e_{10} will be ruled out as possible alignments for c_{10}, because 淤泥, a descendant of c_{10}, is aligned to *sludge*, which is not a descendant of either e_9 or e_{10}. By contrast, e_3 is a possible alignment for c_{10} because the alignment does not violate the well-formedness constraints. The annotator then needs to decide whether this possible phrase alignment can be actually made. This is a challenging task since, for a given phrase, there usually are more than one candidate from which a single alignment needs to be picked. For instance, for e_3, there are in total three possible phrase alignments, namely c_{10}, c_3 and c_9, all of which obey the well-formedness constraints. Since a non-terminal node is not allowed to be aligned to multiple non-terminal nodes on the other side, the annotator needs to choose one among all the candidates. This highlights the point that the alignment of non-terminal nodes cannot be deterministically inferred from the alignment of terminal nodes. This is especially true given our approach where some terminal nodes are left unaligned on the word level. For instance, the reason why c_9 is a possible alignment for e_3 is because the word 这样 is left unaligned. If 这样 were aligned with *so that*, c_9 could not be aligned with e_3 since *so that* is not a descendant of e_3 and aligning the two nodes will violate Constraint *b*.

While Constraints *b* and *c* can be enforced automatically given the word alignments, the decisions regarding the alignment of non-terminal nodes which satisfy Constraint *a* are based on linguistic considerations. One key consideration is to determine which non-terminal nodes encapsulate the grammatical relations signaled by the unaligned words so that the alignment of the non-terminal nodes will effectively capture the unaligned words in their syntactic context. When identifying non-terminal nodes to align, we follow two seemingly conflicting general principles:

- Phrase alignment should not sever key dependencies involving the grammatical relation signaled by an unaligned word.
- Phrase alignment should be minimal, in the sense that the phrase alignment should contain only the elements involved in the grammatical relation, and nothing more.

The first principle ensures that the grammatical relation is properly encapsulated in the aligned non-terminal nodes. For example in Figure 1, if we attach the English preposition *from* to *running* and aligning them to 流到, we would fail to capture the fact that *from* signals a relation between *prohibit* and *running downhill*. Aligning VP_{c3} with S_{e3} captures this relation.

The first principle in and of itself is insufficient to produce desired alignment. Taken to the extreme, it can be trivially satisfied by aligning the two root nodes of the sentence pair. We also need the alignment to be minimal, in the sense that aligned non-terminal nodes should contain only the elements involved in the grammatical relation, and nothing more. These two requirements used in conjunction ensure that a unique phrase alignment can be found for each unaligned word. The phrase alignments (VP_{c1}, VP_{e1}), (VP_{c2}, VP_{e2}), (VP_{c3}, S_{e3}), as illustrated in Figure 1, all satisfy these two principles.

In addition to making phrase alignments, the annotator needs to assign labels to phrase alignments. We have four labels that are designed along two dimensions: the presence/absence of word order difference and the presence/absence of unaligned function words. The name and definition of each of the four labels are listed below, and an example for each label is given in Figure 2:

a REO, reordering that does not involve unaligned function words (Figure 2a)
b UFW, unaligned function words (Figure 2b)
c REU, reordering that also involves unaligned function words (Figure 2c)
d STD, structural divergence due to cross-linguistic differences (Figure 2d)

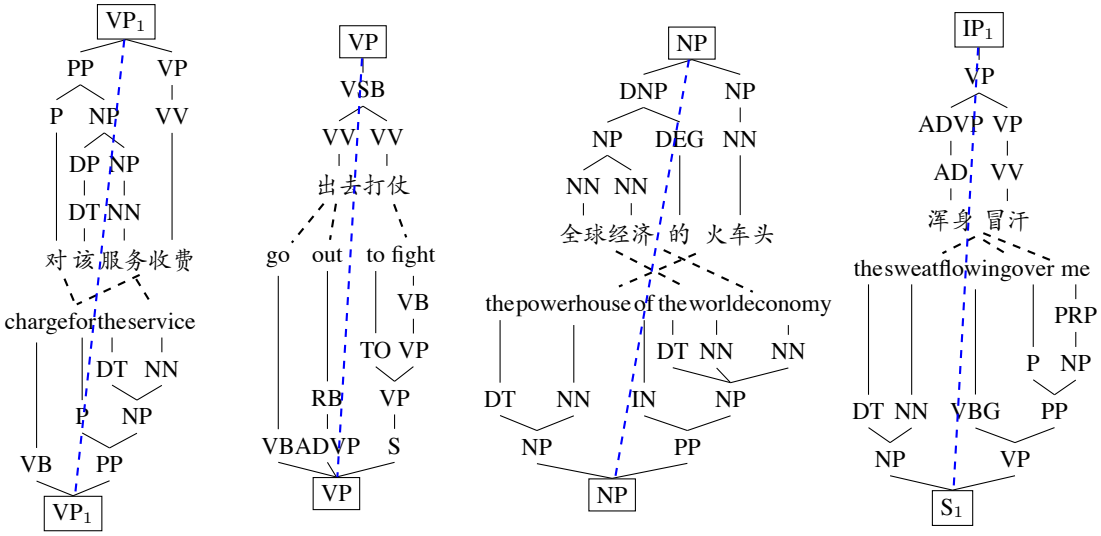


Figure 2: Phrase alignment types

Figure 2a is an example where there is a reordering of the immediate children of the aligned VP nodes. This is a very typical word order difference between Chinese and English. In Chinese, the PP modifier is before the verb while in English the PP modifier is after the verb. The phrase alignment illustrated by Figure 2b has an unaligned function word, namely the English infinitive marker *to*, which has no counterpart in Chinese. There are both reordering (difference in the relative order of *powerhouse* and *economy*) and unaligned function words (Chinese 的 and English *of*) in the phrase alignment in Figure 2c. Figure 2d provides an example where the aligned phrases have structural divergence caused by cross-linguistic differences between Chinese and English, which we will discuss in some detail in Section 4.

4 A common incompatibility issue between parse trees

During the annotation process, we encountered some incompatibility issues between parse trees. For a comprehensive and detailed discussion of the issues, see (Deng and Xue, 2014). Here we report the most common issue, which is caused by differences between treebank annotation guidelines. As already mentioned, the English parse trees we use are annotated based on the original PTB annotation stylebook (Bies et al., 1995) as well as its extensions (Warner et al., 2004), while the Chinese parse trees are annotated based on the CTB annotation guidelines (Xue and Xia, 2000) and its extensions (Zhang and Xue, 2012). Since PTB and CTB are independently annotated, there are some differences in how certain structures are annotated. The main issue is that certain structures are so flat as to make some nodes that should be aligned impossible to be aligned. In general, our alignment task favors deeper structures over shallower ones so that the annotator can have more choices. This is an issue for both Chinese and English parse trees. To get a concrete idea of the issue, take a look at Figure 3.

As shown by Figure 3, VP_{c1} and the English string *probably decrease rapidly with distance*, and VP_{e1} and the Chinese string 随 距离 而 快速 减少, cannot be aligned although they match in meaning and should be aligned. They cannot be aligned because there is no node for either of the two strings in the respective parse tree. Note that the incompatibility between the two trees here is due to a difference in annotation style but not a deep cross-linguistic difference. Both PTB and CTB simplified the annotation task by making the tree structures flatter to increase annotation speed, but the simplification does not always come from the same places. The consequence of these annotation decisions is that relevant structures are sometimes incompatible, which has negatively affected their utility for MT purposes (Chiang,

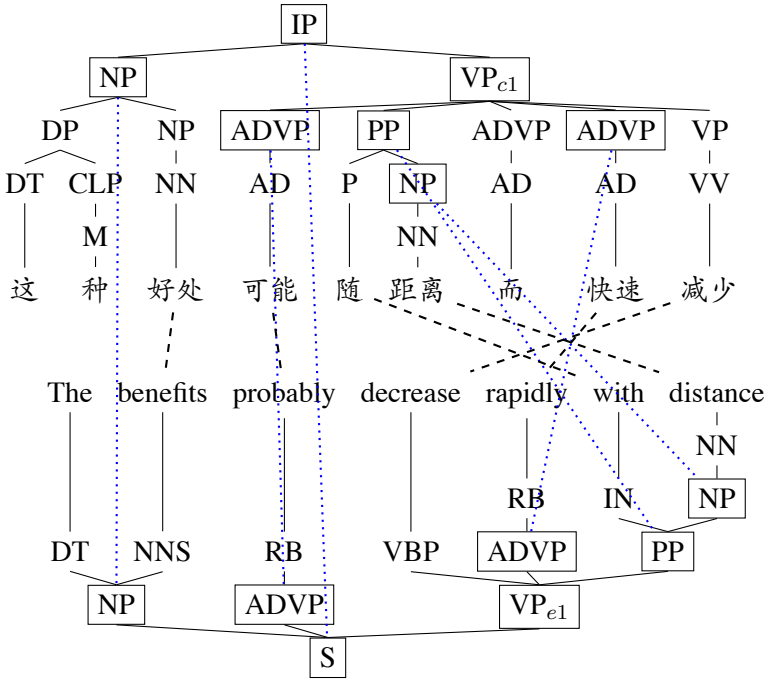


Figure 3: Unalignable nodes due to differences in tree representation

2010).

To solve this incompatibility issue, we need to create more structures through binarization, which can be done automatically. Still take Figure 3 for instance, on the English side, if we create a new VP by combining VP$_{e1}$ and its sister ADVP, the resulting VP can be aligned with VP$_{c1}$. On the Chinese side, if we do binarization to create a VP that dominates the string 随 距离 而 快速 减少, VP$_{e1}$ would have an alignment. Since changing tree structures has the potential risk of causing inconsistency with parse trees in the original treebanks and had better be done systematically after all the annotation is finished, we have not done binarization as of the writing of this paper. For the time being, we assign the label UA (short for Unalignable Node) to nodes which should be aligned but cannot be aligned so that we can gather some statistics on the extent of the problem. We will come back to revisit the nodes carrying UA such as VP$_{c1}$ and VP$_{e1}$ by proposing systematic changes to the original treebanks.

The UA case discussed above should not be confused with another case of incompatibility, namely structural divergence between parallel sentences in translation (Dorr, 1994). As shown above, UA is basically an artificial issue that is caused by difference in parsing guideline design and fixable through automatic binarization. Structural divergence arises mainly due to genuine cross-linguistic differences. We provide an example of structural divergence (STD) in Figure 2d. As shown in the figure, the two aligned phrases (VP and S) are structurally quite different: the English string is a clause with the NP *the sweat* as the subject and the VP *flowing over me* as the predicate (the example is taken out of the sentence *I felt the sweat flowing over me* to save space). The Chinese string is a simple verb phrase where the adverb 浑身 (literally whole-body) modifies the verb 冒汗 (literally emerge-sweat). In terms of meaning correspondence, 浑身 expresses the meaning of the English PP *over me* and the verb matches in meaning with *the sweat flowing*. We have run an experiment on STD and found that the STD cases are pretty rare (on average 5 instances in a file with 500 sentence pairs), indicating that the structural difference between Chinese and English is not so fundamental as to make a big impact on alignment annotation.

5 Annotation experiments

We did two experiments on our annotation. The first is about inter-annotator agreement (IAA), which is a way of both evaluating the annotation quality and judging the intuitiveness of the annotation task. An unintuitive annotation task would force the annotator to make subjective choices, which would result in low IAA. Since the annotation task involves parse trees, ideally we need annotators who are trained in syntax, but that would put a constraint on the pool of qualified annotators and make it difficult for the annotation to scale up. In our annotation experiments, we use four annotators who are fluent in both English and Chinese but have no prior linguistic training, led by a syntactician who performs the final adjudication.

As of this writing, we have completed the single annotation of 8,932 sentence pairs, 2,500 of which are double annotated. The IAA statistics presented in Table 1 are based on the double-annotated 2,500 sentence pairs, which are divided into 5 chunks of 500 sentence pairs each. The statistics are for phrase alignment only, and the micro-average for the 5 chunks is 0.87 (F1), indicating we are able to get good quality annotation for this task. In addition, the agreement statistics for the 5 chunks are very stable, even though they are performed by different pairs of annotators, indicating we are getting consistent annotation from different annotators.

Table 2 shows the result of the second experiment, namely the distribution of the different types of phrase alignment. It shows that alignments that contain unaligned function words outnumber those that do not, and that alignments that do not involve reordering outnumber those that do. It also shows that an overwhelming number of alignments that involve reordering also have unaligned function words. This means that the function words are potentially useful "triggers" for reordering, which is an important issue that MT systems are trying to address.

Chunk No.	precision	recall	F1-measure
1	0.91	0.86	0.89
2	0.92	0.80	0.86
3	0.89	0.89	0.89
4	0.88	0.88	0.88
5	0.89	0.89	.086
micro-average	0.90	0.85	0.87

Table 1: Statistics of IAA

	Annotator	+UFW	-UFW	total
+REO	1	6,473	379	6,852
	2	6,670	379	7,049
-REO	1	7,328	6,872	14,200
	2	7,797	7,334	15,131
total	1	13,801	7,251	21,052
	2	14,467	7,713	22,180

Table 2: Statistics of phrase alignment by types

6 Related work

Parallel treebanks are not something new. However, most of the existing parallel treebanks (Li et al., 2012; Megyesi et al., 2010) do not have phrase alignments. Some (Sulger et al., 2013; Kapanadze, 2012) do have phrase alignments, but neither discussion about the interaction between word-level and phrase-level alignments nor report of IAA is provided. There have been a few recent attempts at automatically aligning subtrees (comparable to our phrases) in the context of MT research, and the automatic alignments are evaluated against a small manually aligned data set. For example, (Tinsley et al., 2007) evaluated an unsupervised algorithm on 810 parsed English-French pairs annotated with subtree alignment. (Xiao and Zhu, 2013) also developed unsupervised subtree alignment methods (EM and Variational Bayes) and evaluated their automatic alignment model on 637 sentences from the Chinese TreeBank (and use the other 99 for tuning). (Sun et al., 2010b; Sun et al., 2010a) also report work on aligning subtrees and evaluate their impact on MT. However, we are not aware of any attempt to systematically harmonize word alignment with the alignment of phrases, or subtrees, or to systematically study the incompatibilities between parallel parse trees.

7 Conclusion

In this paper we report our effort on the construction of a Chinese-English parallel treebank with both word-level and phrase-level alignments. When constructing the treebank, we systematically consider the interaction between word alignments and phrase alignments, and try to harmonize the two kinds of alignments by removing redundancies and incompatibilities between them. We show that this type of annotation can be performed with high inter-annotator consistency. Given our intention for the treebank to be a resource for MT, the next step is to synchronize the parallel parse trees, and of course, to perform automatic hierarchical alignment experiments and MT experiments.

Acknowledgements

This work is supported by the IBM subcontract No. 4913014934 under DARPA Prime Contract No. 0011-12-C-0015 entitled "Broad Operational Language Translation". We would like to thank Libin Shen and Salim Roukos for their inspiration and discussion during early stages of the project, Abe Ittycheriah and Niyu Ge for their help with setting up the data, Loretta Bandera for developing and maintaining the annotation tool, and three anonymous reviewers for their helpful comments. We are grateful for the hard work of our four annotators: Hui Gao, Shiman Guo, Tse-ming Wang and Lingya Zhou. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor or any of the people mentioned above.

References

Lars Ahrenberg. 2007. LinES: An English-Swedish parallel Treebank. In *Proceedings of Nodalida 2007*, pages 270--273, Tartu, Estonia.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995. Bracketing guidelines for Treebank II style Penn Treebank project. Technical report, University of Pennsylvania.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79--85.

David Chiang. 2010. Learning to translate with source and target syntax. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1443--1452.

John Denero and Dan Klein. 2007. Tailoring word alignments to syntactic machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 17--24.

Dun Deng and Nianwen Xue. 2014. Aligning Chinese-English parallel parse trees: is it feasible? In *Proceedings of the 8th Linguistic Annotation Worshop (the LAW VIII)*.

Bonnie J. Dorr. 1994. Machine translation divergences: a formal description and proposed solution. *Computational Linguistics*, 20(4):597--633.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 205--208.

Victoria Fossum, Kevin Knight, and Steven Abney. 2008. Using syntax to improve word alignment precision for syntax-based machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 44--52.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *HLT-NAACL*, pages 273--280.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 961--968.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, pages 66--73.

Oleg Kapanadze. 2012. Building parallel Treebanks for the lesser-resourced languages. Technical report, Universität des Saarlandes.

Xuansong Li, Niyu Ge, and Stephanie Strassel. 2009. Tagging guidelines for Chinese-English word alignment. Technical report, Linguistic Data Consortium.

Xuansong Li, Stephanie Strassel, Stephen Grimes, Safa Ismael, Mohamed Maamouri, Ann Bies, and Nianwen Xue. 2012. Parallel Aligned Treebanks at LDC: New Challenges Interfacing Existing Infrastructures. In *Proceedings of LREC-2012*, Istanbul, Turkey.

Ding Liu and Daniel Gildea. 2008. Improved tree-to-string transducer for machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 62--69.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 609--616.

Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 558--566.

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

Beata Megyesi, Bengt Dahlqvist, Eva A. Csato, and Joakim Nivre. 2010. The English-Swedish-Turkish Parallel Treebank. In *Proceedings of LREC-2010*, Valletta, Malta.

I. Dan Melamed. 1998. Annotation style guide for the Blinker project. Technical report, University of Pennsylvania.

Adam Pauls, Dan Klein, David Chiang, and Kevin Knight. 2010. Unsupervised syntactic alignment with inversion transduction grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 118--126.

Sebastian Sulger, Miriam Butt, Tracy Holloway King, and Paul Meurer et.al. 2013. ParGramBank: The ParGram Parallel Treebank. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 550--560.

Jun Sun, Min Zhang, and Chew Lim Tan. 2010a. Discriminative induction of sub-tree alignment using limited labeled data. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1047--1055.

Jun Sun, Min Zhang, and Chew Lim Tan. 2010b. Exploring syntactic structural features for sub-tree alignment using bilingual tree kernels. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 306--315.

John Tinsley, Ventsislav Zhechev, Mary Hearne, and Andy Way. 2007. Robust language pair-independent subtree alignment. In *Proceedings of Machine Translation Summit XI*.

Colin Warner, Ann Bies, Christine Brisson, and Justin Mott. 2004. Addendum to the Penn Treebank II style bracketing guidelines: BioMedical Treebank annotation. Technical report, University of Pennsylvania.

Tong Xiao and Jingbo Zhu. 2013. Unsupervised sub-tree alignment for tree-to-tree translation. *Journal of Artificial Intelligence Research*, 48:733--782.

Nianwen Xue and Fei Xia. 2000. The bracketing guidelines for Penn Chinese Treebank project. Technical report, University of Pennsylvania.

Nianwen Xue, Fei Xia, Fu dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207--238.

Xiuhong Zhang and Nianwen Xue. 2012. Extending and scaling up the chinese treebank annotation. In *Proceedings of the Second CIPS-SIGHAN Joint Conference on Chinese Language Processing*.

3arif: A Corpus of Modern Standard and Egyptian Arabic Tweets Annotated for Epistemic Modality Using Interactive Crowdsourcing

Rania Al-Sabbagh[†], **Roxana Girju**[†], **Jana Diesner**[‡]
[†]Department of Linguistics and Beckman Institute
[‡]School of Library and Information Science
University of Illinois at Urbana-Champaign, USA
{alsabba1, girju, jdiesner} @illinois.edu

Abstract

We present *3arif*[1], a large-scale corpus of Modern Standard and Egyptian Arabic tweets annotated for epistemic modality[2]. To create *3arif*, we design an interactive crowdsourcing annotation procedure that splits up the annotation process into a series of simplified questions, dispenses with the requirement for expert linguistic knowledge and captures nested modality triggers and their attributes semi-automatically.

1 Introduction

Epistemic modality, according to Palmer (2001), defines the speaker's subjective knowledge, beliefs and judgments about the world's states of affairs. Epistemic modality is used as a linguistic feature for multiple NLP tasks and applications, including sentiment analysis (Abdul-Mageed and Diab 2011), opinion mining (Benamara et al. 2012) and scientific discourse evaluation (Waard and Maat 2012), among others.

To-date, there are no large-scale modality-annotated Arabic corpora compared to English (Baker et al. 2010, 2012; Rubinstein et al. 2013), Chinese (Cui and Chi 2013), Portuguese (Hendrickx et al. 2012) and Japanese (Matsuyoshi et al. 2010). The creation of modality-annotated corpora is non-trivial because there is no consensus definition of modality and its attributes in theoretical linguistics to be rendered into annotation tasks and guidelines. Furthermore, most current modality annotation schemes rely on sophisticated theoretically-grounded guidelines that require annotators from linguistics background; hence, annotation is usually restricted to small-scale in-lab settings.

In this paper, we present *3arif*, a large-scale Arabic corpus annotated for epistemic modality. *3arif* comprises 9822 unique tweets in Modern Standard Arabic (MSA) and Egyptian Arabic (EA), annotated for 9966 tokens that map to 214 unique types of epistemic modality. Each epistemic modality is annotated for sense, polarity, intensification, tense, holder(s) and scope(s). The reason that *3arif* features the tweets' genre with an emphasis on MSA and EA tweets is that it comes as part of a larger project to incorporate linguistic features, such as modality, with network-based features to automatically identify the key players of Twitter's political discourse in counties of political unrest such as Egypt. We harvested *3arif* from a variety of Twitter users including newspapers, TV stations, political campaigns, among others, as well as individuals. As a result *3arif* is diglossic for MSA, the formal Arabic variety, and EA, the native Arabic dialect of Egypt.

For the annotation of *3arif*, we design a simplified procedure that depicts the following ideas: first, it defines each annotation task as a series of open and closed questions that do not require sophisticated linguistics background and, meanwhile, provide annotators with self-explanatory annotation guidelines; second, it is interactive so that questions are displayed/hidden based on annotators' prior answers; and finally, it semi-automatically identifies and merges nested epistemic modality based on annotators' answers to a number of easy-to-administer questions.

[1] Pronounced as *ʕa:rif* in Arabic IPA and as *EArif* in Buckwalter's transliteration scheme. It means *I/he know(s)*.
[2] *3arif* is available at http://www.rania-alsabbagh.com/3arif.html

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1521–1532, Dublin, Ireland, August 23-29 2014.

We evaluate our annotation results using Krippendorff's reliability (Krippendorff 2011) and agreement. Results show high inter-annotator reliability and agreement rates and indicate that our annotation scheme and procedure are efficient. The contribution of this research, therefore, is twofold: first, we create a novel resource for Arabic NLP which is expected to enhance research on modality automatic identification and extraction; second, we present an efficient and easy-to-administer annotation procedure with interactive crowdsourcing potentials for the complex task of modality annotation.

The rest of this paper is organized as follows: Section 2 outlines our annotation scheme including annotation tasks, guidelines and the interactive structure; Section 3 gives examples for the representation of the final annotation outputs; Section 4 describes corpus harvesting and sampling; Section 5 discusses the results and presents a disagreement analysis; Section 6 compares and contrasts our work to related work; and Section 7 highlights the points not covered in this current version of *3arif*.

2 Annotation Scheme

Our annotation scheme consists of six tasks to label sense, polarity, intensification, tense, holders and scopes for each epistemic modality. Prior to the beginning of the interactive annotation procedure, we highlighted all candidate epistemic modalities in each tweet using a string-match algorithm and the lexicons from Al-Sabbagh et al. (2013, 2014). The algorithm finds all potential epistemic modality triggers (i.e. words and phrases that may convey epistemic modality) within each tweet in our corpus and marks them as annotation units. A total of 9966 candidate epistemic modality triggers are highlighted in 9822 tweets.

2.1 Task 1: Sense

Sense annotation is to decide for each highlighted candidate trigger in context whether it actually conveys epistemic modality. The same lexical verb اشعر *A$Er* is used as an epistemic modality trigger anticipating a future possibility in example 1; but as a non-modal lexical verb in example 2.

1. [3] **اشعر** ان[نا سنكسر رقم ال30 مليون متظاهر]
A$Er *An[nA snksr rqm Al30 mlywn mtZAhr]*
I **feel** that [we will get 30+ million protesters].

2. #هيكل: **اشعر** بالفخر والقلق أيضا في ذكرى حرب أكتوبر
#hykl: ***A$Er*** *bAlfxr wAlqlq >yDA fy *krY Hrb >ktwbr*
#Heikl: I **feel** proud but also worried when I remember October's war.

We define sense annotation as a synonymy judgment task, following Al-Sabbagh et al. (2013). Epistemic modality is represented by an exemplar set manually selected so that: (1) each exemplar is an unambiguous epistemic trigger, (2) exemplars are in both MSA and EA, (3) exemplars comprise both simple words and multiword expressions, (4) exemplars are both affirmative and negative, and (5) exemplars are of different lexical intensities. Furthermore, we create multiple versions of the same set so that we cover the inflections for gender, number, person, tense, mood, and aspect in Arabic. We then use the set that morphologically matches the candidate trigger to be annotated. Presented with a pre-highlighted candidate trigger in context and the exemplar set, annotators are to decide whether the given candidate trigger is synonymous to the exemplar set, and is hence an epistemic modality trigger, or not.

If an annotator decides that a given candidate trigger does not convey epistemic modality, no further questions about polarity, intensification, tense, holders or scopes are displayed. To guarantee that annotators do not select the non-synonymous option as an easy escape, they are not allowed to move forward without submitting at least one synonym of their own to the candidate trigger.

Designing the interactive procedure as such results in disagreement propagation. If one annotator decides that a given candidate trigger is not epistemic, but another annotator decides that it is, the former will not have to answer any further questions about polarity, intensification, tense, holders or scopes; whereas the latter will have to provide answers for each of those annotation tasks.

[3] Throughout the examples, epistemic modality triggers are represented in boldface and scopes are in-between square brackets.

2.2 Task 2: Polarity

Task 2 uses as input the candidates labeled as valid epistemic modality triggers in Task 1 and labels each as either affirmative or negative. An affirmative trigger indicates that the speaker holds the given state of affairs (i.e. propositions) as TRUE; whereas a negative trigger indicates that the given propositions are held as FALSE by the speaker.

To decide on whether the polarity is affirmative or negative, annotators are instructed to look for the absence/presence of such negation markers as:

- **Negation particles** such as مش *m$* (not), لا *lA* (not) and غير *gyr* (not), among others.
- **Negation affixes** like the circumfix *m...$* in مظنش *mZn$* (I do not think).
- **Negative polarity items** like عمري *Emry* (never) and لم يعد *lm yEd* (no longer).
- **Negative auxiliaries** where negation is placed on the past tense auxiliary as in مكنتش واثق *mknt$ wAvq* (I was not sure).
- **Inherently-negative triggers** that encode negation in their lexical meanings such as مستحيل *mstHyl* (impossible).

Annotators are instructed that using multiple negation markers results in an affirmative sense. Thus, ليس من المستحيل *lys mn AlmstHyl* (it is not impossible) means that the proposition is actually possible according to the speaker. Put differently, it means that the speaker holds the proposition as TRUE. Annotators are required to give the reason for negation if they decide that a given trigger is negative.

2.3 Task 3: Intensification

Epistemic modality triggers can have different lexical intensities (i.e. intensities encoded in the lexical meaning of the word/phrase regardless of the context). For instance, even without a context, Arabic speakers know that متأكد *mt>kd* (I am/he is sure) expresses higher possibility than متهيألي *mthy>ly* (I imagine). When used in context, the trigger's lexical intensity can be maintained as is. Yet, it can also be amplified or mitigated by various linguistic means such as:

- **Modification:** adverbs like تماما *tmAmA* (absolutely) and بالفعل *bAlfEl* (indeed), among others, amplify lexical intensity; whereas mitigation can be caused by such adverbs as تقريبا *tqrybA* (almost) and غالبا *gAlbA* (most probably), among others.
- **Categorical negation** typically amplifies lexical intensity as in مش ممكن أبدا *m$ mmkn >bdA* (it is not possible at all).
- **Emphatic expressions** such as قد *qd* (indeed) and والله *wAllh* (I swear), among others, lead to lexical intensity amplification.
- **Coordination** of two or more triggers usually results in intensity amplification as in عارف ومتأكد *EArf wmt>kd* (I know and I am sure).

The annotators' task for intensification is to decide for each candidate labeled as a valid epistemic modality trigger in Task 1 whether its lexical intensity is amplified (AMP), mitigated (MTG), or maintained (AS IS). During interactive annotation, annotators are asked to provide the reason for their selection; that is, whether the lexical intensity is affected by an adverb, categorical negation, an emphatic expression, coordination, or any other reason.

2.4 Task 4: Tense

In this version of *3arif*, we work on the present and past tenses only. Thus, Task 4 is to decide for each valid epistemic trigger from Task 1 whether it is present (PRS) or past (PST). Tense can be marked either morphologically by inflections and affixes or contextually by auxiliary verbs such as كان *kAn* (was), among others. Annotators are also required to give their reasons for selecting either PRS or PST.

2.5 Task 5: Holder

Holder annotation is to identify the holder of the epistemic modality which is the ±RATIONAL entity that expresses its knowledge, beliefs or judgments about the world's states of affairs.

Holders can be −RATIONAL entities as in example 3. The entity that is making the assumption that the former Palestinian president - Yasser Arafat - may have died of natural causes is the report issued by the French government.

3. [تقرير فرنسي: [وفاة #عرفات **ربما** تعود لاسباب طبيعية]
*tqryr frnsy: [wfAp #ErfAt **rbmA** tEwd lAsbAb TbyEyp]*
A French report: [natural causes **might** be behind the death of #Arafat].

The holder is not necessarily the same as the trigger's grammatical subject. In example 4, the grammatical subject of يبدو *ybdw* (seems) is الاعلان الدستوري *AlAElAn Aldstwry* (the constitutional declaration). However, the entity that is making the judgment about this declaration is the French government, which is then the real holder of *ybdw*.

4. [فرنسا: [الاعلان الدستوري الجديد لمرسي لا **يبدو** انه يسلك الاتجاه الصحيح]
*frnsA: [AlAElAn Aldstwry Aljdyd lmrsy lA **ybdw** Anh yslk AlAtjAh AlSHyH]*
France: [Morsi's new constitutional declaration does not **seem** to be a correct move].

Twitter users do not only post their own knowledge, beliefs and judgments about the world's states of affairs, but also they (1) directly and indirectly quote others and (2) make assumptions about others' knowledge, beliefs and judgments. This means that we can have nested holders, according to Wiebe et al. (2005) and Saurí and Pustejovsky (2009), where we know about others' knowledge, beliefs and judgments only though the writer or the Twitter user in our case.

In example 5, the Twitter user quotes Elbaradei stating that he may run for presidency if the people want him to. That is, the holder of the epistemic modality is actually Elbaradei not the Twitter user.

5. البرادعي: **قد** [أترشح في انتخابات الرئاسة] إذا طلب الشعب
*AlbrAdEy: **qd** [>tr$H fy AntxAbAt Alr}Asp] <*A Tlb Al$Eb*
Elbaradei: I **may** [run for presidency] if the people want me to.

The holder of the epistemic modality in example 6 is not the Twitter user, either. However, the Twitter user is not quoting anyone here, but is rather making an assumption about what the Egyptian National Party holds as TRUE.

6. [الحزب الوطني **مقتنع** ان[ه ممكن يرجع] #Jan25
*AlHzb AlwTny **mqtnE** An[h mmkn yrjE] #Jan25*
The National Party is **convinced** that [it may get back to authority]. #Jan25

We can have two or more nested holders. In example 5, we have two: the first is ElBaradei and the second is the Twitter user who is quoting ElBaradei. Similarly, in example 6, we have two nested holders: the first is the Egyptian National Party and the second is the Twitter user who makes the assumptions about the party's beliefs.

In example 7, however, we have three nested holders. The first is الاخوان *AlAxwAn* (the Muslim Brotherhood) that holds as TRUE the proposition that the Military Council is conspiring against them. That belief of the Muslim Brotherhood is communicated to us through the politician ابو الفتوح *Abw AlftwH* (Abulfotoh) who is then the second holder. Yet, Abulfotoh has not posted his assumption about the Muslim Brotherhood's belief on his personal account. Instead, he has been quoted by another Twitter user, who is the third holder.

7. [ابو الفتوح: الاخوان **تصوروا** ان [هناك مؤامرة من العسكري]
*Abw AlftwH: AlAxwAn **tSwrwA** An [hnAk m&Amrp mn AlEskry]*
Abulfotoh: The Muslim Brotherhood members **thought** that [there was a conspiracy by the Military Council].

During the interactive procedure, annotators are first asked whether the holder is the same as the Twitter user. If not, more questions are displayed to determine: (1) who the real holder is; (2) whether the tweet is a(n) (in)direct quote (e.g. there are direct quotation markers or such words as قال *qAl* (he said) and صرح *SrH* (he declared), among others), or the tweet conveys the Twitter user's assumptions about others.

When the holder is not the same as the Twitter user, annotators are asked to mark the boundaries of the linguistic unit that corresponds to the holder in the tweet's text, following the maximal length principle from Szarvas et al. (2008), so that they mark the largest possible, meaningful linguistic unit. Hence, in example 8 the holder is *the Islamist opponents in #KSA* not only *the Islamist opponents*.

8. [الإسلاميون المعارضون في #السعودية **موقنون** أن[ها تسعى لقتل الثورة في #مصر]
*Al<slAmywn AlmEArDwn fy #AlsEwdyp **mwqnwn** >n[hA tsEY lqtl Alvwrp fy #mSr]*
Islamist opponents in #KSA **know for sure** that [it tries to put an end to #Egypt's revolution].

2.6 Task 6: Scope

Scopes are the states of affairs modified by the epistemic modality triggers. Modality scopes in Arabic are most likely realized as clauses, deverbal nouns or to-infinitives, according to Al-Sabbagh et al. (2013). We use the same maximal length guideline from Task 5 so that the scope segment marked by the annotators is the largest possible segment typically delimited by: (1) punctuation markers and (2) subordinate conjunctions such as لان *lAn* (because) and لو *lw* (if), among others.

In the case of nested triggers as in example 9, where a trigger and its scope are both embedded in another trigger's scope, the interactive procedure prompts the annotators to label each trigger and its scope separately at first. Afterwards, we automatically merge them as we further explain in Section 3.

9. [[الحزب الوطني **مقتنع** أن[ه **ممكن** [يرجع]] #Jan25
*AlHzb AlwTny **mqtnE** >n[h **mmkn** [yrjE]] #Jan25*
The National Party is **convinced** that [it **may** [get back to power]] #Jan25

Annotators are instructed that a single trigger may have one or more scopes. In example 10, the trigger بيتهيألهم *bythy>lhm* (they imagine) scopes over two complement clauses, which annotators are required to identify. Furthermore, annotators are given the guideline that two or more triggers - typically conjoined by a coordinating conjunction - can share the same scope as in example 11. In the cases like example 11, each trigger and its attributes are first annotated separately and then once our system finds out that they share the same polarity, intensification, tense, holder, and scope, they are merged together as we show in Section 3.

10. [أولادنا **بيتهيألهم** ان [دم اخواتهم راح هدر] وان[هم عندهم ثأر مع السلطة بكل أشكالها]
*>wlAdnA **bythy>lhm** An [dm AxwAthm rAH hdr] wAn[hm Endhm v>r mE AlslTp bkl >$kAlhA]*
Our children **imagine** that [their friends were killed for no reason] and that [they now have to take revenge from the authorities].

11. البرادعي **عارف ومتاكد** ان [نسبة 12 % بس هتنتخبه] وعلشان كدة مش هيرشح نفسه
*AlbrAdEy **EArf wmtAkd** An [nsbp 12% bs htntxbh] wEl$An kdp m$ hyr$H nfsh*
Elbaradei **knows and is sure** that [only 12% will vote for him]. So, he will not run for presidency.

Annotators are instructed that scopes are not necessarily adjacent to their triggers. In example 12, the scope starts three words to the right of its trigger باقتنع *bAqtnE* (get convinced) given that the adverbial phrase اكتر واكتر *Aktr wAktr* (more and more) falls in between it and its scope.

12. [كل يوم بيعدي **باقتنع** اكتر واكتر ان[نا كنا محتاجيين دكتاتور وطني عادل]
*kl ywm byEdy **bAqtnE** Aktr wAktr An[nA knA mHtAjyn dktAtwr wTny EAdl]*
Every day, I **get more and more convinced** that [we needed a patriotic and fair dictator].

Annotators are also instructed that scopes can (1) precede, (2) follow or (3) surround their triggers. Many of the aforementioned examples have the scopes following their triggers. Yet, in example 13 the scope surrounds its trigger and in example 14 it precedes its trigger.

13. [وعود مرسي ليست **فيما يبدو** دين عليه]
*[wEwd mrsy lyst **fymA ybdw** dyn Elyh]*
[Morsi's promises are not **seemingly** doable].

14. **فيما يبدو** [حملة تشويه ثورة يناير وإعادة عقارب الساعة تماما إلى الوراء بدأت]
*[Hmlp t$wyh vwrp ynAyr w<EAdp EqArb AlsAEp tmAmA <lY AlwrA' bd>t] **fymA ybdw***

[A campaign to distort the image of January's revolution and to restore everything back to its original state has started], **seemingly**.

3 Final Output Representation

All elicited answers during annotation are automatically organized into the representations illustrated in the examples below. The representation of example 15 reads as follows: the USER (i.e. the Twitter user) used to moderately hold as TRUE the proposition that the revolutionist candidates were unable to compete for presidency. We know that this is a past belief that the USER used to have because annotators have labeled the trigger تصورت *tSwrt* (I thought) as past (PST). There are no nested holders given that the USER is the same as the holder. The intensity value of MODerate comes from the fact that تصورت *tSwrt* (I thought) is of a moderate lexical intensity being weaker than such epistemic triggers as متاكد *mtAkd* (I am sure) and عارف *EArf* (I know) but stronger than such epistemic triggers as اظن *AZn* (I guess) and متهيالي *mthyAly* (I imagine). Meanwhile, the lexical intensity of *tSwrt* is neither amplified nor mitigated; hence annotators have given it an AS IS intensification label in Task 3. Consequently, in the final annotation output the original lexical intensity value has been used to represent how far the holder used to consider his/her belief as TRUE.

15. [في البداية **تصورت** ان [مرشحي الثورة اضعف من المنافسة للرئاسة]
*fy AlbdAyp **tSwrt** An [mr$Hy Alvwrp ADEf mn AlmnAfsp llr}Asp]*
At first, I **thought** that [the revolutionist candidates are too weak to compete for presidency].

rep. USER, MOD PST TRUE, (*mr$Hy Alvwrp ADEf mn AlmnAfsp llr}Asp*)

Example 16 shows how two epistemic modality triggers in the same tweet are given two separate representations because they share the same holder but neither the same intensity nor the same scopes. The first representation illustrates the epistemic trigger ارى *ArY* (I think) and reads as follows: the USER currently holds as TRUE the proposition that the media is misleading the people; s/he is MODerately confident about that. The second representation is for the epistemic trigger واضح *wADH* (obviously). It indicates that the same USER strongly holds as TRUE the proposition that the media is trying to stop the change that the people are longing for. Both triggers are labeled as present (PRS) tense. Furthermore, both triggers are labeled as maintaining their lexical intensity AS IS. The trigger ارى *ArY* (I think) is then labeled in the final representation as being of MODerate intensity because it is weaker than متاكد *mtAkd* (I am sure), for instance, but stronger than متهيالي *mthyAly* (I imagine); whereas the trigger واضح *wADH* (obviously) is labeled as indicating a strong (STRG) belief being synonymous to متاكد *mtAkd* (I am sure) and اعرف *AErf* (I know) among other triggers that express speakers' high confidence about their knowledge, beliefs and judgments.

16. [ارى ان [الاعلام يقدم شباب يخدرون الشعب] **واضح** ان[هم يقاومون التغيير الذى نطمح له]
***ArY** An [AlAElAm yqdm $bAb yxdrwn Al$Eb] **wADH** An[hm yqAwmwn Altgyyr Al*y nTmH lh]*
I **think** [the media presents young speakers who mislead the people]. **Obviously**, [they are resisting the change we are longing for].

rep1. USER, MOD PRS TRUE, (*AlAElAm yqdm $bAb yxdrwn Al$Eb*)
rep2. USER, STRG PRS TRUE, (*hm yqAwmwn Altgyyr Al*y nTmH lh*)

Example 17 illustrates how two coordinating epistemic triggers sharing the same polarity, tense, intensification, holder and scope are represented. They are simply merged in one representation. The same example shows how assumptions made by Twitter users about others' knowledge, beliefs and judgments are represented. The representation reads as follows: the USER MODerately holds as TRUE the proposition that Elbaradei strongly (STRG) holds as TRUE that only 12% of the Egyptians will vote for him for presidency. The values of TRUE, MODerate and present (PRS) assigned to the USER's assumption about Elbaradei are default values used to mark Twitter users' assumptions about others' knowledge, beliefs and judgments.

17. البرادعى **عارف ومتاكد** ان [نسبة 12 % بس هتنتخبه] وعلشان كدة مش هيرشح نفسه
*AlbrAdEy **EArf wmtAkd** An [nsbp 12% bs htntxbh] wEl$An kdp m$ hyr$H nfsh*
Elbaradei **knows and is sure** that [only 12% will vote for him]. So, he will not run for presidency.

rep. USER, MOD PRS TRUE, (*AlbrAdEy*, STRG PRS TRUE, (*nsbp 12% bs htntxbh*))

Example 18 represents an epistemic trigger with multiple scopes. The example also represents Twitter users making assumptions about others' knowledge, beliefs and judgments. As we mentioned in example 17, the values of TRUE, MODerate and present (PRS) assigned to the USER's assumption are assigned by default. The trigger بيتهيألهم *bythy>lhm* (they imagine) is labeled as a present (PRS) tense affirmative trigger. Its original lexical intensity - which is weak (WK) - is labeled as being maintained AS IS. The trigger بيتهيألهم *bythy>lhm* (they imagine) is of a weak lexical intensity because it is weaker than متاكد *mtAkd* (I am sure) and even اظن *AZn* (I think).

18. [أولادنا بيتهيألهم ان [دم اخواتهم راح هدر] وان[هم عندهم ثأر مع السلطة بكل أشكالها]
*>wlAdnA **bythy>lhm** An [dm AxwAthm rAH hdr] wAn[hm Endhm v>r mE AlslTp bkl >$kAlhA]*
Our children **imagine** that [their friends were killed for no reason] and that [they now have to take revenge from the authorities].

rep. USER, MOD PRS TRUE, (*>wlAdnA,* WK PRS TRUE ,(*dm AxwAthm rAH hdr; hm Endhm v>r mE AlslTp bkl >$kAlhA*))

Example 19 illustrates embedded triggers. Its representation reads as: the USER MODerately holds as TRUE that the Egyptian National Party strongly (STRG) holds as TRUE that it (i.e. the Egyptian National Party) may get back to ruling. It is important to notice that both the matrix trigger مقتنع *mqtnE* (is convinced) and the embedded trigger (i.e. ممكن *mmkn* (may)) share the same holder which is the Egyptian National Party.

19. [[الحزب الوطني **مقتنع** ان[ه **ممكن** [يرجع]] #Jan25
*AlHzb AlwTny **mqtnE** An[h mmkn [yrjE]] #Jan25*
The National Party is **convinced** that [it **may** [get back to power]].

rep. USER, MOD PRS TRUE, (*AlHzb AlwTny*, STRG PRS TRUE,(MOD PRS TRUE, (*yrjE*)))

Example 20 shows how reported knowledge, beliefs and judgments are represented. The USER in this example has no other role but to report Darrag's strong belief that the army will interfere to stop the chaos.

20. دراج: [#الجيش **حتما** سيتدخل في حالة الفوضى] #مصر #مرسي #الاخوان
*drAj: [#Aljy$ **HtmA** sytdxl fy HAlp AlfwDY] #mSr #mrsy #AlAxwAn*
Darrag: [the #army will **definitely** interfere in the case of chaos] #Egypt #Morsi #Ikhwan

rep. USER, report, (*drAj*, STRG PRS TRUE (*#Aljy$ sytdxl fy HAlp AlfwDY*))

4 Corpus Harvesting

In order to restrict our corpus to political discourse and ensure that we compile a representative corpus of epistemic modality, we harvested our corpus so that each tweet (1) has at least one trendy political English or Arabic hashtag such as #Egypt and #مرسي *mrsy* (Morsi)[4], and (2) has at least one epistemic modality trigger from the Arabic Modality Lexicons of Al-Sabbagh et al. (2013, 2014). Table 1 gives statistics for the sampled corpus that comprises 9822 unique tweets, with 9966 candidate epistemic modality triggers that map to 214 unique types.

	Tokens	**Types**
Epistemic candidates	9966	214
All words	175964	47696

Table 1: Statistics for the sampled corpus

5 Annotation Results

5.1 Evaluation Methodology and Metrics

Our annotation tasks are of two types: (1) Tasks 1-4 are label-based where there is a pre-defined set of labels from which annotators choose; and (2) Tasks 5-6 are segmentation-based where the output of the annotation is a text segment. For the segmentation-based tasks, we use an all-or-nothing method to

[4] A total of 304 unique English and Arabic hashtags are found in the sampled corpus.

measure reliability and agreement: for segments to be considered as agreement, they must share both the beginning and end boundaries. We use Krippendorff's alpha α (Krippendorff 2011) as our inter-annotator reliability measure, following the most recent work on modality annotation for other languages including English (Rubinstein et al. 2013) and Chinese (Cui and Chi 2013). For more details on Krippendorff's alpha and a comparison of inter-annotator agreement measures, we refer the reader to Artstein and Poesio (2008).

5.2 Results

We use the surveygizmo services to implement our interactive annotation procedure given that their survey structure is one that allows for using conditional branching and skip logic[5]. We distributed the survey on Twitter and we had three annotators participating. According to the short qualifying quiz given at the beginning of the survey, all three participants are native Egyptian Arabic (EA) speakers who have at least two-year experience with using Twitter. They are also university graduates who, therefore, master Modern Standard Arabic. None of the participants has a linguistics background.

Table 2 shows alpha and agreement rates for each annotation task. We measure the rates in four different scenarios so that we can (1) estimate the effect of the inclusion of the NON-EPISTEMIC category agreement, (2) estimate the effect of disagreement propagation from Task 1, and (3) evaluate the guidelines and procedures for each annotation task separately. The four scenarios are:

- **w/NONE w/DP:** candidates agreed upon as non-epistemic and disagreement propagating from Task 1 are both included.
- **w/NONE w/o DP:** candidates agreed upon as non-epistemic are included, but disagreement propagating from Task 1 is excluded.
- **w/o NONE w/DP:** candidates agreed upon as non-epistemic are excluded, but disagreement propagating from Task 1 is included.
- **w/o NONE w/o DP:** candidates agreed upon as non-epistemic and disagreement propagating from Task 1 are both excluded. This scenario focuses on each annotation task separately without any distractions.

		Alpha				**Agreement**			
		w/NONE		**w/o NONE**		**w/NONE**		**w/o NONE**	
Annotation Task		**w/ DP**	**w/o DP**	**w/ DP**	**w/o DP**	**w/ DP**	**w/o DP**	**w/ DP**	**w/o DP**
1	Sense	--	0.899	--	--	--	0.949	--	--
2	Polarity	0.904	0.974	0.798	0.949	0.939	0.983	0.895	0.976
3	Intensification	0.880	0.942	0.658	0.768	0.926	0.966	0.844	0.939
4	Tense	0.911	0.995	0.772	0.983	0.947	0.997	0.909	0.994
5	Holder	0.878	0.930	0.672	0.727	0.933	0.956	0.884	0.969
6	Scope	0.825	0.916	0.620	0.618	0.899	0.955	0.819	0.911

Table 2: Inter-annotator alpha reliability and agreement rates

In the case of Task 1 (i.e. sense annotation), only the second scenario is applicable: we cannot exclude the candidates agreed upon as non-epistemic because the target is to know how reliable the annotation is with regards to distinguishing between epistemic and non-epistemic candidates. It is the first annotation task, thus there is no prior disagreement propagation. From Table 2, we derive the following observations:

- Disagreement in Task 1 propagates ~ 0.05 to 0.1 disagreement for the other annotation tasks.
- Adding the agreed upon non-epistemic candidates yields up to ~ 0.2 gain for both alpha reliability and agreement rates.
- For an end-to-end automatic system that first identifies triggers and then their attributes, the benchmark rates are those from the w/NONE w/DP scenario.

[5] http://www.surveygizmo.com/

5.3 Discussion and Disagreement Analysis

Among the factors that lead to high inter-annotator alpha reliability and agreement rates are that: (1) the vast majority of negation is explicitly marked by negation particles that are easy to detect by human annotators; (2) the vast majority of triggers are used without any amplification or mitigation markers; and (3) punctuation markers are surprisingly informative for marking scope boundaries and direct quotations and, hence, holders.

Sense-related disagreement is attributed to: (1) nominal triggers with main grammatical functions, (2) stative triggers, (3) opinionated-evidential triggers and (4) highly-polysemous triggers.

The majority of epistemic triggers are adjunct constituents that add an extra-layer of meaning and can be removed without disturbing the syntactic structure of their propositions. Yet, in example 21, احتمال *AHtmAl* (a possibility) is the grammatical subject of the proposition it modifies. Most of the exemplars from Section 2.1 are adjuncts and, thus, none can be both a lexical and a grammatical substitute for احتمال *AHtmAl* (a possibility) in such a context.

21. **احتمال** ان [رئيس منتخب يحل المجلس اثناء صياغة دستور جديد] **احتمال** وهمي
AHtmAl *An [r}ys mntxb yHl Almjls AvnA' SyAgp dstwr jdyd]* ***AHtmAl*** *whmy*
The **possibility** that [an elected president dissolves the parliament during the constitution's write-up] is an unrealistic **possibility**.

Stative triggers such as يعرف *yErf* (he knows) and يدرك *ydrk* (he realizes) invoke disagreement as to whether they indicate the acquisition of new information; that is, they literally mean *perceive*, or they mark confirmed beliefs as in *be sure that*. For example 22, the annotators have two interpretations: (1) a non-modal interpretation that *whoever says so does not perceive that the Supreme Guide cannot make resolutions without the Brotherhood*, and (2) a modal interpretation that *whoever says so does not believe that the Supreme Guide cannot make resolutions without the Brotherhood.*

22. [الذي يقول هذا الكلام لا **يعرف** ان [المرشد لا يستطيع اخذ قرار دون الرجوع الى الجماعة]
*Al*y yqwl h*A AlklAm lA* ***yErf*** *An [Almr$d lA ystTyE Ax* qrAr dwn AlrjwE AlY AljmAEp].*
Whoever says so does not **perceive/believe** that [the Supreme Guide cannot make resolutions without the Brotherhood].

Opinionated-evidential triggers like يزعم *yzEm* (he claims) do not only mark reported speech, but also they communicate the reporter's own opinion about the truth value of the reported proposition. They entail that from the reporter's perspective the proposition is FALSE. Hence, annotators disagree as to whether *yzEm* and similar triggers should be labeled as epistemic or not. We have eventually excluded such triggers as epistemic and have included them as evidential triggers for another corpus that is left for a future publication.

Highly-polysemous triggers like يمكن *ymkn* (can/possible) lead to disagreement because in many cases even the context is ambiguous. In example 23, both interpretations of *it is not possible that* (epistemic) and *it is not doable that* (abilitive) seem to be acceptable.

23. [لا **يمكن** [فهم كتاب مرسي "ثائر من الشرق" الا بتامل الكتابين المجاورين: "سرقات صغيرة" و"جنون الحكم"]
lA ***ymkn*** *[fhm ktAb mHmd mrsy "vA}r mn Al$rq" AlA btAml AlktAbyn AlmjAwryn: "srqAt Sgyrp" w "jnwn AlHkm"*]
It is **not possible/doable** [to understand Morsi's book - *A Revolutionist from the East* - without reading the other two books of *Small Robberies* and *Ruling Mania*].

Intensity-related disagreement is attributed to (1) intensity on the holder that propagates to the trigger and (2) negation with moderate-intensity triggers. In example 24, the USER uses categorical negation on the holder لا يوجد اي انسان عاقل *lA ywjd Ay AnsAn EAql* (there is no one sane person). For some annotators, the power of categorical negation spreads to the trigger, moving its intensity up the scale. As for negation with moderate-intensity triggers, some annotators think that لا يمكن *lA ymkn* (not possible) is synonymous to *impossible*. Hence, they consider the negation as an amplification marker.

24. لا يوجد أي انسان عاقل **يعتقد** بأن [الارهاب يعالج بالسياسة]
lA ywjd >y AnsAn EAql ***yEtqd*** *b>n [AlArhAb yEAlj bAlsyAsp]*
There is no one sane person who **thinks** that [terrorism can be defeated through politics].

Polarity-related disagreement is mainly caused by negation due to (1) negated holders and (2) contextual negation. Negated holders as in example 24 perplex the annotators as to whether the negation scopes over the holder only or both the holder and the trigger. Thus, for some annotators, يعتقد *yEtqd* (he thinks) is affirmative; and for others it is negative. By contextual negation we mean using words such as المشكلة *Alm$klp* (the problem) to describe triggers as in example 25. The USER says that *the problem is to think that it is a small-scale conflict.* To describe this as a *problem* means that the USER thinks of the proposition as FALSE; that is, according to the USER it is actually a large-scale conflict.

25. المشكلة إننا **نتصور** إن [الصراع محصور فى الدائرة الضيقة اللى بنتحرك فيها]
*Alm$klp <nnA **ntSwr** <n [AlSrAE mHSwr fY AldA}rp AlDyqp AllY bntHrk fyhA]*
The problem is to **think** that [the conflict is only happening at this small-scale we are working on].

Holder-related disagreement is attributed mainly to generic nouns and impersonal pronouns such as الشعب *Al$Eb* (the people) and الواحد *AlwAHd* (one). Some annotators interpret them as implicitly referring to the USER. Therefore, they select the USER as the only holder with zero nesting in example 26. Other annotators interpret them as referring to people in general but not necessarily with the USER included; and thus, they select two-level nested holders.

26. الشعب **يعرف** ان [الممارسة الديمقراطية هى التي ستأتي باعضاء مجلس الشعب والرئيس القادم]
*Al$Eb **yErf** An [AlmmArsp AldymwqrATyp hy Alty st>ty bAEDA' mjls Al$Eb wAlr}ys AlqAdm]*
People **know** that [democracy will result in real parliamentary and presidential elections].

Scope-related disagreement is attributed to (1) ambiguous subordinate conjunctions, (2) triggers modifiers, (3) absent punctuation markers, and (4) embedding within the scope boundaries. For instance, in example 27, the adverbial clause starting with بعد *bEd* (after) confuses the annotators as to whether it is part of the scope or it describes the verb epistemic trigger اتوقع *AtwqE* (I expect).

27. **اتوقع** جدا ان [اعتصام التحرير يتفض بنفس طريقة فض الاعتصام الاخير بعد ظهور اشكال غريبة فلجان الامن]
***AtwqE** jdA An [AEtSAm AltHryr ytfD bnfs Tryqp fD AlAEtSAm AlAxyr bEd Zhwr A$kAl grybp fljAn AlAmn]*
I very much **expect** that [the sit-in in Tahrir will be broken up in the same way as the last sit-in after seeing some strange faces at the security checkpoints].

Tense yields almost perfect inter-annotator alpha reliability and agreement rates. The one main disagreement factor, however, is such contexts as ابتديت اصدق *Abtdyt ASdq* (I started to believe). While the majority of annotators agree that such contexts mark present tense knowledge, beliefs and judgments, some annotators consider them as past tense.

5.4 Majority Statistics for *3arif*

Based on majority annotations, Table 3 gives statistics for *3arif* in terms of sense, polarity, intensification and tense. Furthermore, approximately 62% of the triggers have zero-nested holders (i.e. the Twitter user is the same as the holder). As for scope syntactic structures, they are distributed as 86% clauses, 9% deverbal nouns and the rest are to-infinitives.

	Sense		**Polarity**		**Intensification**			**Tense**	
	Epistemic	Non-epistemic	True	False	Amplified	Mitigated	As is	Present	Past
Tokens	5591	4375	3425	2166	1083	330	4178	4399	1192
Types	209	175	176	134	133	50	150	175	104

Table 3: Majority statistics for *3arif*

6 Related Work

Epistemic modality has been the focus of many annotation projects for multiple languages. Diab et al. (2009) annotate three belief categories for English: (1) committed belief is when writers indicate that they hold propositions as TRUE, (2) non-committed belief is when writers hold propositions as FALSE, and (3) not applicable is when propositions are not denoting beliefs at all. Interest is given to writers' beliefs only. Thus, a default value for the modality holder is the writer, and nested holders are not an-

notated. Their corpus contains 10k words of running text from different domains and genres, including newswire, blog data, email and letter correspondence and transcribed dialogue data. Inter-annotator agreement rate is 0.95 including the NONE category where no belief markers exist.

Baker et al. (2010, 2012) simultaneously annotate modality and modality-based negation to build modality taggers to enhance Urdu-English machine translation systems. Their annotation scheme distinguishes eight modality types: requirements, permissions, success, effort, intention, ability, desires and beliefs. Originally, their annotation scheme labels three attributes for each modality type: triggers, holders and targets (i.e. scopes). Yet, holders have not been eventually labeled. A unique feature of their annotation scheme is using a simplified operational procedure to label modality semantic meanings. The procedure relies on a list of thirteen choices of the form of H (modal) [P true/false] where H is a holder and P is a proposition or an event. The annotators' task is then to select the best form to represent the modality meaning of a given trigger. Reported kappa κ inter-annotator agreement rates are 0.82 for triggers and 0.76 for targets.

Rubinstein et al. (2013) propose a linguistically-motivated scheme for modality annotation in the MPQA English corpus. They attain macro alpha inter-annotator reliability rates of 0.89 and 0.65 for sense and scope, respectively. Cui and Chi (2013) apply the same scheme from Rubinstein et al. (2013) to the Chinese Penn Treebank and get alpha inter-annotator reliability rates of 0.81and 0.39 for sense and scope annotation, respectively.

Al-Sabbagh et al. (2013) annotate epistemic modality in MSA and EA tweets. We attain kappa inter-annotator agreement rates of 0.90 and 0.93 for sense and scope annotation, respectively, for only 548 epistemic tokens.

Our annotation results, therefore, are comparable to the results in the literature. Furthermore, our annotation scheme is orthogonal to most of the aforementioned schemes. However, the key differences between our work and related work are:

- We annotate nested modality, unlike Diab et al. (2009) and Baker et al. (2010, 2012).
- We use a wider range of negation and intensification markers compared to prior work, especially Al-Sabbagh et al. (2013)
- We use interactive crowdsourcing with simplified guidelines, unlike in-lab annotations including Rubinstein et al. (2013) and Cui and Chi (2013), among others.

7 Uncovered Points in *3arif*

The current version of *3arif* does not cover modality entailment that example 28 illustrates. The USER criticizes whoever holds as TRUE the proposition that Egypt can blackmail UAE using the Iranian threat. This criticism entails that the USER holds the same proposition as FALSE.

28. [يخطئ من **يظن** ان [#مصر يمكن ان تساوم الامارات بورقة #ايران]
*yxTY' mn **yZn** An [#mSr ymkn An tsAwm #Al<mArAt bwrqp #<yrAn]*
Whoever **thinks** that [Egypt can blackmail #UAE using #Iran] is wrong.

We do not also cover the future tense, the interrogative, the imperative or the hypothetical moods. This is because they have different interpretations when it comes to intensification and polarity that we do not cover in this version of *3arif* but we will in future work.

8 Conclusion

We presented *3arif*, a large-scale corpus annotated for epistemic modality in MSA and EA tweets. We used a simplified approach that defines each annotation task as a series of questions, implemented interactively. Our scheme covers a wide range of the most common annotation units mentioned in the literature, including modality sense, polarity, intensification, tense, holders and scopes. We deal with nested holders that are crucial in a highly interactive genre such as tweets where users frequently quote others and make assumptions about them. We also automatically merge triggers with shared holders and scopes based on elicited annotators' answers. The annotation procedure yields reliable results and creates a novel resource for Arabic NLP. For future versions of the corpus, we plan to cover the points

from Section 7. *3arif* will also be used to train and test an automatic machine learning system to identify epistemic modality and its attributes in MSA and EA tweets.

References

Muhammad Abdul-Mageed and Mona Diab. 2011. Subjectivity and Sentiment Annotation of Modern Standard Arabic Newswire. In *Proceedings of the 5th Linguistic Annotation Workshop (LAW V)*, pages 110-118, Portland, Oregon, June 23-24, 2011.

Rania Al-Sabbagh, Jana Diesner and Roxana Girju. 2013. Using the Semantic-Syntactic Interface for Reliable Arabic Modality Annotation. In *Proceedings of IJCNLP'13*, pages 410-418, Nagoya, Japan, October 14-18, 2013.

Rania Al-Sabbagh, Roxana Girju and Jana Diesner. 2014. Unsupervised Construction of a Lexicon and a Pattern Repository of Arabic Modal Multiword Expressions. In *Proceedings of the 10th Workshop of Multiword Expressions at EACL 2014*, pages 114-123, Gothenburg, Sweden, April 26-27, 2014.

Ron Artstein and Massimo Poesio. 2008. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, volume 34, issue 4, pages 555-596.

Kathrin Baker, Michael Bloodgood, Mona Diab, Bonnie Dorr, Nathaniel W. Filardo, Lori Levin and Christine Piatko. 2010. A Modality Lexicon and its Use in Automatic Tagging. In *Proceedings of LREC'10*, pages 1402-1407, Valetta, Malta, May 19-21, 2010.

Kathryn Baker, Michael Bloodgood, Bonnie J. Dorr, Chris Callison-Burch, Nathaniel W. Filardo, Christine Piatko, Lori Levin and Scott Miller. 2012. Modality and Negation in SIMT. *Computational Linguistics*. volume 38, issue 2, pages 411-438.

Farah Benamara, Baptiste Chardon, Yannick Mathieu, Vladimir Popescu and Nicholas Asher. 2012. How do Negation and Modality Impact on Opinions. In *Proceedings of the ACL-2012 Workshop on ExProM-2012,* pages 10-18, Jeju, Republic of Korea, July 13, 2012.

Yanyan Cui and Ting Chi. 2013. Annotating Modal Expressions in the Chinese Treebank. In *Proceedings of the IWC 2013 Workshop on Annotation of Modal Meaning in Natural Language (WAMM)*, pages 24-32, Potsdam, Germany, March 19, 2013.

Mona Diab, Lori Levin, Teruko Mitamura, Owen Rambow, Vinodkumar Prabhakaran, and Weiwei Guo. 2009. Committed Belief Annotation and Tagging. In *Proceedings of the 3rd Linguistic Annotation Workshop, ACL-IJCNLP'09*, pages 68-73, Suntec, Singapore, August 6-7, 2009.

Iris Hendrickx, Amàlia Mendes and Silvia Mencarelti. 2012. Modality in Text: A Proposal for Corpus Annotation. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12),* pages 1805-1812, Istanbul, Turkey, May 21-27, 2012.

Klaus Krippendorff. 2011. Computing Krippendorff's Alpha Reliability. Annenberg School of Communication, Departmental Papers: University of Pennsylvania.

Suguru Matsuyoshi, Megumi Eguchi, Chitose Sao, Koji Murakami, Kentaro Inui and Yuji Matsumoto. 2010. Annotating Event Mentions in Text with Modality Focus and Source Information. In *Proceedings of LREC'10*, pages 1456-1463, Valletta, Malta, May 19-21, 2010.

Frank R. Palmer. 2001. *Mood and Modality*. 2nd Edition. Cambridge University Press, Cambridge, UK.

Aynat Rubinstein, Hillary Harner, Elizabeth Krawczyk, Daniel Simoson, Graham Katz and Paul Portner. 2013. Toward Fine-Grained Annotation of Modality in Text. In *Proceedings of the IWC 2013 Workshop on Annotation of Modal Meaning in Natural Language (WAMM)*, pages 38-46, Potsdam, Germany, March 19, 2013.

Roser Saurí and James Pustejovsky. 2009. FactBank: A Corpus Annotated with Event Factuality. *Language Resources and Evaluation*, volume 43, pages 227-268.

György Szarvas, Veronika Vincze, Richárd Farkas and János Csirik. 2008. The BioScope Corpus: Annotation for Negation, Uncertainty and their Scope in Biomedical Texts. In *Proceedings of BioNLP 2008: Current Trends in Biomedical Natural Language Processing*, pages 38-45, Columbus, Ohio, USA, June 2008.

Anita de Waard and Henk Pander Maat. 2012. Epistemic Modality and Knowledge Attribution in Scientific Discourse: a Taxonomy of Types and Overview of Features. In *Proceedings of the 50th ACL,* pages 47-55, Jeju, Republic of Korea, July 12, 2012.

Janyce Wiebe, Theresa Wilson and Claire Cardie. 2005. Annotating Expressions of Opinions and Emotions in Language. *Language Resources and Evaluation*, volume 39, issue 2-3, pages 163-210.

Empirical Analysis of Aggregation Methods for Collective Annotation

Ciyang Qing, **Ulle Endriss**, **Raquel Fernández** and **Justin Kruger**
Institute for Logic, Language and Computation
University of Amsterdam
{qciyang | justin.g.kruger}@gmail.com
{ulle.endriss | raquel.fernandez}@uva.nl

Abstract

We investigate methods for aggregating the judgements of multiple individuals in a linguistic annotation task into a collective judgement. We define several aggregators that take the reliability of annotators into account and thus go beyond the commonly used majority vote, and we empirically analyse their performance on new datasets of crowdsourced data.

1 Introduction

Human annotation of linguistic resources has become indispensable in computational linguistics, especially with regards to semantic and pragmatic information, which is yet beyond the reach of robust automatic labelling. Most annotation campaigns involve a small group of trained annotators who may not always agree on their judgements. The reliability of the annotation is typically assessed by quantifying the level of inter-annotator agreement, while the final annotation to be released is consensuated amongst experts. In recent years, however, crowdsourcing methods such Amazon's Mechanical Turk (AMT) have shaken up this scenario by making it possible to rapidly recruit large numbers of untrainned annotators at a low cost. This offers great opportunities—in particular, if we consider that the community of speakers is the highest authority regarding linguistic knowledge—but also creates several challenges: amongst others, how to obtain good quality annotations from untrainned and unmonitored individuals, and how to combine large numbers of possibly conflicting judgements into a single joint annotation. In this paper we focus on the latter challenge. Our aim is to investigate and empirically test methods for aggregating the judgements of large numbers of individuals in a linguistic annotation task conducted via crowdsourcing into a *collective judgement*.

Most researchers who turn to crowdsourcing to collect data use majority voting to combine the participants' responses (Sayeed et al., 2011; Zarcone and Rüd, 2012; Venhuizen et al., 2013). Although in the limit it makes sense to take the judgement of the majority as reflecting the view of the community, in practice we cannot reach out to the full population of speakers, which means that the possible biases amongst the participants we manage to recruit may distort the outcome. Also, given the nature of crowsourcing (rewarding speed rather than quality), some participants may not respond truthfully according to their intuitions as speakers. To address these issues, we propose aggregation methods that go beyond majority voting by taking into account the reliability of individual annotators at the time of aggregation.[1] Our approach is related to existing work on analysing the quality of annotated data by examining, for instance, (dis)agreement patterns amongst annotators (Bhardwaj et al., 2010; Peldszus and Stede, 2013; Ramanath et al., 2013). However, while the main aim of this kind of studies is to gain insight into the difficulty of an annotation task or into the feasibility of using untrainned annotators for particular tasks, our focus is on exploiting patterns of judgements for the purpose of aggregation into a single collective annotation—an aspect that has received far less attention in the literature.

We make the following contributions: (i) we make available two new datasets of judgements gathered with AMT for two multi-category annotation tasks; (ii) we define several aggregation methods based, on

[1] Other aspects can contribute to reduce the shortcomings of crowdsourcing at earlier stages, such as task design and annotator recruiting constraints. However, here we specifically deal with improving quality at the time of aggregation.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1533–1542, Dublin, Ireland, August 23-29 2014.

the one hand, on an approach by inspired by social choice theory (Endriss and Fernández, 2013; Kruger et al., 2014), and on the other hand, on probabilistic generative models pioneered by Dawid and Skene (1979); and (iii) we systematically evaluate the performance of the proposed methods on three different annotation tasks.[2]

The paper is structured as follows: In the next section, we introduce our aggregation methods. In Section 3, we evaluate their performance on different datasets and analyse the results. We then examine two further aspects: the impact of the number of annotators in Section 4 and the presence of highly unreliable annotators in Section 5. We conclude in Section 6 with plans for future work.

2 Aggregation Methods

In this section, we define several methods for deriving a collective judgement in a linguistic annotation task from a set of individual annotations. We focus on simple classification tasks where collecting these individual annotations via a crowdsourcing platform is feasible.

2.1 Notation and Terminology

In our model, an *annotation task* consists of three finite sets: the *items* J, the *categories* K, and the *annotators* N. Each annotator is asked to label some of the items with a category. A *group annotation* A is an $|N| \times |J|$ matrix, with a_{ij} representing the category $k \in K$ that annotator $i \in N$ assigned to item $j \in J$. Let N_j denote the set of annotators who annotated item j (i.e., a_{ij} is undefined if $i \notin N_j$).

We want to aggregate the information contained in a group annotation into a single *collective annotation* that assigns a category to each item. An *aggregator* is a function F that maps a group annotation A into a collective annotation $F(A)$, a vector of categories with dimensionality $|J|$ labelling every item with a category. The most widely used aggregator is the *simple plurality rule* (SPR)—known as *simple majority* in the two-category case—which returns a collective annotation where each item j is labelled with the category chosen most often for j by the group, i.e., $\text{SPR}(A)_j \in \text{argmax}_{k \in K} |\{i \in N_j \mid a_{ij} = k\}|$. Since the SPR may lead to a tie, if we require a single category for each item, a tie-breaking method (such as random tie-breaking) must be adopted. For the purposes of this paper, we assign the special category *'undecided'* whenever an aggregator produces a tie (this is reasonable also in practice: we would not want to commit to a randomly chosen category for an annotated linguistic resource).

2.2 Frequency-based Aggregation

In previous work, we introduced (Endriss and Fernández, 2013) and further refined (Kruger et al., 2014) a framework for deriving a collective annotation inspired by social choice theory. They propose so-called *bias-correcting rules* (BCR's), which try to take the reliability of annotators into account by considering the *frequencies* with which annotators choose certain categories. For example, if annotator i uses category k very often, then this might be a sign that i is overusing k and we should give her votes for k less weight. However, if k is also a frequent choice of the population of annotators at large, then this might again temper that effect.

For a given group annotation A, define the *individual frequency* of annotator i choosing category k—$\text{Freq}_i(k)$—as the number of times i chooses k, divided by the total number of items she annotates. Define the *global frequency* of k—$\text{Freq}(k)$—as the number of times k is chosen by someone, divided by the total number of individual annotations. Thus, if $\text{Freq}_i(k)$ is high, particularly if $\text{Freq}_i(k) > \text{Freq}(k)$, we may want to give a relatively low weight to any instance of annotator i choosing catgeory k.

Every BCR defines a family of weights w_{ik}, specifying for each annotator $i \in N$ and each category $k \in K$ how much weight to give to i's choice of k:

$$F_w(A)_j \quad \in \quad \operatorname*{argmax}_{k \in K} \sum_{i \in N_j | a_{ij} = k} w_{ik}$$

[2]The new datasets and an implementation of our aggregation methods are available at `http://www.illc.uva.nl/Resources/CollectiveAnnotation/`.

Diff	*difference-based BCR*	$w_{ik} = 1 + \text{Freq}(k) - \text{Freq}_i(k)$	**Rat**	*ratio-based BCR*	$w_{ik} = \text{Freq}(k)/\text{Freq}_i(k)$
Com	*complement-based BCR*	$w_{ik} = 1 + 1/\lvert K\rvert - \text{Freq}_i(k)$	**Inv**	*inverse-based BCR*	$w_{ik} = 1/\text{Freq}_i(k)$

Table 1: Weights used for canonical Bias Correcting Rules.

In case of a tie, we assign category *'undecided'*. Table 1 defines the weights for four specific BCR's. Thus, for example, if an annotator uses k in 50% of the cases, while the general population only uses k in 20% of all cases, then under Diff she has weight 0.7 whenever she chooses k. Note that Com and Inv do not take global frequencies into account, while Diff and Rat do.

2.3 Agreement-based Aggregation

Suppose each item has a *true* (but unknown) category (its *gold standard*). We may view an annotator's judgement as a noisy signal of the gold standard. We now want to design an aggregator as a maximum likelihood estimator for this ground truth. This approach has been pioneered by Dawid and Skene (1979). Variants have been used for diverse purposes by, amongst others, Snow et al. (2008), Carpenter (2008), Raykar et al. (2010), Ipeirotis et al. (2010), Li et al. (2013), and Passonneau and Carpenter (2013).

Let $p(a_{ij} = k \mid g_j = k^\star)$, with k not necessarily distinct from $k^\star$, be the probability of agent $i \in N_j$ annotating item j with category $k \in K$, given that the gold standard category of j is $k^\star \in K$. If we can obtain estimates of these probabilities, then we can use them to calibrate the weights of the annotators. The challenge, particularly for multi-category annotation tasks, is that the number of probabilities to estimate is fairly large (in particular, it is quadratic in $|K|$). To be able to provide reasonable estimates, we need a large amount of data *from every individual annotator*. But this precisely we do not have in crowdsourcing: we have a lot of data, but it comes from many different annotators. We thus make two simplifying assumptions, aimed at aggressively reducing the number of parameters to estimate:[3]

(1) We assume that $p(a_{ij} = k^\star \mid g_j = k^\star)$, i.e., annotator i's probability of choosing the *correct* category, does not depend on either j or $k^\star$. It only depends on i's accuracy. Thus, we can abbreviate $\text{acc}_i := p(a_{ij} = k^\star \mid g_j = k^\star)$.

(2) We assume that when annotator i does not choose the correct category $k^\star$, then she is equally likely to pick any of the *wrong* categories $k \neq k^\star$: $p(a_{ij} = k \mid g_j = k^\star) = \frac{1-\text{acc}_i}{|K|-1}$.

Assumption (1) is not uncommon (Li et al., 2013), but it clearly is a limiting assumption: accuracy not depending on j means that we cannot model the fact that some items are more difficult to label correctly; accuracy not depending on k means that we cannot model the fact that some categories are harder to comprehend than others. Assumption (2) and its alternatives only come into play when there are more than two categories; as large parts of the literature focus on the two-category case, this issue has received less attention. One of the limitations of assumption (2) is that we cannot model that some categories may "look similar" and are likely to get confused with each other.

On the positive side, in our simplified model we only have a single parameter to estimate for each annotator, namely its accuracy acc_i. Now suppose, hypothetically, we knew the acc_i's (which we do not in practice). Which category should we pick for item j? To answer this question we need to consider probabilities such as $p(g_j = k \mid A_j)$, the probability that k is the true category for item j given our observation of column A_j. If we do not want to make any assumptions regarding possible priors for either gold standards or annotation biases (i.e., if we opt for the default assumption of uniform priors), then we can instead work with $p(A_j \mid g_j = k)$. Specifically, we should choose k over k' if $p(A_j \mid g_j = k) > p(A_j \mid g_j = k')$, i.e., if:

$$\prod_{i|a_{ij}=k} \text{acc}_i \prod_{i|a_{ij}=k'} \frac{1-\text{acc}_i}{|K|-1} \prod_{i|a_{ij}\notin\{k,k'\}} \frac{1-\text{acc}_i}{|K|-1} > \prod_{i|a_{ij}=k'} \text{acc}_i \prod_{i|a_{ij}=k} \frac{1-\text{acc}_i}{|K|-1} \prod_{i|a_{ij}\notin\{k,k'\}} \frac{1-\text{acc}_i}{|K|-1}$$

$$\prod_{i|a_{ij}=k} \frac{(|K|-1)\cdot\text{acc}_i}{1-\text{acc}_i} > \prod_{i|a_{ij}=k'} \frac{(|K|-1)\cdot\text{acc}_i}{1-\text{acc}_i}$$

[3] That is, we are trading generality of the model against estimation quality of its parameters (see also Section 3.4).

Taking logarithms on both sides, we see that giving each annotator a weight of $\log \frac{(|K|-1)\cdot \text{acc}_i}{1-\text{acc}_i}$ results in an optimal aggregator. Let us call the corresponding aggregator the *oracle rule* **Ora**. Importantly, this is not a practically useful rule, as in reality we do *not* know the acc_i's. As we shall see, however, it is a useful benchmark, as it allows us to distinguish between loss in quality due to the simplicity of our model and loss in quality accrued during estimation (given that Ora is perfect w.r.t. the latter dimension).[4]

In practice, we need to estimate the acc_i's. We use a particularly simple method and estimate acc_i as i's *agreement* agr_i with the SPR, defined as follows:[5]

$$\text{agr}_i \quad := \quad \frac{|\{j \in J \mid a_{ij} = \text{SPR}(A)_j\}| + 0.5}{|\{j \in J \mid i \text{ annotates } j\}| + 1}$$

We call the rule we obtain using this method, i.e., the rule giving weight $\log \frac{(|K|-1)\cdot \text{agr}_i}{1-\text{agr}_i}$ to annotator i, the *agreement-based rule* **Agr**. There are two natural refinements of Agr one might consider. First, we could attempt to take priors regarding gold standards into account. If $p(k)$ is the prior probability of encountering (true) category k, then we get $p(g_j = k \mid A_j) \propto p(A_j \mid g_j = k) \cdot p(k)$. This corresponds to adding $\log p(k)$ as an extra weight in favour of category k. We can estimate $p(k)$ using either $\text{Freq}(k)$ or the SPR. The second possible refinement is to iterate the process used to estimate acc_i, i.e., to use Agr in place of SPR to compute better estimates agr'_i of acc_i, and so forth. That is, we could use the EM algorithm (Dawid and Skene, 1979) to estimate acc_i. As we shall see, Agr outperforms both of these refinements for the datasets considered in this paper.

3 Performance on Different Datasets

In this section, we evaluate the performance of our aggregation methods on three datasets from three different categorical annotation tasks for which gold standard annotations are readily available. One of these tasks—Recognising Textual Entailment—is a binary classification task and includes non-expert annotations collected by Snow et al. (2008). The other two tasks—Preposition Sense Disambiguation and Question Dialogue Acts—are multi-category tasks for which we have collected new crowdsourced annotations for the purposes of the present study.[6]

3.1 Recognising Textual Entailment (RTE)

This dataset is based on the task proposed by Dagan et al. (2006) in the PASCAL Recognizing Textual Entailment (RTE) Challenge. The RTE task involves deciding whether the meaning of a sentence (the *hypothesis*) can be inferred from a *text*. The original RTE1 Challenge testset consists of 800 text-hypothesis pairs (e.g., T:*"In central Antioquia two ranges of the Colombian Andeas meet"*, H:*"Antioquia is in Colombia."*) with a gold standard annotation that classifies each of them as either *true* (1) or *false* (0), depending on whether H can be inferred from T or not. The released expert annotation is perfectly balanced, with 400 items annotated as 0 and 400 as 1.

Snow et al. (2008) used Amazon's *Mechanical Turk* (AMT) to collect 10 non-expert annotations for each of the 800 items. The annotation task included a total of 164 AMT workers who annotated between 20 items (124 annotators) and 800 items each (only one annotator). Amongst the non-expert annotations, category 1 is slightly more frequent ($\approx 57\%$) than category 0.

Table 2a shows the results of applying the aggregation rules (and the oracle rule) to this data. Here (as later in Tables 2b and 2c), the first columns shows observed agreement (A) between the collective annotation output by each rule and the gold standard.[7] The following columnss show precision and recall for each category. We can see that all rules outperform the SPR.[8] Agr yields better results (93.3%)

[4]Snow et al. (2008) used Dawid and Skene's model to calibrate annotator judgements in terms of the gold standard. In contrast, we only use Ora as a benchmark to get a better understanding of the limitations of our probabilistic model.

[5]The smoothing terms (0.5 and 1) ensure that agr_i will never be 0 or 1, i.e., $\log \frac{(|K|-1)\cdot \text{agr}_i}{1-\text{agr}_i}$ is always well-defined.

[6]For practical reasons, we have opted for evaluating our methods against a gold standard. However, we note that in linguistic tasks, especially those concerning semantics and pragmatics, there may simply not be a 'true' category—a collective annotation may be the closest we can get to representing the view of the community.

[7]All aggregators assign category *'undecided'* in case of a tie. Therefore, any ties are counted as instances of disagreement.

[8]The SPR leads to 65 ties; the other rules lead to none.

	A	0	1
SPR	0.856	.96/.79	.91/.93
Com	0.916	.93/.90	.91/.93
Inv	0.893	.87/.92	.91/.87
Diff	0.915	.94/.88	.89/.95
Rat	0.908	.94/.88	.88/.94
Agr	0.933	.93/.93	.93/.94
[Ora]	0.941	.93/.96	.96/.93

(a) RTE

	A	1	2	3
SPR	0.813	.89/.96	.82/.40	.82/.92
Com	0.820	.87/.95	.70/.46	.82/.92
Inv	0.807	.88/.95	.62/.51	.82/.85
Diff	0.833	.86/.96	.80/.46	.82/.93
Rat	0.840	.87/.96	.81/.49	.82/.93
Agr	0.827	.85/.98	.88/.40	.80/.93
[Ora]	0.833	.85/.98	.88/.43	.81/.93

(b) PSD

	A	1	2	3	4
SPR	0.857	.86/.98	.87/1.0	.92/.75	.90/.42
Com	0.870	.87/.98	.87/1.0	.88/.77	.88/.49
Inv	0.877	.91/.91	.94/.98	.84/.77	.72/.73
Diff	0.867	.84/.98	.87/1.0	.89/.78	.91/.44
Rat	0.870	.84/.99	.87/1.0	.92/.77	.91/.47
Agr	0.867	.84/.99	.87/1.0	.92/.77	.91/.44
[Ora]	0.870	.85/.99	.87/1.0	.92/.77	.91/.47

(c) QDA

Table 2: Observed agreement with the gold standard and precision/recall per category for each task.

than any of the BCR's in this case. For the SPR, category 1 has higher recall than precision, while the opposite is the case for category 0. This is in line with the slightly higher frequency of category 1 in the AMT annotations. The BCR's should be able to correct for this bias and to some extent they do (note the increase in category 0's recall: 88% or higher for any of the BCR's vs. 79% for the SPR). In this dataset, the best-performing BCR is Com (91.6% agreement), keeping a good balance between precision and recall for both categories. If we use the refinement of Agr with priors, then the observed agreement drops slightly (to 92.9% if we estimate gold standard distributions using Freq(k), and to 93.1% if we use the SPR). If we use the EM algorithm to estimate acc_i, the system stabilises after six iterations and the resulting rule also does slightly worse than Agr (93.0%).

3.2 Preposition Sense Disambiguation (PSD)

This annotation task is based on the dataset used in the SemEval 2007 task on word-sense disambiguation of prepositions (Litkowski and Hargraves, 2007). The SemEval dataset consists of roughly 25,000 sentences each containing one of the 34 most common English prepositions. The gold standard annotation was constructed by a single lexicographer who tagged each preposition instance with a sense from the sense inventories given by the Oxford Dictionary of English (ODE).

For our non-expert data collection, we used the 150 sentences with the preposition *among*, which according to ODE has four senses. We simplified the task by collapsing senses 3 and 4, as there is only one item classified with sense 4 by the gold standard and that sense is closest to sense 3.[9] The annotation task was conducted using AMT. We showed the workers the following sense definitions of *among* and asked them to select the appropriate sense for each sentence:

(1) situated more or less centrally in relation to other things, e.g., *"There are flowers hidden among the roots of the trees."*

(2) being a member of a larger set, e.g., *"Snakes are among the animals most feared by man."*

(3) shared by some members of a group or community, e.g., *"Members of the government bickered among themselves."*

The distribution of categories according to the gold standard is 37.3%, 23.3%, and 39.3% for sense 1, 2, and 3, respectively. The non-expert annotation task included 45 AMT workers who annotated between 15 items (26 annotators) and 150 items each (only one annotator; another annotated 135 items). Amongst the AMT annotations, the relative frequency of the categories is 40.6%, 18.8%, and 40.6%, respectively.

The results are shown in Table 2b.[10] The rules with the highest agreement with the gold standard are Diff (83.3%) and Rat (84%), i.e., the rules that take into account the global frequency of the categories. Rat outperforms not only the other three BCR's and the SPR (81.3%) but also Agr (82.7%) and Ora (83.3%). Recall for sense 2 (the rarest category) is low across rules, although less so for the BCR's, which manage to correct slightly for the annotators' bias against this category.[11]

[9] The original ODE sense definitions for *among* can be found at `http://tinyurl.com/ode-among`.

[10] The SPR leads to 6 ties; the other rules lead to none. The two refinements of Agr (priors and EM) do not affect the outcome.

[11] After inspecting the data, we suspect that the gold standard overuses sense 2. For instance, in the folowing sentence *among* is tagged with sense 2 although sense 1 seems more appropriate: "[...] *like icebergs 90 per cent is under the water and that is making them incredibly difficult to see* **among** *the waves."*

3.3 Question Dialogue Acts (QDA)

The second dataset we collected is based on the Switchboard corpus (Godfrey et al., 1992). The corpus includes a gold standard annotation prepared by trained annotators, labelling each utterance with a dialogue act tag from the SWBD-DAMSL annotation scheme (Jurafsky et al., 1997).

For our crowdsourcing experiment, we restricted ourselves to four types of question dialogue acts: Yes-No questions, Wh-questions, Declarative questions (including both declarative wh- and yes-no questions), and Rhetorical questions. We extracted 300 questions from the corpus, 35% of which were annotated as Yes-No in the gold standard, 30% as Wh, 20% as Declarative, and 15% as Rhetorical. The AMT workers were shown the following category definitions (here slightly simplified for space reasons):

(1) Yes-No: Questions with a standard form that could be answered with "yes" or "no" (*"Is that the only pet that you have?"*)

(2) Wh: Questions with a standard form that ask for specific information using wh-words (*"What kind of pet do you have?"*)

(3) Declarative: Questions with a statement-like form that nevertheless ask for an answer (*"You have how many pets."*)

(4) Rhetorical: Questions that do not need to be answered. They can have the form of any of the question types above, but they are asked only to make a point (*"If I ever wanted to have a pet, how could I work?"*)

Each item consists of a short dialogue fragment showing three utterances before and after the question to be annotated. The AMT workers were asked to classify the highlighted question with one of the four question types above. Here is a sample item (with reduced context for space reasons):

A: I understand.
A: **Where is home for you?**
B: Originally, was born in Missouri.

A total of 63 AMT workers participated in the annotation task, annotating between 10 items (24 annotators) and 200 items each (only one annotator). Amongst these non-expert annotations, the relative frequencies for category 1 to 4 are 36.6%, 34.1%, 18.4%, and 10.9%, respectively.

Table 2c shows the results of applying the aggregation rules to this data, plus the outcome of the oracle. [12] Inv yields the best result (87.7%), even outperforming Ora (87%). The annotators tend to overuse the common categories (1 and 2), resulting in high recall but low precision. In contrast, the less frequent categories (3 and 4) tend to be underused, resulting in high precision but low recall. Note how applying Inv leads to particularly high recall for rhetorical questions (category 4). The price to pay is the drop in precision for this category compared to the other rules. The dual effect is that precision for Yes-No (1) and Wh (2) is higher with Inv than with the other rules, while recall is lower.

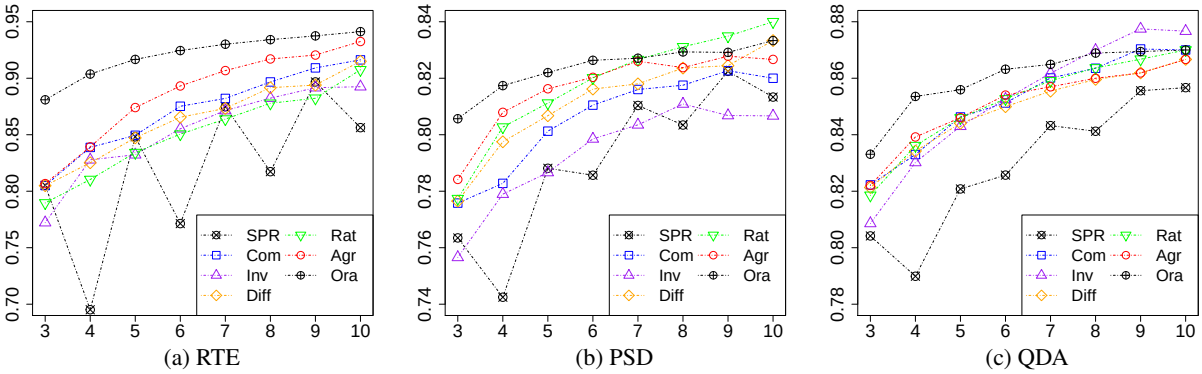
3.4 Comparative Analysis

First, let us compare Agr and Ora. The good performance of Agr suggests that our simple probabilistic model is not *too* simplistic; the trade-off between loss in generality and gain in ability to estimate parameters mentioned in Section 2 appears to be appropriate. The fact that Ora outperforms Agr only slightly suggests that the number of parameters in our model is sufficiently small to be estimated well using the amount of data typically available in linguistic annotation taks conducted via crowdsourcing.

Second, the fact that Agr (modestly) outperforms its refinement using an estimated prior can be explained by the fact that, in our datasets, annotators tend to overuse frequent categories and underuse rare categories. The reason why iterating the rule used to estimate accuracies did not improve performance of Agr for our datasets is less clear, but may be related to the well-known fact that EM can get stuck in a local optimum. The positive take-away message is that the simplest form of our agreement rule resulted in the best performance (at least for our three datasets).

Third, the differences in performance between different BCR's point at an interesting difference in types of bias. Recall that Com and Inv judge the reliability of an annotator only in terms of her own annotations and penalise frequent use of a category. Diff and Rat correct for this effect in case the global frequency is high as well. This means that if a population of annotators has a *shared bias* against or in favour of a category, then Diff and Rat cannot track this well. This explains the fact that Com outperforms Diff and Inv outperforms Rat in the QDA data (see Table 2c): in this task many annotators appeared to

[12] The SPR leads to 7 ties; the other rules to none. Once again, the observed agreement for Agr drops slightly for the two refinements discussed (priors and EM).



(a) RTE (b) PSD (c) QDA

Figure 1: Observed agreement with the gold standard (y-axis) for varying NAI (x-axis).

have difficulties recognising rhetorical questions, i.e., they had a shared bias against labelling an item as Rhetorical. For a dataset with clear *individual biases*, on the other hand, we would expect Diff/Rat to outperform Com/Inv. We do not have a clear case of such a phenomenon in the data analysed here. For the PSD task, Diff/Rat do outperform Com/Inv (see Table 2b), but we believe that the explanation for this finding is a different one: Arguably, the gold standard annotation overuses category 2 (see Footnote 11). This means that high-quality annotators are seen as underusing it and get penalised by Com/Inv. For Diff/Rat this effect is tempered by the fact that the population as a whole is underusing category 2 (relative to the questionable gold standard).

Finally, much can be learned from contraposing the frequency- and agreement-based approach. Suppose the gold standard is uniformly distributed (as for RTE). Then the expected value of $\text{Freq}_i(k)$ is $\frac{1}{|K|}$, i.e., it does not depend on acc_i at all. Thus, the two approaches track entirely different parameters, yet both achieve respectable results. This suggests that combining them might prove fruitful (see Section 5). Certainly, an approach based on a richer probabilistic model would be able to track both kinds of parameters, but as we had argued, this might be infeasible with the relatively small amount of data per annotator we can collect through crowdsourcing. In some sense, what we have done with our rules is trying to make up for the scarcity of data by exploiting our domain knowledge (e.g., regarding the relationships between observed frequency and annotator reliability) to reduce the parameter space.

4 Impact of Number of Annotators

The cost and quality of an annotated linguistic resource created via crowdsourcing crucially depends on the number of annotators that label each item. Having low numbers of coders will make the task more affordable (in terms of time and money), but it will also make the aggregation process more vulnerable to low-quality annotators. Snow et al. (2008) showed how the number of annotators per item (henceforth NAI) influences the performance of the SPR. Here we further explore the impact of NAI on the quality of the collective annotation obtained by different aggregation methods.

For each of the three datasets and each NAI n ($3 \leqslant n \leqslant 9$), we randomly resampled n annotations for each set of items presented to a worker in one go (i.e., for each HIT in AMT terminology). This allowed us to generate a subset of the original dataset with n annotators per item. We generated 1000 such random subsets for each n, applied our aggregators to each subset (and also computed the oracle outcome). We then calculated the average observed agreement with the gold standard. To test whether the differences observed are statistically significant, we calculated the difference in performance between pairs of rules on each subset and computed the 95% (one-sided) confidence intervals by using its distribution over the 1000 subsets. If the proportion of subsets on which this difference is strictly greater than 0 is higher than 95%, we consider the difference to be significant.

The results are shown in Figure 1. We can see that, as the NAI increases, the performance of the rules generally improves (except for the oscillation of the SPR due to tie-breaking). This improvement is greater when the NAI is small (from 3 to 5), which suggest that a minimum of 5 annotators per item is

	0	6
SPR	0.856	0.911
Com	0.916	0.930
Inv	0.893	0.933
Diff	0.915	0.928
Rat	0.908	0.926
Agr	0.933	0.929
[Ora]	0.941	0.944

(a) RTE

	0	9
SPR	0.813	0.820
Com	0.820	0.840
Inv	0.807	0.840
Diff	0.833	0.820
Rat	0.840	0.833
Agr	0.827	0.827
[Ora]	0.833	0.827

(b) PSD

	0	6
SPR	0.857	0.867
Com	0.870	0.883
Inv	0.877	0.903
Diff	0.867	0.873
Rat	0.870	0.877
Agr	0.867	0.867
[Ora]	0.870	0.883

(c) QDA

Table 3: Effect on observed agreement when removing 6 spammers in RTE, 9 in PSD, and 6 in QDA.

recommended. We can also observe that Agr has a robust performance on all datasets when the NAI is between 5 and 7: its improvement over the SPR is statistically significant in all cases for the three tasks, except on PSD when NAI is 7, in which case it is neither significantly better nor significantly worse than the SPR. Note that in all datasets Agr only needs 6 or 7 annotators per item to achieve an accuracy comparable to the SPR using 10 annotators per item.

The robustness of Agr with low NAI is not surprising, given that it already assigns low weights to workers who consistently disagree with the majority. Discounting such problematic workers is particularly important when there are relatively few workers per item. But as the NAI increases, it becomes more likely that random annotators will cancel each other out. It is then that we observe the greatest advantage of using BCR's. This can be seen in the plots for PSD and QDA with high NAI. In those cases the improvement of the best performing BCR's (Rat on PSD and Inv on QDA) over the other rules approaches significance although does not reach the 95% threshold (e.g., on QDA when the NAI is 9, Inv is strictly better than Agr for 93.4% of the subsets).

5 Removal of Low-Quality Annotators

Next we discuss how removing easily recognisable low-quality annotators ("spammers") before aggregation affects the quality of results. The BCR's make the implicit assumption that annotators are sincere. This can be problematic, given the nature of crowdsourcing, where it is not uncommon to encounter workers giving random rather than truthful responses (Sheng et al., 2008; Raykar and Yu, 2012). BCR's are vulnerable to this phenomenon. Here we propose to combine the frequency- and agreement-based approach by using the agreement rate of an annotator with the SPR outcome to identfy and remove spammers prior to applying the frequency-based BCR's.

We take *spammers* to be those annotators that annotate a large number of items (i.e., we have sufficient evidence to judge) and that systematically deviate from the plurality outcome. In the specific context of our datasets, we have implemented this idea by labelling as spammers those annotators who annotated at least 20% of the total number of items and whose agreement rate with the SPR is below the median agreement rate. This corresponds to 6 annotators in the RTE dataset, 9 in the PSD dataset, and 6 in the QDA dataset. The effect of removing these low-quality annotators from the population can be seen in Table 3 showing observed agreement of the different aggregation rules (and the oracle rule) with the gold standard before and after spammer removal.

The results show that, with one exception, after removing spammers the performance of the BCR's improves significantly. The exception concerns Diff and Rat for the PSD dataset. Recall that the gold standard for this dataset, arguably, overuses category 2 (see Footnote 11 and Section 3.4). That is, high-quality annotators are (wrongly) judged to be underusing category 2. Before spammer removal, this effect is tempered by the presence of a few annotators delivering 'random' annotations (thereby artificially increasing the frequency of category 2). After spammer removal, this positive effect is diminished and rules such as Diff and Rat suffer in performance. Con and Inv, on the other hand, can compensate for this effect simply by giving very high weights to those (high-quality) annotators who still use the relatively rare category 2. Also for RTE and QDA, amongst the BCR's the rules not based on global frequencies, i.e., Com and Inv, benefit most. Indeed, after spammer removal Com/Inv perform better than Diff/Rat for all three datasets. Overall, Inv with spammer removal is our best-performing rule.

Not surprisingly, Agr and Ora gain relatively little from spammer removal since, given our definition of a spammer, the removed annotators already had very low weights to begin with. In fact, the performance of these aggregation rules may even drop slightly after removing spammers (see Tables 3a and 3b).

6 Conclusions

We have argued that simply using the majority/plurality rule to aggregate individual linguistic judgments in a crowdsourcing annotation task is far from optimal. Instead, we have proposed several methods that weight the annotators' judgements by exploiting either the frequency with which they choose particular categories or the degree to which they agree with the full population of annotators. We have tested our methods on existing datasets and we have also created two new datasets. Our results show how annotation tasks with different characteristics can benefit from different types of aggregation methods. Our aggregation methods result in small but robust gains across datasets, both in terms of accuracy achieved and in terms of the number of annotators required to obtain acceptable results.

Besides BCR's, in our previous work we also proposed a *greedy consensus rule*, albeit only for the two-category case (Endriss and Fernández, 2013) . This rule sequentially locks in simple majorities in the order of relative majority strength, but along the way disregards annotators who disagree with too many of those strong majorities. It performs well on the RTE dataset (almost as well as Agr). Intuitively speaking, it can track *item difficulty*, by first settling the easy items (with clear majorities) and thereby learning which annotators are most reliable to then have them decide on the harder items. Here we have not included this rule as there is no single most natural way of generalising it to the multi-category case. Arriving at such a generalisation in a principled manner is an important direction for future work.

It would also be interesting to get a clearer understanding of the links between methods for assessing inter-annotator agreement (Artstein and Poesio, 2008) and methods of aggregation (i.e., methods that may be applied to data of possibly rather poor inter-annotator agreement, as is the case for parts of our datasets). A relevant observation in this context is that the notions of individual and global frequency at the core of our BCR's also play a role in agreement coefficients, namely to compute chance agreement: π (Scott, 1955) uses global frequencies and κ (Cohen, 1960) uses individual frequencies.

While the definition of Agr was motivated by a simple probabilistic model, the BCR's were motivated by rules of thumb regarding links between observed frequencies and reliability. We have noted before that the BCR's do not track the same phenomena as Agr; rather, they seem to complement each other, an observation we have exploited explicitly when removing spammers before applying a BCR. Identifying a suitable probabilistic model for our frequency-based BCR's promises to be a fruitful future line of research, as it would allow for a better comparison (and eventually integration) of the two approaches.

References

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Vikas Bhardwaj, Rebecca J Passonneau, Ansaf Salleb-Aouissi, and Nancy Ide. 2010. Anveshan: a framework for analysis of multiple annotators' labeling behavior. In *Proc. 4th Linguistic Annotation Workshop*, pages 47–55. ACL.

Bob Carpenter. 2008. Multilevel Bayesian Models of Categorical Data Annotation. Technical report, LingPipe.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges*, volume 3944 of *LNCS*, pages 177–190. Springer-Verlag.

Alexander P. Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28.

Ulle Endriss and Raquel Fernández. 2013. Collective annotation of linguistic resources: Basic principles and a formal model. In *Proc. 51st Annual Meeting of the Association for Computational Linguistics (ACL-2013)*, pages 539–549.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. SWITCHBOARD: Telephone Speech Corpus for Research and Devlopment. In *Proc. IEEE Conference on Acoustics, Speech, and Signal Processing*, pages 517–520.

Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality Management on Amazon Mechanical Turk. In *Proc. 2nd Human Computation Workshop (HCOMP-2010)*.

Dan Jurafsky, Elizabeth Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function-annotation coder's manual, draft 13. Technical Report TR 97-02, Institute for Cognitive Science, University of Colorado at Boulder.

Justin Kruger, Ulle Endriss, Raquel Fernández, and Ciyang Qing. 2014. Axiomatic analysis of aggregation methods for collective annotation. In *Proc. 13th Int'l Conference on Autonomous Agents and Multiagent Systems (AAMAS-2014)*, pages 1185–1192. IFAAMAS.

Hongwei Li, Bin Yu, and Dengyong Zhou. 2013. Error rate analysis of labeling by crowdsourcing. In *Proc. Machine Learning meets Crowdsourcing, Workshop at the Int'l Conference on Machine Learning (ICML-2013)*.

Kenneth C. Litkowski and Orin Hargraves. 2007. SemEval-2007 Task 06: Word-Sense Disambiguation of Prepositions. In *Proc. 4th Int'l Workshop on Semantic Evaluations (SemEval-2007)*.

Rebecca J. Passonneau and Bob Carpenter. 2013. The benefits of a model of annotation. In *Proc. 7th Linguistic Annotation Workshop*, pages 187–195. ACL.

Andreas Peldszus and Manfred Stede. 2013. Ranking the annotators: An agreement study on argumentation structure. In *Proc. 7th Linguistic Annotation Workshop*, pages 196–204. ACL.

Rohan Ramanath, Monojit Choudhury, Kalika Bali, and Rishiraj Saha Roy. 2013. Crowd prefers the middle path: A new iaa metric for crowdsourcing reveals turker biases in query segmentation. *Proc. 51st Annual Meeting of the Association for Computational Linguistics (ACL-2013)*, pages 1713–1722.

Vikas Raykar and Shipeng Yu. 2012. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *Journal of Machine Learning Research*, 13:491–518.

Vikas Raykar, Shipeng Yu, Linda Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research*, 11:1297–1322.

Asad Sayeed, Bryan Rusk, Martin Petrov, Hieu Nguyen, Timothy Meyer, and Amy Weinber. 2011. Crowdsourcing syntactic relatedness judgements for opinion mining in the study of information technology adoption. In *Proc. Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH-2011)*.

William A. Scott. 1955. Reliability of content analysis: the case of nominal scale coding. *Public Opinion Quaterly*, 19(3):321–325.

Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proc. 14th ACM Int'l Conference on Knowledge Discovery and Data Mining (KDD-208)*.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP-2008)*, pages 254–263.

Noortje Venhuizen, Valerio Basile, Kilian Evang, and Johan Bos. 2013. Gamification for word sense labeling. In *Proc. 10th Int'l Conference on Computational Semantics (IWCS-2013)*, pages 397–403.

Alessandra Zarcone and Stefan Rüd. 2012. Logical metonymies and qualia structures: An annotated database of logical metonymies for German. In *Proc. Language Resources and Evaluation Conference (LREC-2012)*, pages 1799–1804.

Annotation Adaptation and Language Adaptation in NLP

Qun Liu
CNGL Centre for Global Intelligent Content
National Centre for Language Technology
School of Computing
Dublin City University
Dublin, Ireland
qliu@computing.dcu.ie

Invited Speaker Abstract

Adaptation technologies are always useful in NLP when there is discrepancy between the training scenario and use scenario. They are also effective in alleviating the data scarcity problem. Domain adaptation is the most popular kind of adaptation technologies and is intensively researched. In this talk we will introduce two other kinds of adaptation technologies: annotation adaptation and language adaptation. Annotation adaptation is used to improve the performance of an automatic annotation task by leveraging corpora with different annotation schemas, while language adaptation is used to solve an NLP problem in one language by utilizing the linguistic knowledge which is learnt from solving the same problem in another language. We investigate these technologies mainly for the tasks of word segmentation and parsing, however similar technologies may be developed for other NLP tasks also.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, page 1543, Dublin, Ireland, August 23-29 2014.

Collective Named Entity Disambiguation using Graph Ranking and Clique Partitioning Approaches

Ayman Alhelbawy[*†] and **Robert Gaizauskas** [*]
[*]The University of Sheffield, Regent Court, 211 Portobello Street, Sheffield, S1 4DP, U.K
[†]Faculty of Computers and Information, Fayoum University, Fayoum, Egypt
ayman,R.Gaizauskas@dcs.shef.ac.uk

Abstract

Disambiguating named entities (NE) in running text to their correct interpretations in a specific knowledge base (KB) is an important problem in NLP. This paper presents two collective disambiguation approaches using a graph representation where possible KB candidates for NE textual mentions are represented as nodes and the coherence relations between different NE candidates are represented by edges. Each node has a local confidence score and each edge has a weight. The first approach uses Page-Rank (PR) to rank all nodes and selects a candidate based on PR score combined with local confidence score. The second approach uses an adapted Clique Partitioning technique to find the most weighted clique and expands this clique until all NE textual mentions are disambiguated. Experiments on 27,819 NE textual mentions show the effectiveness of both approaches, outperforming both baseline and state-of-the-art approaches.

1 Introduction

Named entities (NEs) have received a lot of attention from the NLP community over the last two decades (see, e.g. Nadeau and Sekine (2007)). Most of this work has focussed on the task of recognizing the boundaries of NE mentions in text and classifying them into one of several classes, such as Person, Organization or Location. However, references to entities in the real world are often ambiguous: there is a many-to-many relation between NE mentions in text and the entities denoted by these mentions in the world. For example, the same NE mention "Norfolk" may refer to a person, "Peter Norfolk, a wheelchair tennis player", a place in the United Kingdom, "Norfolk County", or a place in the United States like "Norfolk, Massachusetts"; conversely, one entity many be known by many names, such as "Cat Stevens", "Yusuf Islam" and "Steven Georgiou". The task of named entity disambiguation (NED) is to establish a correct mapping between each NE mention in a document and the entity it denotes in the real world. Following most researchers in this area, we treat entries in a large knowledge base (KB) as surrogates for real world entities when carrying out NED and, in particular, use Wikipedia as the reference KB against which to disambiguate NE mentions. NED is important for tasks like KB population, where we want to extract new information from text about an entity and add it to a pre-existing entry for that entity in a KB, or for information retrieval where we may want to cluster or filter results for different entities with the same textual mentions.

The main hypothesis underlying this work is that different NEs in a document help to disambiguate each other. However, other textual mentions in the document are also ambiguous. So, what is needed is a *collective disambiguation* approach that jointly disambiguates all NE textual mentions.

In our approaches we model each possible candidate for every NE mention in a document as a distinct node in a graph and model candidate coherence by links between the nodes. Figure 1 shows an example of the disambiguation graph for three mentions "A", "B", and "C" found in a document, where the candidate entities for each mention are referred to using the lower case form of the mention's letter together with a distinguishing subscript. The goal of disambiguation is to find a set of nodes where only one candidate is selected from the set of entities associated with each mention, e.g. a_3, b_2, c_2.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1544–1555, Dublin, Ireland, August 23-29 2014.

We propose two different approaches to find the best disambiguation candidates in the graph. The first approach starts by finding the most confident and coherent set of disambiguation entities and iteratively expands this set until all NE textual mentions are disambiguated. The second approach ranks all nodes in the solution graph using the Page-Rank algorithm, then re-ranks all nodes by combining the initial confidence and graph ranking scores. We consider several different measures for computing the initial confidence assigned to each node and several measures for determining and weighting the graph edges. Node linking relies on the fact that the textual portion of KB entries typically contains mentions of other NEs. When these mentions are hyper-linked to KB entries, we can infer that there is some relation between the real world entities corresponding to the KB entries, i.e. that they should be linked in our solution graph. These links also allow us to build up statistical co-occurrence counts between entities that occur in the same context, which may be used to weight edges in our graph.

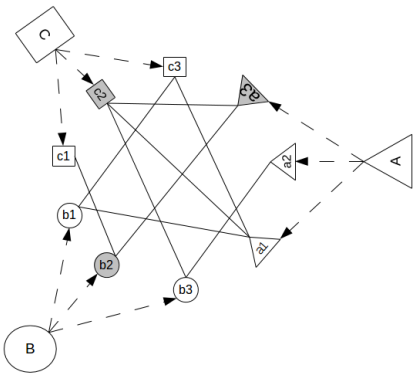


Figure 1: Example of solution graph

We evaluate our approaches on the AIDA dataset (Hoffart et al., 2011). Comparison with the baseline and some state-of-the-art approaches shows our proposed approaches offers substantial improvements in disambiguation accuracy.

The rest of the paper is organized as follows: Section 2 discusses related work. Section 3 discusses selection of NE candidate entities from the Wikipedia knowledge base and the assignment of confidence scores to each candidate. Formulation of the NE disambiguation problem in terms of a graph model is presented in section 4. Sections 5 and 6 describe the clique partitioning and ranking disambiguation approaches for collective NED. The experimental dataset and experimental results are presented in Section 7. Section 8 concludes the paper and presents some suggestions for future work to improve the results.

2 Related Work

Named Entity Disambiguation has received a lot attention in the past few years. Perhaps the best known related work is the Entity Linking (EL) shared task challenge first proposed by the National Institute of Standards and Technology (NIST) as part of the Knowledge Base Population (KBP) track within the Text Analysis Conference (TAC) in 2009 (McNamee and Dang, 2009). EL is a similar but broader task than NED: NED is concerned with disambiguating a textual NE mention where the correct entity is known to be one of the KB entries, while EL also requires systems to deal with the case where there is no entry for the NE in the reference KB. Ji et al. (2011) group and summarise the different approaches to EL taken by participating systems.

In general, there are two main lines of approach to the NED problem. The first, *single entity disambiguation approaches (SNED)*, disambiguates one entity at a time without considering the effect of other NEs. These approaches use local context textual features of the mention and compare them to the textual features of NE candidate documents in the KB, and link to the most similar. The first approach in this line was Bunescu and Pasca (2006), who measure similarity between the textual context of the NE mention and the Wikipedia categories of the candidate. More similarity features were added by Cucerzan (2007) who realized that topical coherence between a candidate entity and other entities in the context will improve NED accuracy by calculating the nodes' coherence based on the their incoming links in Wikipedia and the overlaps in Wikipedia categories. Milne and Witten (2008) improve Cucerzan's work by calculating the topical coherence using Normalized Google Distance and restrict the context entities to the unambiguous entities. Different query expansion approaches are incorporated into this framework, such as using context term expansion (Gottipati and Jiang, 2011) and acronym expansion (Zhang et al., 2011). Sen (2012) proposed a latent topic model to learn the context entity association. Machine learning is widely used in SNED as some approaches deal with the problem as a search result ranking problem. Supervised learn-to-rank models are used to re-rank the ambiguous candidate set (Zheng et al., 2010; Dredze et al., 2010; Alhelbawy and Gaizauskas, 2012; Nebhi, 2013).

The second line of approach is *collective named entity disambiguation (CNED)*, where all mentions of entities in the document are disambiguated jointly. These approaches try to model the interdependence between the different candidate entities for different NE mentions in the query document, and reformulate the problem of NED as a global optimization problem whose aim is to find the best set of entities. As this new formulation is NP-hard, many approximations have been proposed. Kulkarni et. al. (2009) presents a collective approach for entity linking that models the coherence between all pairs of entity candidates for different mentions as a probabilistic factor graph. They present two approximations to solve this optimization problem where the interdependence between decisions is modelled as the sum of the pairs' dependencies. Alhelbawy and Gaizauskas (2013) proposed a sequence dependency model using HMMs to model NE interdependency. Another approximation uses a mixture of local and global features to train the coefficients of a linear ranking SVM to rank different NE candidates (Ratinov et al., 2011). Shirakawa et al. (2011) cluster related textual mentions and assign a concept to each cluster using a probabilistic taxonomy. The concept associated with a mention is used in selecting the correct entity from the Freebase KB.

Graph models are widely used in collective disambiguation approaches. All these approaches model NE interdependencies, while different methods may be used for disambiguation. Han (2011) uses local dependency between NE mention and the candidate entity, and semantic relatedness between candidate entities to construct a referent graph, proposing a collective inference algorithm to infer the correct reference node in the graph. Hoffert (2011) poses the problem as one of finding a dense sub-graph, which is infeasible in a huge graph. So, an algorithm originally used to find strongly interconnected, size-limited groups in social media is adapted to prune the graph, and then a greedy algorithm is used to find the densest graph.

The word sense disambiguation (WSD) task has many similarities to NED, since in both cases the goal is to determine which of a set of predefined senses or reference entities is the correct interpretation of a surface string in context. Many researchers have used graph-based approaches successfully for the WSD task. Sinha and Michalecea (2007) proposed using four different graph centrality algorithms – Indegree, PageRank, Closeness and Betweenness for WSD. We propose to use a clique partitioning algorithm, originally proposed by Born et al. (1973), for NED. Clique algorithms have been successfully used for WSD problems. Gutiérrez et al. (2011; 2012), for example, use an N-cliques graph partitioning technique to identify sets of highly related senses. However, this approach has not been used for NED.

Our second proposed model uses the Page-Rank algorithm (PR), which to our knowledge has also not previously been applied to NED. PR was proposed by Page et al. (1999) to produce a global rank for web pages based on the hyperlink structure of the web. Xing and Ghorbani (2004) adapted PR to take into account the weights of links and the nodes' importance. PR and Personalized PR algorithms have been used successfully in WSD (e.g. Sinha and Mihalcea (2007), Agirre and Soroa (2008; 2009)).

3 Named Entity Candidates Selection

Given an input document D containing a set of pre-tagged NE textual mentions $M = \{m_1, m_2, m_3 \dots m_k\}$, we need to select all possible candidate interpretations for each m_i from the knowledge base. I.e. for each NE textual mention $m_i \in M$ we select a set of candidates $E_i = \{e_{i,1}, e_{i,2}, e_{i,3} \dots e_{i,j}\}$ from the KB. The NE textual mention m_i is used to search the KB entry titles using Lucene[1] to find entries with titles that fully or partially contain the NE textual mention. The following example shows the possible candidates for the textual mention "Sheffield": "Sheffield, New Zealand,", "University of Sheffield", "Sheffield United F.C.", "Sheffield, Massachusetts", "Fred Sheffield", "Sheffield, Alabama", etc. The result of this search is quite large and this increases the likelihood of the correct entry occurring somewhere in the list, i.e. it improves recall. However, the challenge now moves to the disambiguation step. In this step, we need to assign a confidence score to each candidate, as shown in the following section.

[1] https://lucene.apache.org/

3.1 Candidate Confidence Score

For each candidate $e_{i,j}$, a set of initial confidence scores $IConf(e_{i,j})$ is assigned. These scores are calculated for each NE candidate independent of other candidates or the candidates for other NE textual mentions in the document. Three scores are calculated locally using the NE textual mention context. There is also one global confidence score, entity popularity (EP), which is calculated globally independent of the document or the textual mention context by using the Freebase KB (Bollacker et al., 2008). The four confidence scores to be calculated for each NE candidate as follows:

- Cos: The cosine similarity between the NE textual mention and the KB entry title.

- JwSim: While the cosine similarity between a textual mention in the document and the candidate NE title in the KB is widely used in NED, this similarity is a misleading feature. For example, the textual mention "Essex" may refer to either of the following candidates "Essex County Cricket Club" or "Danbury, Essex", both of which are returned by the candidate generation process. The cosine similarity between "Essex" and "Danbury, Essex" is higher than that between "Essex" and "Essex County Cricket Club", which is not helpful in the NED setting. We adopted a new mention-candidate similarity function, $jwSim$, which uses Jaro-Winkler similarity as a first estimate of the initial confidence value for each candidate. This function considers all terms found in the candidate entity KB entry title, but not in the textual mention as disambiguation terms. The percentage of disambiguation terms found in the query document is used to boost in the initial $jwSim$ value, in addition to an acronym check (whether the NE textual mention could be an acronym for a specific candidate entity title). Experiments show that $jwSim$ performs much better than the standard cosine similarity.

- Ctxt: The cosine similarity between the sentence containing the NE mention in the query document and the textual description of the candidate NE in the KB (we use the first section of the Wikipedia article as the candidate entity description).

- EP: Entity popularity refers to connectivity to this entity. It has been used successfully as a discriminative feature for NED (Nebhi, 2013). Freebase provides an API interface to get an entity's popularity score, which is computed during Freebase data indexing. This score is a function of the entity's inbound and outbound link counts in Freebase and Wikipedia[2].

Initial confidence scores are calculated independently for each candidate entity for an NE mention. However, after the initial calculation, initial confidence scores for all candidates for a single NE mention are normalized to sum to 1.

4 Disambiguation Graph Model

In this section we discuss the graph model we use for NED. All candidate entities for the different NE textual mentions in the document are represented as an undirected graph $G = (V, D)$ where V is the node set of all possible candidate entities for different NE textual mentions in the input document and D is the set of edges between nodes. Because the same entity may be found multiple times as a candidate for different textual mentions and each occurrence must be evaluated independently, each node is formed as an ordered pair of textual mention m_i and candidate entity $e_{i,j}$. So, the graph nodes are formulated as a set $V = \{(m_i, e_{i,j}) \mid \forall e_{i,j} \in E_i, \forall m_i \in M\}$.

A set of entities is coherent if real world relations hold between them. We use such relations to link candidate entities for different NE textual mentions in our graph model. Edges are not drawn between different nodes for the same mention. However, they are drawn between two entities when there is a relation between them. Different approaches to determine and weight entity coherence relations are presented in the following section.

[2]https://developers.google.com/freebase/v1/search

4.1 Entity Coherence

Entity coherence refers to the real world relatedness of different entities which are candidate interpretations of different textual mentions in the document. Such relatedness is not based on documentcontext, so the relatedness of two candidate entities is always the same regardless of the query document. Coherence is represented as an edge between nodes in the graph. We used two measures for coherence:

- Entity Reference Relation (Ref): This is a boolean relation between two entities e_1 and e_2. The Ref relation holds if the Wikipedia document for either entity has a link to the other. Since the Wikipedia hyperlinks are directed, this relation is implicitly directed. However, we assume an inverse relation also exists and represented the relation as undirected.

$$\text{Ref}(e_i, e_j) = \begin{cases} 1, & \text{if } e_i \text{ or } e_j \text{ refers to the other} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

- Entity Co-occurrence ($Jprob$): An estimate of the probability of both entities appearing in the same sentence. Wikipedia documents are used to estimate this probability, as shown in (2), where $S(e)$ is the set of all sentences that contain a hyperlink reference to the entity e and S is the set of sentences containing any such entity references.

$$\text{Jprob}(e_i, e_j) = \frac{|S(e_i) \bigcap S(e_j)|}{|S|} \quad (2)$$

5 Cliques Partitioning Disambiguation

The clique model originated in social network studies when Luce and Perry (1949) defined a clique as a set of two or more people who are mutual friends. In graph theory, this pattern is known as a complete sub-graph. Assuming that NEs that appear in the same document can be split into groups of highly cohesive entities, we adopt the clique partitioning technique to find the biggest clique in terms of size and weight. Given an undirected graph $G(V, D)$ where V is the set of all nodes and D is the set of all edges, $G_s = (V_s, D_s)$ is a sub-graph of G where $V_s \subseteq V$ and $D_s \subseteq D$. G_s is called complete sub-graph or clique if and only if each node in V_s has a link in D_s to all other nodes in V_s. The clique partitioning algorithm aims to find all possible complete sub-graphs G_s in an undirected graph G. Our approach iteratively finds the 'best' clique, deletes all 'wrong' candidate entities for textual mentions that are disambiguated by the selected clique and converts the selected clique to a node in the graph to be used in the next iteration. The details are shown in algorithm 1. Figure 2 shows an exampler of the clique partitioning disambiguation algorithm given a graph of candidate entities for six NE textual mentions, 'A','B','C','D','E','F'. Candidate entities are coded with the lower case letter of the NE textual mention plus an index subscript, e.g., 'a1', 'a2', 'a3', etc. Cliques are shown with bold links in different colours.

As described in section 4, one of the properties of the disambiguation graph is that there are no links between candidates of the same NE textual mention. Because of this property, we can guarantee that there is no more than one candidate for each textual mention in any clique.

Data: Undirected graph $G(V, E)$ and for each node $v \in V$ an associated $IConf$ score
Result: Solution sub-graph
while *not all textual mentions are disambiguated* **do**
  1- clique-List = find cliques(G);
  2- weight each clique by summing the $IConf$ scores of all nodes in the clique;
  3- select the highest scoring clique and use its nodes as disambiguation candidates;
  4- remove all wrong candidates for any mention disambiguated in step 3;
  5- merge all nodes in the selected clique into one node with $IConf$ score of the new node = sum of the $IConf$ scores of the merged nodes;
end

Algorithm 1: Clique Disambiguation Algorithm

This approach does not use an entity coherence weighting (e.g. $Jprob$). Rather it just uses the entity

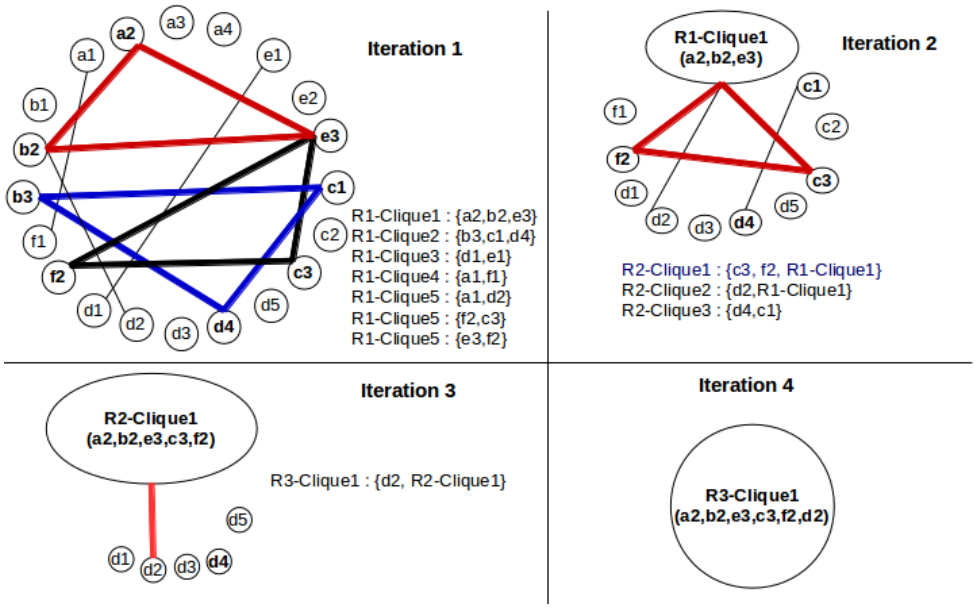


Figure 2: Example of Clique Partitioning Disambiguation

links to find the cliques regardless the relation strength. Because of the huge number of nodes, the clique finder algorithm is not fast. To speed-up the disambiguation, we filtered the nodes with low confidence in the graph, keeping only the top confidence scored 50 NE candidates for each NE textual mention.

6 Graph Ranking Disambiguation

The clique approach disambiguates different NE textual mentions iteratively, where in each iteration one or more NE mentions are disambiguated taking into account the disambiguated mentions from the previous iteration. The graph Ranking approach iteratively ranks all graph nodes depending on the links. So, all NE candidates of all NE textual mentions in the text are ranked together without ignoring any of them. Hence, a selection algorithm is used to combine the initial confidence and the graph rank score, and select the most appropriate NE candidate.

Graph Ranking: The links between different candidates in the graph represent real world relations. These relations are used to reliably boost relevant candidates. In some setups, the weight of these links are set to 1 and in some others they are set to the entities' coherence score. All nodes in the graph are ranked according to these relations using Page-Rank. We adapted a version of the PR algorithm with normalization term to rank the different NE candidates according to entity coherence as shown in equation 3, where N is the number of nodes in the graph, $coh(e_i)$ is the set of nodes that cohere with node e_i and $W(e_i, e_j)$ is the weight of the edge between e_i and e_j nodes. The original PR uses a directed graph while our graph is an undirected graph; so all links are treated as bidirectional.

$$PR(e_i) = \frac{(1-d)}{N} + \frac{d}{F(e_i)} \sum_{e_j \in coh(e_i)} PR(e_j) \times W(e_i, e_j) \tag{3}$$

$$F(e_i) = \sum_{e_j \in coh(e_i)} W(e_i, e_j) \tag{4}$$

The standard PR algorithm assumes the initial rank of all nodes is uniformly equal, while in our approach we used the initial confidence as an initial weight for the candidate nodes. A problem with Page-Rank for our purposes is the dissipation of initial node weight (confidence) over all linked nodes. The final rank of a node is based solely on the importance of linked nodes and the initial confidence plays no further role. In our case this is not appropriate, so the final rank for each mention is calculated after

graph ranking, by combining the graph rank with the initial confidence score. Let us refer to the graph rank of a candidate as $PR(e_i)$. We used two different combination schemes R_s and R_m as described in equations 6 and 5.

$$R_m(e_{i,j}) = ICONF(e_{i,j}) \times PR(e_{i,j}) \quad (5) \qquad R_s(e_{i,j}) = ICONF(e_{i,j}) + PR(e_{i,j}) \quad (6)$$

Decision Making: Selecting the proper candidate is the final phase in the disambiguation process. The simplest approach is to select the highest ranked entity in the list for each mention m_i according to equation 7 or 8, which correpond to the rank combining schemes expressed in equations 5 and 6. Experiments show that overall using the R_m combining scheme is better than the R_s scheme. However, the highest rank, after combining graph rank score and initial confidence score, is not always correct. So we developed a dynamic selection algorithm which uses both combination schemes to pick the best disambiguation candidate. We found that a dynamic choice between the re-ranking schemes, based on the difference between the top two candidates, as described in Algorithm 2, works best. The selected candidate entity is referred to as $\hat{e}$ with the superscript showing the selection scheme.

Data: E_i is a candidate list of one NE textual mention m_i
Result: The best disambiguation NE candidate $\hat{e}_i^g$
$R1 = \{(R_m(e_{i,j}), e_{i,j}) \mid \forall e_{i,j} \in E_i\}$;
$R2 = \{(R_s(e_{i,j}), e_{i,j}) \mid \forall e_{i,j} \in E_i\}$;
Sort $R1$ in descending order ;
Sort $R2$ in descending order ;
$R1diff$ = $R1$[0]-$R1$[1];
$R2diff$ = $R2$[0]-$R2$[1];
if *R1diff* > *R2diff* **then**
  return highest rank scored entity of $R1$, ($R1$[0])
else
  return highest rank scored entity of $R2$, ($R2$[0])
end
Algorithm 2: Selection Algorithm

$$\hat{e}_i^m = \underset{e_{i,j}}{\operatorname{argmax}}\, R_m(e_{i,j}) \quad (7) \qquad \hat{e}_i^s = \underset{e_{i,j}}{\operatorname{argmax}}\, R_s(e_{i,j}) \quad (8)$$

7 Experiments and Results

7.1 Dataset

NIST has released a dataset for use in the TAC KBP entity linking task (EL). But, the task of named entity disambiguation is different from entity linking task, as noted above in Section 2. Also, the NIST dataset is not suitable for evaluating the collective NE disambiguation task because only one NE mention is annotated and disambiguated per query document while we need all mentions of NEs in the document to be annotated and disambiguated to evaluate the performance of the collective named entity disambiguation technique. Another dataset manually annotated for NED is reported in (Kulkarni et al., 2009), but it uses an old version of Wikipedia and it is quite small. We have used another dataset, the AIDA dataset, which is based on the CoNLL 2003 data for NER tagging and in which most tagged NE mentions have been manually disambiguated against Wikipedia (Hoffart et al., 2011). This dataset contains 1393 documents, and 34,965 annotated mentions, where 7136 mention are not linked to Wikipedia[3].

We compare our results to Hoffart's work – Accurate Online Disambiguation of Named Entities (AIDA). For fair comparison, we only considered NE mentions with an entry in the Wikipedia KB, ignoring the 20% of query mentions without a link to the KB, as Hoffart did.

7.2 Evaluation Metric

We use accuracy as the evaluation metric. Micro-averaged accuracy is used as the official metric for the disambiguation task and has been used in much previous and related work. Micro-averaged accuracy

[3]AIDA dataset is available on the web to download http://www.mpi-inf.mpg.de/yago-naga/aida/

corresponds to the percentage of the correctly disambiguated textual mentions and it is calculated as shown in equation 9.

$$A_{micro} = \frac{\text{\#correctly disambiguated mentions}}{\text{Number of NE Mentions}} \tag{9}$$

Macro-averaged accuracy is used to calculate the average percentage of correctly identified named entities. Macro-averaged accuracy is calculated as shown in equation 10.

$$A_{macro} = \frac{\sum_{i}^{num} \frac{\text{Num Correct}(E_i)}{\text{Num Queries}(E_i)}}{\text{\# of unique entities}} \tag{10}$$

7.3 Results

In addition to the state-of-the-art, we used two strong baselines to evaluate the performance of the proposed approaches. The first baseline is a setup where the $IConf$ scores only are used to disambiguate the NE textual mention. In this setup a ranking based on Entity Popularity (EP) does best, with micro- and macro-averaged accuracy scores of 80.55% and 78.09% respectively. This high baseline is close to the state-of-the-art. A summary of the first baseline is shown in Table 1. The second baseline is the basic PR algorithm, where both $IConf$ scores and link weights are ignored. Links between nodes are created wherever any non-zero entity coherence relation, REF or JProb, is found. Micro- and macro-averaged accuracy scores of 70.60% and 60.91% respectively were obtained with this baseline.

	Baseline1		Cliques		PR_I		$\hat{e}^g$	
$IConf$	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}
cos	38.44	45.68	71.59	64.83	70.6	60.83	78.41	72.35
jwSim	61.01	58.81	72.26	69.53	70.61	60.94	83.16	78.28
ctxt	24.58	21.44	58.06	57.37	70.61	60.83	75.45	65.22
EP	80.55	78.09	86.10	81.79	71.78	81.07	87.59	84.19

Table 1: Results using different $IConf$ scores with different approaches

The clique partitioning disambiguation algorithm experiments are setup so a link between nodes is created whenever a non-zero coherence relation is found between nodes regardless its weight. We used different settings for the candidates filter. In the case where no candidates filter is applied, all nodes are considered to find the best initial clique. So, bigger cliques with nodes that have lower confidence may be selected in the first iteration. This approach is very sensitive to the results of the first iteration. Consequently, the accuracy goes down. Also, because of the huge graph size, the clique partitioning algorithm takes a long time. At the other extreme, if we use only a small number of candidates with the highest confidence scores, then the accuracy also goes down because in most cases the correct disambiguation entity is filtered out of the graph. We used the highest 50 candidates in the graph and all other nodes are deleted. Table 1 shows the results of using different initial confidence scores in clique partitioning disambiguation.

Graph ranking disambiguation experiments were setup in three different settings in order to evaluate the contribution of different features like initial confidence and link weights. For all setups, we used different decision making approaches $\hat{e}^m$, $\hat{e}^s$ and $\hat{e}^g$. The results when using $\hat{e}^g$ are better than $\hat{e}^m$ and $\hat{e}^s$ for all setups. So, we report the results of $\hat{e}^g$ only. Different setups are as follows:

- PR_I: In this setup, the $IConf$ scores are used to be the initial rank for Page-Rank while the links between nodes are uniformly weighted to one. As in the PR baseline, links are created wherever Ref or $Jprob$ are not zero. Table 1 shows the results both without $IConf$ combination, i.e. using only the PR score for ranking, and after combining the initial confidence score using dynamic decision making (indicated by $\hat{e}^g$) When comparing these results to the PR baseline, we notice a slight positive effect of using the initial confidence as an initial rank instead of uniform ranking. The major improvement comes by combining the initial confidence with the PR score. All combining

methods improve the results over the baseline results when using the the same confidence score while the dynamic selection algorithm overcomes other basic methods, i.e. $\hat{e}^m$ and $\hat{e}^s$.

- PR_C: In the second setup, entity coherence features are tested by setting the edge weights to the coherence score and the initial node rank is set to be uniform when running the PR algorithm. So, initial confidence scores are not considered in graph ranking but just considered in disambiguation decision making. This setting is intended to evaluate the contribution of different coherence relations. We compared $Jprob$ and Ref edge weighting approaches, where for each approach edges were created only where the coherence score according to the approach was non-zero. We also investigated a variant, called $Jprob + Ref$, in which the Ref edge weights are normalized to sum to 1 over the whole graph and then added to the JProb edge weights (here edges result wherever $Jprob$ and Ref scores are non-zero). Results in Table 2 show the $JProb$ feature seems to be more discriminative than the Ref feature but the combined $Jprob + Ref$ feature performs better than each separately, just outperforming the baseline. We used the best $IConf$ score, i.e. EP, for re-ranking. Again, combining the $IConf$ with the PR score improves the results.

- PR_{IC}: This setup uses different combinations of $IConf$ and entity coherence scores in PR. Table 3 shows the accuracy when using different combinations of all entity coherence scores and some selected (i.e. the best) $IConf$ scores. Here the $Jprob + Ref$ combination does not add any value over $Jprob$ alone. Interestingly using $IConf$ score with differentially weighted edges does not show any benefit over using $IConf$ score and uniformly weighted edges (Table 1).

	PR		$\hat{e}^g$	
Edge Weight	A_{micro}	A_{macro}	A_{micro}	A_{macro}
$Jprob$	66.52	55.83	83.31	80.38
Ref	67.48	59.76	81.80	78.53
$Jprob + ref$	72.69	65.71	83.46	80.69

Table 2: Results using weighted edges (PR_C)

		$\hat{e}^g$	
$IConf$	Edge Weight	A_{micro}	A_{macro}
jwSim	$Jprob$	82.56	76.16
jwSim	Ref	78.61	71.12
jwSim	$Jprob + Ref$	81.97	75.63
EP	$Jprob$	86.29	82.77
EP	Ref	83.16	80.01
EP	$Jprob + Ref$	86.10	82.80

Table 3: Results using initial confidence and weighted edges (PR_{IC})

To compare our results with the state-of-the-art, we report Hoffart et al.'s (2011) results as they re-implemented two other systems and ran them over the AIDA dataset which we used to evaluate our approach. We also compare with Alhelbawy and Gaizauskas (2013) and Shirakawa et al. (2011) who carried out their experiments using the same dataset. Table 4 shows a comparison between the results of our proposed approaches and the state-of-the-art. Both proposed approaches exceed the results of the state-of-the-art. However our approaches are very simple and direct to apply, unlike Hoffart et al.'s and Shirakawa et al.'s which are considerably more complex. Also, our approaches do not need any kind of training, unlike the Alhelbawy approach.

7.4 Discussion

The Page-Rank algorithm was originally designed for directed graphs while our coherence features are undirected. So, the node rank depends on both incoming and outgoing links (when converting the undirected graph to a directed graph). That explains the little improvement over basic PR when using the

	B1	B2	Cliques	PR_C	PR_I	PR_{CI}	Cucerzan	Kulkarni	Hoffart	Shirakawa	Alhelbawy
A_{macro}	78.09	60.91	81.79	80.98	84.19	82.80	43.74	76.74	81.91	83.02	74.18
A_{micro}	80.55	70.60	86.11	83.59	87.59	86.10	51.03	72.87	81.82	82.29	78.49

Table 4: Summary of Presented Approaches and State-of-the-art Results. B1 and B2 are baselines.

initial confidence as an initial rank before using PR (see Table 1). However, when comparing PR results in Tables 2 and 1, we can see that the PR algorithm is more sensitive to the links than to initial ranks. The combined coherence approach ($Jprob + Ref$) actually has a value other than the different weighting it supplies; the approach results in more edges than either of the combined approaches do alone. In all PR results wherever edge weights are applied, the result of using the combined coherence measures outperforms either of them singly.

Informal failure analysis was carried out to determine reasons for disambiguation failure. Reasons identified include:

1. The correct NE candidate does not exist in the graph. In such cases the disambiguation approach selected is irrelevant and what is needed is improved candidate selection.
2. Lack of edges. When there are no edges between any of the query NE mention candidate entities and other mentions' candidates. In this case the decision depends only on the $IConf$ score.
3. Where the Freebase popularity score (EP) is used, whenever this score for the correct NE candidate is 0, which means the selection process is based on the PR score.

Table 5 shows an example of the highest three NE candidates for three NE mentions taken from a document (overall the document contains textual mentions for ten different NEs). The first one is "Ford" and is disambiguated correctly to "Ford Motor Company", where the PR and popularity scores are higher than any of the other candidates. The second one is ,"Magna", disambiguated correctly, where the first two NE candidates have the same PR score but the popularity score discriminates between them. The third, "Markham", is disambiguated to "Clements Markham" while it should be disambiguated to "Markham, Ontario". The problem in this case is that all NE candidates for the mention "Markham" are not linked to any entity candidates for any other NE mentions in the document (problem 2 above). Therefore, the popularity score dominates the final rank score.

NE Candidate	PR score $\times 10^{-3}$	FB Rank $\times 10^{-3}$	our Rank $\times 10^{-3}$
Ford			
Ford Motor Company	21.37	62.12	1.32
Ford Galaxie	4.59	10.94	0.05
Ford GT	2.83	11.43	0.03
Magna			
Magna International	2.65	4.78	0.013
Magna Powertrain	2.65	2.18	0.005
Germania	0.83	3.46	0.003
Markham			
Clements Markham	0.83	4.42	0.004
Markham Waxers	0.83	3.67	0.003
Edwin Markham	0.83	2.89	0.002

Table 5: Example show the first three NE candidates for three NE mentions with scores

8 Conclusion

Our results show that graph ranking and cliques partitioning approaches in conjunction with the candidate confidence scores and entity coherence across a disambiguation graph can be used as an effective approach to collectively disambiguate named entity textual mentions in a document. Our proposed features are very simple and easy to extract, and work well when employed in PR or clique partitioning algorithms. Also, entity coherence is a discriminative feature when using graph models for NED. In future work we plan to explore enriching the edges between nodes, by incorporating semantic relations extracted from an ontology, and extending the scope of entity co-occurrence to be the document instead of the sentence. Also, it is worth investigating whether using the entity coherence score can help when evaluating clique weight in the clique partitioning algorithm.

References

Eneko Agirre and Aitor Soroa. 2008. Using the multilingual central repository for graph-based word sense disambiguation. In *LREC*.

Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41. Association for Computational Linguistics.

Ayman Alhelbawy and Rob Gaizauskas. 2012. Named entity based document similarity with svm-based re-ranking for entity linking. In *Advanced Machine Learning Technologies and Applications*, volume 322 of *Communications in Computer and Information Science*, pages 379–388. Springer Berlin Heidelberg.

Ayman Alhelbawy and Robert Gaizauskas. 2013. Named entity disambiguation using hmms. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 159–162.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Coen Bron and Joep Kerbosch. 1973. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577.

Razvan C. Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL*. The Association for Computer Linguistics.

S. Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP-CoNLL*, volume 6, pages 708–716.

M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 277–285. Association for Computational Linguistics.

S. Gottipati and J. Jiang. 2011. Linking entities to a knowledge base with query expansion. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 804–813. Association for Computational Linguistics.

Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. 2011. Word sense disambiguation: a graph-based approach using n-cliques partitioning technique. In *Natural Language Processing and Information Systems*, pages 112–124. Springer.

Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. 2012. A graph-based approach to wsd using relevant semantic trees and n-cliques model. In *Computational Linguistics and Intelligent Text Processing*, pages 225–237. Springer.

X. Han, L. Sun, and J. Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

J. Hoffart, M.A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

H. Ji and R. Grishman. 2011. Knowledge base population: successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1148–1158. Association for Computational Linguistics.

S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM.

R.Duncan Luce and AlbertD. Perry. 1949. A method of matrix analysis of group structure. *Psychometrika*, 14(2):95–116.

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

D. Milne and I.H. Witten. 2008. Learning to link with wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM.

D. Nadeau and S. Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Kamel Nebhi. 2013. Named entity disambiguation using freebase and syntactic parsing. In CEUR-WS.org, editor, *Proceedings of the First International Workshop on Linked Data for Information Extraction (LD4IE 2013) co-located with the 12th International Semantic Web Conference (ISWC 2013)*. Gentile, A.L. ; Zhang, Z. ; d'Amato, C. & Paulheim, H.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, November.

L. Ratinov, D. Roth, D. Downey, and M. Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*.

P. Sen. 2012. Collective context-aware topic models for entity disambiguation. In *Proceedings of the 21st international conference on World Wide Web*, pages 729–738. ACM.

Masumi Shirakawa, Haixun Wang, Yangqiu Song, Zhongyuan Wang, Kotaro Nakayama, Takahiro Hara, and Shojiro Nishio. 2011. Entity disambiguation based on a. Technical report, Technical report, Technical Report MSR-TR-2011-125, Microsoft Research.

Ravi Sinha and Rada Mihalcea. 2007. Unsupervised graph-basedword sense disambiguation using measures of word semantic similarity. In *Semantic Computing, 2007. ICSC 2007. International Conference on*, pages 363–369. IEEE.

Wenpu Xing and Ali Ghorbani. 2004. Weighted pagerank algorithm. In *Communication Networks and Services Research, 2004. Proceedings. Second Annual Conference on*, pages 305–314. IEEE.

Wei Zhang, Yan Chuan Sim, Jian Su, and Chew Lim Tan. 2011. Entity linking with effective acronym expansion, instance selection and topic modeling. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 1909–1914. AAAI Press.

Z. Zheng, F. Li, M. Huang, and X. Zhu. 2010. Learning to link entities with knowledge base. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 483–491. Association for Computational Linguistics.

Analysis and Refinement of Temporal Relation Aggregation

Taylor Cassidy
IBM Research
Army Research Laboratory
Adelphi, MD 20783, USA
taylor.cassidy.ctr@mail.mil

Heng Ji
Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY 12180, USA
jih@rpi.edu

Abstract

To obtain a complete temporal picture of a relation it is necessary to aggregate fragments of temporal information across relation instances in text. This process is non-trivial even for humans because temporal information can be imprecise and inconsistent, and systems face the additional challenge that each of their classifications is potentially false. Even a small amount of incorrect proposed temporal information about a relation can severely affect the resulting aggregate temporal knowledge. We motivate and evaluate three methods to modify temporal relation information prior to aggregation to address this challenge.

1 Introduction

Temporal information about relations is conveyed in text at varying levels of completeness and specificity. A sentence may indicate that a relation starts, ends, or that it is ongoing at a particular time. Furthermore, a time expression may be expressed at a variety of granularity levels (e.g., hour, day, or year). For instance, *"Collins, ..., is a 61-year-old veteran who went 444-434 in six seasons as a manager, 1994-1996 with Houston"* provides bounds on both the start and end date of the a relation but at a coarse granularity. Conversely, *"Ivory Coast President Laurent Gbagbo on state television Friday dissolved parliament"* conveys temporal information about an arbitrary part of Gbagbo's presidency at a finer granularity: the relation simply holds true at the document creation time (DCT). Single instances in which a relation of interest is related to a time expression often fail to convey complete, fine-grained temporal information. Thus, it is necessary to *aggregate* information from multiple relation-time *temporal relationship* mentions to gain a complete temporal picture of a relation.

We focus on the aggregation of temporal information about relations within the context of the Temporal Slot-Filling (TSF) Task (Ji et al., 2011; Surdeanu, 2013). TSF focusses on a class of relations called *fluents* (Russell and Norvig, 2010), which are properties of named entities whose values may vary over time. Systems must succinctly describe all temporal information about each query relation R – e.g., `title`(Gbagbo, President) – available in a source document collection by assigning it a single, final temporal *four-tuple* (Amigo et al., 2011). Given a relation mention r of R and a time expression γ, a four-tuple $T_\gamma^r = \langle t^{(1)}, t^{(2)}, t^{(3)}, t^{(4)} \rangle$ characterizes their temporal relationship; namely, $t^{(1)}$ and $t^{(2)}$ represent the earliest and latest possible start date for R, while $t^{(3)}$ and $t^{(4)}$ represent the earliest and latest possible end dates, as inferred from the relation mention's context (sec. 3). For instance, a sentence indicating that Gbagbo was President on 2010-02-12 yields $\langle -\infty, \text{2010-02-12}, \text{2010-02-12}, +\infty \rangle$, while the sentence *"Gbagbo has been in power since 2000"* yields $\langle \text{2000-01-01}, \text{2000-12-31}, \text{2000-01-31}, +\infty \rangle$. The intuitively best aggregation of these four-tuples expresses what we learn from both texts, that the relation started in 2000 and remained ongoing at 2010-02-12, i.e. $\langle \text{2000-01-01}, \text{2010-02-12}, \text{2010-02-12}, +\infty \rangle$, with no clear indication as to its end. Straightforward cases like these were used to justify the simple aggregation methods used by all TSF systems to date (Surdeanu, 2013; Ji et al., 2011). However, in reality even humans often must deal with vague and/or conflicting temporal information across documents,

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1556–1566, Dublin, Ireland, August 23-29 2014.

and systems must furthermore deal with the fact that each of their temporal relationship classifications is potentially false.

To address the various properties of text and temporal representation that influence aggregation and affect final four-tuple quality, we first improve an existing gold standard dataset (sec. 4.1). We then describe two key factors affecting systems' aggregation performance: (1) erroneous classifications attributed high confidence by systems, and (2) a lack of relation-bounding classifications (sec. 4.2). We propose three methods to better prepare a relation's multiple mention context derived four-tuples for aggregation into a final four-tuple. The first applies simple rules to predicative nominal titles with explicit time information (e.g., *"former President"*), the second filters and re-labels four-tuples based on entity lifespan (sec. 5.3), and the third adds four-tuples based on mentions of relations other than, but temporally linked to, the query relation (sec. 5.4). We then discuss results and identify remaining challenges for aggregating temporal information across relation mentions (sec. 6 and 7). A Glossary of selected terms can be found in the appendix.

2 Related Work

The most similar work on temporal relation information aggregation are Wang et al. (2012), who use an Integer Linear Programming framework to enforce the validity of induced temporal relation information as well as enforce inter-relation constraints, and Dylla et al. (2013), who collect temporal information about relations, mostly about start and end times, using a temporal probabilistic data base framework to aggregate and enforce constraints based on relation argument existence. All TSF systems we are aware of have used either max-constrain or Validity-Ensured Incremental max-constrain aggregation algorithms (Surdeanu, 2013; Ji et al., 2011), which we cover in section 4. None we are aware of have applied background knowledge to constrain intermediate four-tuples (sec. 3) before or after aggregation. In this work we modified our previous work CUNYTSF (Artiles et al., 2011), which is the only publicly available TSF system we are aware of. CUNYTSF employs two supervised models, one based on a string kernel defined in terms of dependency paths between named entities involved in a relation and context time expressions, and the other based on bags-of-words derived from small windows surrounding these tokens and shallow dependency relations. CUNYTSF achieved the highest and second-highest scores of five systems in two TSF shared tasks (Surdeanu, 2013; Ji et al., 2011).

3 Temporal Slot Filling (TSF)

The 2013 Temporal Slot-Filling (TSF) (Surdeanu, 2013) task was part of the Knowledge Base Population (KBP) track of the Text Analysis Conference (TAC). Systems were given a list of 273 fluent relation instances as queries, each with a supporting document. Query relations were evenly distributed across relation types, which consisted of people's *titles, marriages, employments or memberships, and residences (city, state, and country)*, and companies' *top members or employees*. The task was to obtain a final four-tuple T_R for each query relation $R = \langle q,\ s \rangle$ using the source corpus for provenance. For each element in T_R a system must provide a document in which R is entailed, and offsets for the relation arguments (the query-entity q and slot-filler s) and the normalized time expression from which the four-tuple element is derived.

The KBP source collection consists of about 1 million newswire, 1 million web text, and 100,000 discussion forum documents. Gold standard annotation was obtained by annotators who, using a tool, searched the source corpus for documents that provide temporal information about each query relation. Given a mention r of R in a document d for which temporal information about R could be inferred, annotators assigned an intermediate temporal relationship label (Table 1) (Ji et al., 2011) to $\langle r, \gamma \rangle$, where γ is viewed as an interval of dates $[\gamma_s, \gamma_e]$ derived either based on (1) a normalized time expression in d, or (2) the document creation time of d. We denote the temporal extension of R at the day granularity $R_{ex} = [R_s, R_e]$, where R_s and R_e are the start and end dates of R. The intermediate label l mediates the relationship between γ and R_{ex}, characterizing a possible relationship between R and γ. [1] After systems submitted results for the shared task, any corresponding document not included in the original annotation

[1] We add AFTER_END* and BEFORE_START* but omit motivation due to space constraints.

that were determined to express R was exhaustively annotated for temporal information about R. A gold standard final four-tuple G_R is obtained for each R by applying an aggregation procedure (sec. 4.1) to the intermediate temporal relationship labels assigned to mention-time classification instances (Surdeanu, 2013).

In this work we adopt the evaluation metric used for the TSF shared task (Ji et al., 2011; Surdeanu, 2013).

Intermediate Relation	**four-tuple**
BEGINNING	$\langle \gamma_s, \gamma_e, \gamma_s, \infty \rangle$
ENDING	$\langle -\infty, \gamma_e, \gamma_s, \gamma_e \rangle$
BEG_AND_END	$\langle \gamma_s, \gamma_e, \gamma_s, \gamma_e \rangle$
WITHIN	$\langle -\infty, \gamma_e, \gamma_s, \infty \rangle$
THROUGHOUT	$\langle -\infty, \gamma_s, \gamma_e, \infty \rangle$
BEFORE_START	$\langle \gamma_e, \infty, \gamma_e, \infty \rangle$
AFTER_END	$\langle -\infty, \gamma_s, -\infty, \gamma_s \rangle$
BEFORE_START*	$\langle \gamma_s, \infty, \gamma_s, \infty \rangle$
AFTER_END*	$\langle -\infty, \gamma_e, -\infty, \gamma_e \rangle$
NONE	$\langle -\infty, \infty, -\infty, \infty \rangle$

Table 1: Intermediate temporal relationship function for $\langle r, \gamma \rangle$

Invalidity Source	Frequency
Conflicting Information	13
Multiple Instances	7
Wrong Intermediate Label	20
Vague Time Normalization	8
Other	8

Table 2: Reasons for Invalidity in Gold Standard Final Four-Tuples

4 Aggregating Intermediate Relations

Temporal information about instances of R must be aggregated to yield a complete temporal picture of the relation with respect to the background corpus. We denote with $I(R)$ the set of intermediate four-tuples associated with R. The purpose of the four-tuple representation is to be as accurate as possible in representing the extent to which a given corpus provides information about the start and end time of R, R_s and R_e, while preserving the vagueness inherent in the text. Each four-tuple element of $I(R)$ represents temporal information about R_s and/or R_e, most often with respect to the context associated with a particular mention r of R. Temporal information at a corpus level is derived via a process of aggregation over the elements of $I(R)$. In this section we describe how both human annotators and systems have approached aggregation.

4.1 Aggregating Manually Annotated Intermediate Relations

Gold standard four-tuples were obtained by applying the Max-Constrain (MC) algorithm (Equation 1) to each $I(R)$ obtained via manual annotation using the labels in Table 1 (Surdeanu, 2013; Ji et al., 2011).[2]

$$T_R = \langle max(t^{(1)}), min(t^{(2)}), max(t^{(3)}), min(t^{(4)}) \rangle \quad (1)$$

Here, $max(t^{(k)})$ is the greatest $t^{(k)}$ from any intermediate four-tuple $T_r \in I(R)$, while $min(t^{(k)})$ is the least.

Let a four-tuple T be *valid* iff. $t^{(1)} \leq t^{(2)} \wedge t^{(3)} \leq t^{(4)} \wedge t^{(1)} \leq t^{(4)}$, and *correct* if $t^{(1)} \leq R_s \leq t^{(2)} \wedge t^{(3)} \leq R_e \leq t^{(4)}$. If R has only one start and one end date, and $R_s \leq R_e$, and each intermediate four-tuple $T_r^{\gamma} \in I(R)$ is valid and correct, then the final four-tuple obtained via MC is guaranteed to be valid and correct. Fifty-six gold standard final four-tuples were invalid and therefore discarded prior to evaluation (Surdeanu, 2013). We analyzed them by hand to determine the source of their invalidity (see Table 2). [3] We corrected instances until IMC (Algorithm 1) yielded a valid four-tuple.

[2] See http://surdeanu.info/kbp2013 for more details.

[3] Note that there may be more instances of each type described in table 2

[4] Here, $\max(t^{(i)} \leq x^{(i)}) := \max(\left\{ t^{(i)} \in \mathbf{t}^{(i)} \middle| t^{(i)} \leq x^{(i)} \right\})$, where $\mathbf{t}^{(i)} := \left\{ t^{(i)} \in T \middle| T \in I(R) \right\}$

Algorithm 1 Inclusive Max-Constrain (IMC)[4]

Require: $I(R) = \{T_0, T_1, \ldots, T_{N-1}\}$
Ensure: T_R
  $X \leftarrow$ max-constrain$(I(R)) = \langle x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)} \rangle$
  $Y \leftarrow \langle \max(t^{(1)} \leq x^{(2)}), \min(t^{(2)} \geq x^{(1)}), \max(t^{(3)} \leq x^{(4)}), \min(t^{(4)} \geq x^{(3)}) \rangle$
  $T_R \leftarrow \langle \max(t^{(1)} \leq y^{(2)}), \min(t^{(2)} \geq y^{(1)}), \max(t^{(3)} \leq y^{(4)}) \rangle, \min(t^{(4)} \geq y^{(3)})$
  return T_R

4.2 System Derived Intermediate Relations

As suggested in section 4.1, MC is sensitive to inconsistent four-tuples. In response to this all prior work that has not used MC to combine system-produced $I(R)$ has used an algorithm similar to Validity-Ensured Incremental (VEI) Max-Constrain (Algorithm 2) (Artiles et al., 2011). Here, $I(R)$ is ordered by classifier confidence and T_R is initialized as the trivial four-tuple and updated incrementally. Starting with the highest-confidence four-tuple $T_{R,0} \in I(R)$, MC is applied to $\{T_R, T_{R,i}\}$ to yield T^*. In a given iteration, T^* is only accepted as the updated T_R if it is valid. Intuitively, higher confidence intermediate four-tuples are more likely to be correct, thus the incremental algorithm tries to ensure that erroneous low-confidence four-tuples are less likely to be aggregated. In practice, however, a single high-confidence incorrect label can derail the entire process (sec. 5).

Algorithm 2 Validity-Ensured Incremental (VEI) Max-Constrain Aggregation to yield final four-tuple

Require: $I(R) = \{T_0, T_1, \ldots, T_{N-1}\}$
Ensure: $T_R = \langle t^{(1)}, t^{(2)}, t^{(3)}, t^{(4)} \rangle$
  $T_R \leftarrow \langle -\infty, \infty, -\infty, \infty \rangle$
  $i \leftarrow 0$
  while $i < N$ **do**
    $T^* \leftarrow \langle max(t^{(1)}, t_i^{(1)}), min(t^{(2)}, t_i^{(2)}), max(t^{(3)}, t_i^{(3)}), max(t^{(4)}, t_i^{(4)}) \rangle$ {Pairwise MC}
    if $t^{*(1)} \leq t^{*(2)} \wedge t^{*(3)} \leq t^{*(4)} \wedge t^{*(1)} \leq t^{*(4)}$ **then**
      $T_R \leftarrow T^*$ {Validity Check}
    end if
  end while
  return T_R

5 Challenges and Solutions

This section outlines our modifications to CUNYTSF, inspired by a preliminary error analysis. We implement three methods geared toward better preparing $I(R)$ for aggregation into a final four-tuple..

5.1 Preliminary Error Analysis

We ran the publicly available system CUNYTSF described in (Artiles et al., 2011) on the queries used in TSF2013, using the KBP2013 source collection, and evaluated against the corrected gold standard described in section 4.1. [5]

Error analysis revealed the main source of errors to be WITHIN labels with high confidence. To be exact, the final four-tuple for 116 queries (of 271) was influenced by a WITHIN label that yielded a $t^{(3)}$ later than the $g^{(4)}$ date, while 20 were influenced by WITHIN dates that were too early. Under VEI, once a labeled instance $\langle r, \gamma, \text{WITHIN} \rangle$ is aggregated into T_R, if $\gamma > R_e$ then any correctly labeled instance $\langle r, \gamma, \text{ENDING} \rangle$ will yield an invalid four-tuple and thus be rejected. (Similarly, correct BEGINNING labels will be blocked by incorrect WITHIN labels that are too early). Even correct WITHIN labels cannot set the corrupted aggregation back on track, since pairwise MC will always take the later $t^{(3)}$

[5] System downloaded from http://nlp.cs.rpi.edu/software.html

(algorithm 2). That said, WITHIN labels are often required to retrieve a complete temporal picture of a relation conveyed in a corpus. WITHIN is the most common intermediate label in the source collection, constituting 44% of correct labels, and furthermore, over half of the query relations require at least one WITHIN label to achieve the gold standard final four-tuple, with 10% relying solely on instances labeled WITHIN. To make matters worse, almost all TSF systems to date (except Garrido et al. (2013)) use neither the BEFORE_START* nor AFTER_END* labels in their intermediate temporal relationships classification models, even though high-confidence instances with those labels could prevent the sort of erroneous WITHIN labels alluded to above.

This analysis motivated three methods to curtail the extent to which aggregation-derailing four-tuples were included in $I(R)$ described in sections 5.2, 5.3, and 5.4. We favor VEI over IMC for system-derived $I(R)$ because IMC strongly relies on the assumption that there is a high probability of correctness for each intermediate relationship annotation.

5.2 Title Time of Predication

Nominal predicates are commonly used in English to refer to fluents. For example, attribution of a title to a person can be performed using a transitive verb or copula as in *"Serra was elected Governor"*, or *"Serra is the Governor"*, or as a Noun Phrase (NP) within a clause, as in *"Governor Jose Serra"* or *"Jose Serra, Governor, ..."* (among other ways). We refer to cases in which the subject and object of the relation are contained within a phrase headed by a Noun as Relational NP's (RNP).[6] For RNP that are mentions of fluent relations, there is a time of predication (TOP), i.e. a time at which the relation conveyed is asserted to hold, though this time is not overtly marked by tense or aspect (in English) as in the case of VP's. Tonhauser (2002)'s analysis assumes that the verbal time of predication (VTOP) is the "most salient" time in an utterance, thus relational NP's take their containing clause's verbal time of predication by default though contextual justification may override this tendency. We propose that in news the DCT is just as salient a time since the focus is centered on current affairs, an important entities are often "already introduced" into the discourse by virtue of being public figures. Ad-hoc analysis of the instances considered by CUNYTSF indicate that a compelling reason is required to override RNP's from taking both DCT and VTOP. For instance, in, *"O'Donnell ... suggested Wednesday that the Obama administration - particularly **Vice President Joe Biden**, who **represented** Delaware in the Senate for decades - was behind them"*, *"Vice President"* holds true at DCT, and rejects the VTOP of *"represented"*, presumably only based on logical inference: *no person is both Vice President and represents (a state) in the Senate at the same time*. Similarly, we know that the DCT (2010-08-04) is an invalid TOP in *"In November 2000, Chinese **President Jiang Zemin** paid a state visit to Laos, the first visit to Laos by a Chinese president"*, only because of world knowledge, or, *"The following is a chronology of major events in China- Laotian relations since 1990:"*, earlier in the document.

Though NP's lack tense and aspect, overt temporal modifiers such as *former, then-,* and *ex-* make explicit a *post-relational state* directly following an RNP's relation (Tonhauser, 2002).[7] The tendency for RNP's to take both the verbal predication time as well as the DCT extends to post-relational states. There are many examples in the corpus similar to the following: *"Former US President Bill Clinton and US journalists Euna Lee and Laura Ling returned Wednesday from North Korea, one day after North Korea's leader Kim Jong-Il pardoned the two women"*. Each RNP holds at the DCT, and *"Wednesday"*, as well as the day before that (the VTOP of *"pardoned"*). However, as for VTOP's further into the past, whether the post-relational state holds is less clear. For example, in, *"Secretary of State Hillary Rodham Clinton says former Philippines President Corazon Aquino "helped bring democracy back" to her country after years of authoritarian rule"*, we cannot rule out the possibility that Aquino helped bring democracy back *as President*; whether she did so *as former President* is left open, to be resolved by historical knowledge. This is likely because, unless the relation is of the "Grover Cleveland" type, once the relation becomes a "former" relation it will remain so thereafter.

[6] We adopt a Noun Phrase rather than a Determiner Phrase framework for simplicity.

[7] In this work we omit similar constructions that indicate a pre-relational state at the time of verbal predication, such as "future-", "soon-to-be", and "-elect". These words to not occur often in our data. That said, the extent to which their meanings are analogous to the overt temporal modifiers that introduce post-relational states is not clear, and requires further investigation.

The nature of the contexts that override default TOP for RNP's is complicated, and not well understood. In addition, determining VTOP automatically remains a difficult problem in and of itself (Uz-Zaman et al., 2012). We have shown that newswire data contains relational NP's whose default times of predication - both DCT and verbal - are overridden by context. In addition, even post-relation states of modified RNP's may reject VTOP's. Post-relational states introduced by RNP's modified with *"former"*, *"then"*, and *"ex-"*, however, do appear to unambiguously take the DCT as a time of predication. Furthermore, we observe that CUNYTSF often incorrectly classifies modified RNP's introducing a post-relational state as expressing $\langle r, DCT, \text{WITHIN} \rangle$. To correct these errors we apply hand-written **Title Time of Predication Fix** rules to change the label for all such classification instances to AFTER_END* when the associated time expression is (or is closely related to) the DCT, and attribute 100% confidence to this new label. This correction both removes erroneous WITHIN labels and introduces labeled instances that bound query relations.

5.3 Entity Existence

VEI suffers when confidence values are inaccurate. For the relation `spouse`(Marylin Monroe, Arthur Miller), given the sentence, *"Editor Courtney Hodell said the book would include poems , photographs , reflections on third husband Arthur Miller and other men in Monroe 's life "*, a system is likely to mislabel $\langle r, \gamma \rangle$ as WITHIN, where γ is the document creation time 2010-04-27. The pattern *"husband s"* is a strong indicator of the WITHIN relationship for the `spouse` relation, so confidence for the resulting four-tuple $\langle -\infty, \text{2010-04-27}, \text{2010-04-27}, \infty \rangle$ is likely to be high. Once aggregated, it would be impossible to later aggregate $\langle -\infty, \text{1961-12-31}, \text{1961-01-01}, \text{1961-12-31} \rangle$ upon learning of the couple's divorce in 1961, since the proposed $T^* = \langle -\infty, \text{1961-12-31}, \text{2010-04-27}, \text{1961-12-31} \rangle$ is invalid. A basic clue that a WITHIN label should be changed to AFTER_END* is that q or s no longer exists (either the person has died or the business has dissolved).

To address this challenge we propose **Existence-based Correction and Filtering**. For each relation R we obtain the *existence four-tuple* E_R, by applying MC aggregation to the set of birth and death times in a knowledge base (KB) for the query-entity and slot-filler.[8] The KB is obtained via the Freebase API and scraping Wikipedia Infoboxes. We use a four-tuple instead of an interval of dates because birth and/or death information may not be available at the date granularity. Given the relation `spouse`(Jennifer Jones, Norton Simon) and the KB excerpt in Table 3, we obtain an existence constraint four-tuple $\langle \text{1919-03-02}, \text{1919-03-02}, \text{1993-06-02}, \text{1993-06-02} \rangle$.

Entity	Birth	Death
Jennifer Jones	1919-03-02	2009-12-17
Norton Simon	1907-02-05	1993-06-02

Table 3: Existence Information

We apply algorithm 3, where C contains classifier confidence for each labeled instance in $I(R)$. Above, $I(R)$ was introduced as a list of intermediate four-tuples for a relation R. In our approach, each of these four-tuples is derived deterministically (see Table 1). From here on (as in Algorithm 3) we allow a slightly abuse of notation in which $I(R)$ is viewed as a set of labeled classification instances, each of which yields a four-tuple for R. We omit pseudo-code to handle the analogous cases where instances are re-labeled BEFORE_START* based on the relative position of γ and ϵ_1.

5.4 Relation Precedence

The context of a relation mention often contains temporal information not explicitly tied to a time expression. For example, in, *"Myasnikovich will replace Sergei Sidorsky, who was prime minister since 2003"*, there is no date explicitly tied to the transition of power. Many titles are held by one person after another, in succession, without overlap. Intuitively, if we know the order in which several individuals held the same title then temporal information about one such relation can be used to constrain the other.

[8] For organization query-entities their foundation and defunct dates are considered their "birth" and "death" dates.

Algorithm 3 Existence Based Correction & Filtering Algorithm

Require: $I(R) = \{\langle\gamma_0, l_0\rangle, \ldots, \langle\gamma_k, l_k\rangle\}$; $C = \{c_0, \ldots, c_k\}$; $E_R = \langle\epsilon^{(1)}, \epsilon^{(2)}, \epsilon^{(3)}, \epsilon^{(4)}\rangle$
  while $i < N$ **do**
    if $\gamma_i.s \geq \epsilon_4 \wedge \neg(l_i = \text{NONE})$ **then**
      if $l_i = \text{ENDING} \wedge \gamma.s - \epsilon_4 \leq 31$ **then**
        $c_i \leftarrow 1.0$ {Most likely R holds at the time of death}
      else
        $l_i \leftarrow$ AFTER_END*; $c_i \leftarrow 1.0$
      end if
    else if $\gamma_i.s \leq \epsilon_4 \leq \gamma_i.e \wedge \neg(l_i = \text{NONE})$ **then**
      if $l_i = \text{ENDING}$ **then**
        $c_i \leftarrow 1.0$ {Most likely R holds at the time of death}
      else
        $l_i \leftarrow$ AFTER_END*; $c_i \leftarrow 1.0$
      end if
    end if
  end while
  return $I(R)$

To address this challenge we propose **Precedence-based Query Expansion and Re-labeling**. The title relation is well-represented in Wikipedia, and the infobox for many political title holders contains fields for *preceded by* and *succeeded by*, which specify the person that held the same title before and after the title holder in question. Given a title query R, we extracted the person who preceded and succeeded the query entity from the query entity's infobox (when available). Additional title relation *supporter* queries – R_{pre} and R_{suc}, respectively – were generated using these names, and the same title name as in the official query.[9]

After all classification instances are labeled and existence based correction is applied, we transform all labeled instances for supporter queries into labeled instances for official queries. Given a labeled instance $\langle r_x, \gamma, l\rangle$, where $x =$ pre or suc, we apply the mapping in Table 4 to yield the transformed labeled classification instance $\langle r, \gamma, l'\rangle$. Labeled supporter instances transformed into labeled official query instances are added to $I(R)$, the set of labeled instances for R. The set $I(R)$ is then passed to Aggregation (see Algorithm 2).

Supporter Label l	**Official label** l' ($x =$ pre)	**Official label** l' (when $x =$ suc)
NONE	NONE	NONE
BEFORE_START*	BEFORE_START*	NONE
AFTER_END*	NONE	AFTER_END*
All Others	BEFORE_START*	AFTER_END*

Table 4: Mapping to convert $\langle r_x, \gamma, l\rangle$ to $\langle r, \gamma, l'\rangle$, where x indicates whether the supporter query precedes or succeeds the official query

Just about any instance $\langle r_{\text{pre}}, \gamma, l\rangle$ yields $\langle r, \gamma, \text{BEFORE\_START*}\rangle$ because R_{pre} is known to both start and end before R starts. (And conversely $\langle r_{\text{suc}}, \gamma\rangle$ tends to yield AFTER_END* for $\langle r, \gamma\rangle$.) This is because the last (first) day of R_{pre} and all days before (after) it are guaranteed to be before (after) the start (end) of R. However, note that a AFTER_END* label for $\langle r_{\text{pre}}, \gamma\rangle$ yields NONE for R since dates after the end of R_{pre} may be before, during, or after R. For example, the headline, *"Former President*

[9]In general, knowing that two relations stand in a particular interval relation to one another allows us to posit constraints on one relation upon discovering temporal information about the other. We apply this intuition to the title relation in this work since the information is readily available in a structured form (i.e., the *preceded by* and *succeeded by* fields in Wikipedia info boxes).

Lee Teng-hui on visit in Japan Tokyo", while clearly indicating AFTER_END* for R_{pre} tells us very little about the relationship between the document creation time and R.

6 Results and Analysis

We scored the output for five conditions using the modified gold standard (section 4.1). TF means that title time of predication fix was applied (section 5.2), EC means existence corrections were applied, and Pr means that precedence-based query expansion was applied (section 5.4).

System	P	R	F
CUNYTSF	.337	.294	.314
CUNYTSF + TF	.341	.298	.318
CUNYTSF + EC	.349	.305	.326
CUNYTSF + TF + EC	.353	.309	.329
CUNYTSF + TF + EC + Pr	**.360**	**.315**	**.336**

Table 5: Results calculated using official TSF2013 scorer against corrected gold standard (sec. 4.1), with `anydoc` and `ignore-offsets` parameters set to true, augmented to calculate recall and precision

6.1 Title Time of Predication Fix

The gold standard for `title` had 142 non-infinity tuple element outputs of the form $\langle R, i, t^{(i)} \rangle$. The baseline output had 80 values while baseline + TF had 91. Applying TF, 10 baseline outputs were replaced while 11 were added. In most cases erroneous WITHIN labels are corrected by inserting high-confidence AFTER_END* into $I(R)$. In some cases this allows a correct $t^{(3)}$ to replace a later, incorrect $t^{(3)}$ that came from an erroneous WITHIN label. It is important to note that while some changes barely affect F-measure, they are important because they allow for correct information that would have otherwise been blocked to be aggregated. For example, a bad baseline WITHIN for *"General Prosecutor 's Office of Kyrgyzstan on Tuesday charged the country's former Prime Minister Igor Chudinov with abuse of power"* had blocked a correct WITHIN for *"Kyrgyz Prime Minister Igor Chudinov left Beijing Thursday evening"* - removing this block allowed $t^{(3)}$ to change from 2010-05-04 to 2009-10-14, which is the gold standard value.

6.2 Existence-based Correction and Filtering

Most changes made from existence constraints are beneficial both in terms of an increase in F-measure and in blocking the aggregation of incorrect information. For instance, it is difficult to prevent labeling the following sentence with WITHIN for DCT: *"The **London** home of composer George Frideric Handel is holding an exhibition about its other famous resident – **Jimi Hendrix**"*, but the document context permits AFTER_END*, given *"**Hendrix** died in **London** on Sept. 18 , 1970"*. Given the existence constraint we label the instance AFTER_END*.

On the other hand, in some cases we erroneously change WITHIN to BEFORE_START* using existence constraints, but this type of change does little damage. For example, the fact that CNN was founded on 1980-06-01 changes the label on 1980 from WITHIN to BEFORE_START* for EMPLOYEE(Novak, CNN), given *"Novak , editor of the Evans-Novak Political Report , is perhaps best known as a co-host of several of CNN 's political talk shows , where he often jousted with liberal guests from 1980 to 2005"*. We set $t^{(1)} =$ 1980-01-01 which does not block later inclusion of a correct $\langle R, 1980, \text{BEGINNING} \rangle$, which would set $t^{(1)} =$ 1980-01-01 if it were not already set, and does set $t^{(2)} =$ 1980-12-31. Changing this relation's label from WITHIN to START is not a catastrophic error because it allows for a finer grained, correct start date to be aggregated using VEI (see Algorithm 2) to yield a superior final four-tuple (though CUNYTSF finds no suitable candidates to facilitate this).

6.3 Precedence-based Query Expansion & Re-labeling

Output for affected official queries were improved simply because supporter queries were accurately labeled. For example, *"Kim Choongsoo, Korea's Central Bank Governor, said here on Thursday his nation's economic situation was getting better"* provides a $t^{(4)}$ value for `title`(Lee Seong-tae, Governor) due given the successor relation.

Some gains from label transformation are only possible given the title time of predication fix. For example, multiple instances of *"former president Chen Shui-bian"* and *"Former President Lee Teng-hui"* were converted from WITHIN to AFTER_END* for their respective relations. Because Chen succeeded Lee, the latter instances were transformed to NONE instances for `title`(Chen, President) using Table 4. [10] Changing these labels to NONE made room for a valid $t^{(3)} =$ 2000-01-01 based converting the WITHIN for `title`(Lee, President) to BEFORE_START* for `title`(Chen, President) given, *"... since former President Lee Teng-hui promulgated it 19 years ago, Wang said, and the [DPP] did not try to make any changes to the framework during its eight-year rule between 2000 and 2008 either"*.

Label transformation is robust to misclassification. For example, any of BEFORE_START*, BEGINNING, WITHIN, or ENDING for a predecessor relation R_{pre} will map to before_start* for R. But other types of errors propagate and can lead to disastrous results. For example, due to a normalization quirk *"Utatu President George Strauss"* is recognized as *"Johannes Rau"*, thus the relation `title`(Rau, President) was assigned WITHIN at DCT, which is converted to a BEFORE_START* for Horst Kohler, Rau's successor.

A deeper problem that can lead to error propagation is that fact one person can have the same title in different contexts. When a title is attributed to a person there is often a geo-political or organization entity involved. Mentions that fail to include this third entity are ambiguous; often, this information needs to be inferred from other context sentences. Such errors may be propagated from supporter to official queries. For example, *"Francophonie president Abdou Diouf of Senegal ... "* appears to support the `title`(Abdou Diouf, President). Diouf preceded Abdoulaye Wade as President of Senegal, but the context in question (inaccurately) refers to Diouf's leadership position of Secretary-General (not President) of Organisation internationale de la Francophonie, thus an erroneous BEFORE_START* is aggregated, blocking a correctly labeled (less confident) $\langle r, 2000, \text{START} \rangle$.

7 Conclusion

We have analyzed within the particular context of TSF the process of aggregating partially-specified temporal information about relations across documents. Our analysis and and results indicate that text mentions of relations often ground only a portion of the referent relation in time and that correct interpretation relies on background knowledge about relation participants. In future work we plan a more rigorous data-driven study of nominal time of predication and to attack more ambiguous context-sensitive cases. In addition we aim to induce relation order from text automatically to multiple relation types as well as events.

Acknowledgments

This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), and in addition the U.K. Ministry of Defense under Agreement No. W911NF-06-3-0001 (ITA), U.S. NSF CAREER Award under Grant IIS-0953149, U.S. DARPA Award No. FA8750-13-2-0041 in the Deep Exploration and Filtering of Text (DEFT) Program, IBM Faculty Award, Google Research Award and RPI faculty start-up grant. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

[10] Had the title fix not been applied these WITHIN labels would have been converted to BEFORE_START*.

References

Enrique Amigo, Artiles Javier, Qi Li, and Heng Ji. 2011. An evaluation framework for aggregated temporal information extraction. In *Pric SIGIR2011 Workshop on Entity-Oriented Search*.

Javier Artiles, Qi Li, Taylor Cassidy, and Heng Ji. 2011. Temporal slot filling system description. In *Proc. Text Analytics Conference (TAC2011)*.

Maximilian Dylla, Iris Miliaraki, and Martin Theobald. 2013. A temporal-probabilistic database model for information extraction. *Proceedings of the VLDB Endowment*, 6(14):1810–1821.

Guillermo Garrido, Anselmo Penas, and Bernardo Cabaleiro. 2013. Uned slot filling and temporal slot filling systems at tac kbp 2013. system description. In *Proc. Text Analytics Conference (TAC2013)*.

Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. An overview of the tac2011 knowledge base population track. In *Proc. Text Analytics Conference (TAC2011)*.

Stuart J. Russell and Peter Norvig. 2010. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education.

Mihai Surdeanu. 2013. An overview of the tac2013 knowledge base population track. In *Proc. Text Analytics Conference (TAC2013)*.

Judith Tonhauser. 2002. A dynamic semantic account of the temporal interpretation of noun phrases. In *Proceedings of SALT*, volume 12, pages 286–305.

Naushad UzZaman, Hector Llorens, James F. Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. 2012. Tempeval-3: Evaluating events, time expressions, and temporal relations. *CoRR*, abs/1206.5333.

Yafang Wang, Maximilian Dylla, Marc Spaniol, and Gerhard Weikum. 2012. Coupling label propagation and constraints for temporal fact extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 233–237. Association for Computational Linguistics.

Appendix A. Glossary of Selected Terms

Fluent Relation: A property of a person or organization whose value may change over time. For example, a person's employer.
Temporal Extension: For a relation R, the temporal extension is the interval $[R_s, R_e]$, which represents the period of time between and including the start date R_s and end date R_e of the relation.
Relation Mention: An excerpt of text that expresses a relation.
Time Expression: An excerpt of text that refers to a portion of time, such as "Tuesday" or "next year".
Normalized Time Expression: The portion of time indicated by a time expression expressed in a standard form.
Granularity: The level at which a portion of time is expressed, in terms of calendar and clock units. For example, years are of a coarser granularity than days.
Temporal Four-tuple: For a relation R, a temporal four-tuple $T_R = \left\langle t^{(1)}, t^{(2)}, t^{(3)}, t^{(4)} \right\rangle$ represents an assertion that, based on some evidence, the start date for R is between $t^{(1)}$ and $t^{(2)}$, and its end date is between $t^{(3)}$ and $t^{(4)}$.
Final Temporal Four-tuple: The four-tuple assigned to R (by an annotator or system) after aggregating all temporal information about R.
Valid Temporal Four-tuple: A four-tuple $T = \left\langle t^{(1)}, t^{(2)}, t^{(3)}, t^{(4)} \right\rangle$ is valid if and only if iff. $t^{(1)} \leq t^{(2)} \wedge t^{(3)} \leq t^{(4)} \wedge t^{(1)} \leq t^{(4)}$.
Correct Temporal Four-tuple: A temporal four-tuple $T_R = \left\langle t^{(1)}, t^{(2)}, t^{(3)}, t^{(4)} \right\rangle$ if and only if $t^{(1)} \leq R_s \leq t^{(2)} \wedge t^{(3)} \leq R_e \leq t^{(4)}$
Intermediate Temporal Relationship: Given a relation mention r of relation R and a normalized time expression γ (viewed as a temporal interval), the intermediate temporal relationship between the two characterizes the relationships between the end points of γ and the endpoints of the temporal extension of R, namely γ_s, γ_e, R_s, and R_e. In this work, each intermediate temporal relationship used serves as a mapping from temporal interval to four-tuple (see Table 1 for the relationships used in this work and their mappings).

Intermediate Temporal Four-tuple Set: For a relation R, a system or annotator may derive an intermediate temporal four-tuple for each relation mention r and a corresponding time expression γ by based on an intermediate temporal relationship expressed between the two. The elements of each intermediate four-tuple are derived using the mapping in Table 1. We denote the set of intermediate temporal four-tuples for R as $I(R)$.
Query Relation: A relation that serves as input to a TSF system tasked with returning a final temporal four-tuple for that relation.
Relational Noun Phrase: A noun phrase that expresses a relation. For example, "President Obama" expresses a relation that "Obama"'s title is "President".
Time of Predication: For a given predicate, the time of predication is a time interval for which the predicate is asserted to apply to a specified set of arguments.
Post-relational State: A state immediately following the end of a relation characterized by the relation now longer holding. For example, prepending a title with "former", as in "former President X", introduces a state characterized by X no longer holding the title President.
Temporally Linked Relations: Two relations are temporally linked if their temporal extensions are not independent. For example, if it is known that one's end precedes the other's start.
Provenance: The relevant text that supports the output.

The Wisdom of Minority: Unsupervised Slot Filling Validation based on Multi-dimensional Truth-Finding

Dian Yu[1], Hongzhao Huang[1], Taylor Cassidy[2,3], Heng Ji[1]
Chi Wang[4], Shi Zhi[4], Jiawei Han[4], Clare Voss[2], Malik Magdon-Ismail[1]

[1]Computer Science Department, Rensselaer Polytechnic Institute
[2]U.S. Army Research Lab [3]IBM T. J. Watson Research Center
[4]Computer Science Department, Univerisity of Illinois at Urbana-Champaign
[1]{yud2,huangh9,jih,magdon}@rpi.edu,
[2,3]{taylor.cassidy.ctr,clare.r.voss.civ}@mail.mil
[4]{chiwang1,shizhi2,hanj}@illinois.edu

Abstract

Information Extraction using multiple information sources and systems is *beneficial* due to multi-source/system consolidation and *challenging* due to the resulting inconsistency and redundancy. We integrate IE and truth-finding research and present a novel unsupervised *multi-dimensional truth finding* framework which incorporates signals from multiple sources, multiple systems and multiple pieces of evidence by knowledge graph construction through multi-layer deep linguistic analysis. Experiments on the case study of Slot Filling Validation demonstrate that our approach can find truths accurately (9.4% higher F-score than supervised methods) and efficiently (finding 90% truths with only one half the cost of a baseline without credibility estimation).

1 Introduction

Traditional Information Extraction (IE) techniques assess the ability to *extract information from individual documents in isolation*. However, similar, complementary or conflicting information may exist in multiple heterogeneous sources. We take the Slot Filling Validation (SFV) task of the NIST Text Analysis Conference Knowledge Base Population (TAC-KBP) track (Ji et al., 2011) as a case study. The Slot Filling (SF) task aims at collecting from a large-scale multi-source corpus the values ("slot fillers") for certain attributes ("slot types") of a query entity, which is a person or some type of organization. KBP 2013 has defined 25 slot types for persons (per) (*e.g.*, age, spouse, employing organization) and 16 slot types for organizations (org) (*e.g.*, founder, headquarters-location, and subsidiaries). Some slot types take only a single slot filler (*e.g.*, per:birthplace), whereas others take multiple slot fillers (*e.g.*, org:top employees).

We call a combination of query entity, slot type, and slot filler a ***claim***. Along with each claim, each system must provide the ID of a source document and one or more detailed context sentences as ***evidence*** which supports the claim. A ***response*** (*i.e.*, a claim, evidence pair) is *correct* if and only if the claim is true *and* the evidence supports it.

Given the responses produced by multiple systems from multiple sources in the SF task, the goal of the SFV task is to determine whether each response is true or false. Though it's a promising line of research, it raises two complications: (1) different information ***sources*** may generate claims that vary

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1567–1578, Dublin, Ireland, August 23-29 2014.

in trustability; and (2) a large-scale number of SF ***systems*** using different resources and algorithms may generate erroneous, conflicting, redundant, complementary, ambiguously worded, or inter-dependent claims from the same set of documents. Table 1 presents responses from four SF systems for the query entity ***Ronnie James Dio*** and the slot type ***per:city_of_death***. Systems A, B and D return *Los Angeles* with different pieces of evidence [1] extracted from different information sources, though the evidence of System D does not decisively support the claim. System C returns *Atlantic City*, which is neither true nor supported by the corresponding evidence.

Such complications call for "*truth finding*": determining the *veracity* of multiple conflicting claims from various sources and systems. We propose a novel unsupervised multi-dimensional truth finding framework to study credibility perceptions in rich and wide contexts. It incorporates signals from multiple sources and systems, using linguistic indicators derived from knowledge graphs constructed from multiple evidences using multi-layer deep linguistic analysis. Experiments demonstrate that our approach can find truths accurately (9.4% higher F-score than supervised methods) and efficiently (find 90% truths with only one half cost of a baseline without credibility estimation).

System	Source	Slot Filler	Evidence
A	Agence France-Presse, News	Los Angeles	The statement was confirmed by publicist Maureen O'Connor, who said ***Dio*** died in ***Los Angeles***.
B	New York Times, News	Los Angeles	***Ronnie James Dio***, a singer with the heavy-metal bands Rainbow, Black Sabbath and Dio, whose semioperatic vocal style and attachment to demonic imagery made him a mainstay of the genre, died on Sunday in ***Los Angeles***.
C	Discussion Forum	Atlantic City	***Dio*** revealed last summer that he was suffering from stomach cancer shortly after wrapping up a tour in ***Atlantic City***.
D	Associated Press Worldstream, News	Los Angeles	***LOS ANGELES*** 2010-05-16 20:31:18 UTC ***Ronnie James Dio***, the metal god who replaced Ozzy Osbourne in Black Sabbath and later piloted the bands Heaven, Hell and Dio, has died, according to his wife and manager.

Table 1: Conflicting responses across different SF systems and different sources (query entity = ***Ronnie James Dio***, slot type = ***per:city_of_death***).

2 Related Work & Our Novel Contributions

Most previous SFV work (*e.g.*, (Tamang and Ji, 2011; Li and Grishman, 2013)) focused on filtering incorrect claims from multiple systems by simple heuristic rules, weighted voting, or costly supervised learning to rank algorithms. We are the first to introduce the truth finding concept to this task.

The "truth finding" problem has been studied in the data mining and database communities (*e.g.*, (Yin et al., 2008; Dong et al., 2009a; Dong et al., 2009b; Galland et al., 2010; Blanco et al., 2010; Pasternack and Roth, 2010; Yin and Tan, 2011; Pasternack and Roth, 2011; Vydiswaran et al., 2011; Ge et al., 2012; Zhao et al., 2012; Wang et al., 2012; Pasternack and Roth, 2013)). Compared with the previous work, our truth finding problem is defined under a unique setting: each *response* consists of a claim and supporting evidence, automatically generated from unstructured natural language texts by a SF *system*. The judgement of a *response* concerns both the truth of the claim and whether the *evidence* supports the claim. This has never been modeled before. We mine and exploit rich linguistic knowledge from multiple lexical, syntactic and semantic levels from evidence sentences for truth finding. In addition, previous truth finding work assumed most claims are likely to be true. However, most SF systems have hit a performance ceiling of 35% F-measure, and false responses constitute the majority class (72.02%) due to the imperfect algorithms as well as the inconsistencies of information sources. Furthermore, certain truths might only be discovered by a minority of good systems or from a few good sources. For example, 62% of the true responses are produced only by 1 or 2 of the 18 SF systems.

[1] Hereafter, we refer to "pieces of evidence" with the shorthand "evidences". Note that SF systems may include multiple sentences as "evidence" within their responses.

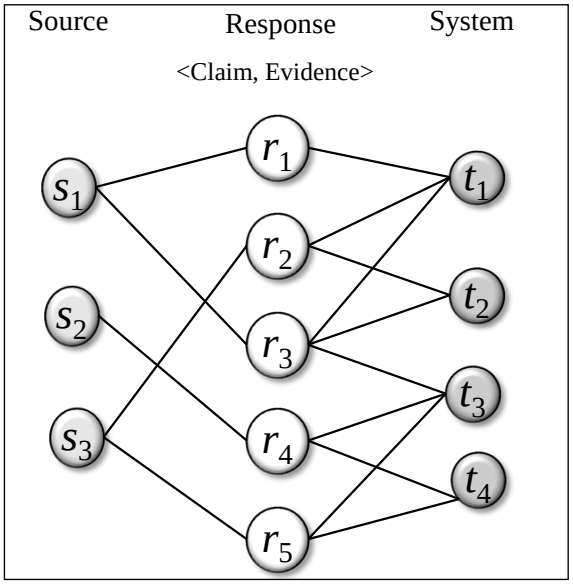


Figure 1: Heterogeneous networks for MTM.

3 MTM: A Multi-dimensional Truth-Finding Model

MTM Construction

A response is trustworthy if its claim is true and its evidence supports the claim. A trusted source always supports true claims by giving convincing evidence, and a good system tends to extract trustworthy responses from trusted sources. We propose a *multi-dimensional truth-finding model (MTM)* to incorporate and compute multi-dimensional credibility scores.

Consider a set of responses $R = \{r_1, \ldots, r_m\}$ extracted from a set of sources $S = \{s_1, \ldots, s_n\}$ and provided by a set of systems $T = \{t_1, \ldots, t_l\}$. A heterogeneous network is constructed as shown in Fig. 1. Let weight matrices be $W^{rs}_{m\times n} = \{w^{rs}_{ij}\}$ and $W^{rt}_{m\times l} = \{w^{rt}_{ik}\}$. A link $w^{rs}_{ij} = 1$ is generated between r_i and s_j when response r_i is extracted from source s_j, and a link $w^{rt}_{ik} = 1$ is generated between r_i and t_k when response r_i is provided by system t_k.

Credibility Initialization

Each source is represented as a combination of publication venue and genre. The credibility scores of sources S are initialized uniformly as $\frac{1}{n}$, where n is the number of sources. Given the set of systems $T = \{t_1, \ldots, t_l\}$, we initialize their credibility scores $c^0(t)$ based on their interactions on the predicted responses. Suppose each system t_i generates a set of responses R_{t_i}. The similarity between two systems t_i and t_j is defined as $similarity(t_i, t_j) = \frac{|R_{t_i} \cap R_{t_j}|}{\log(|R_{t_i}|)+\log(|R_{t_j}|)}$ (Mihalcea, 2004). Then we construct a weighted undirected graph $G = \langle T, E\rangle$, where $T(G) = \{t_1, \ldots, t_l\}$ and $E(G) = \{\langle t_i, t_j\rangle\}$, $\langle t_i, t_j\rangle = similarity(t_i, t_j)$, and apply the TextRank algorithm (Mihalcea, 2004) on G to obtain $c^0(t)$.

We got negative results by initializing system credibility scores uniformly. We also got negative results by initializing system credibility scores using system metadata, such as the algorithms and resources the system used at each step, its previous performance in benchmark tests, and the confidence values it produced for its responses. We found the quality of an SF system depends on many different resources instead of any dominant one. For example, an SF system using a better dependency parser does not necessarily produce more truths. In addition, many systems are actively being improved, rendering previous benchmark results unreliable. Furthermore, most SF systems still lack reliable confidence estimation.

The initialization of the credibility scores for responses relies on deep linguistic analysis of the evidence sentences and the exploitation of semantic clues, which will be described in Section 4.

Credibility Propagation

We explore the following heuristics in MTM.

HEURISTIC 1: *A response is more likely to be true if derived from many trustworthy sources. A source is more likely to be trustworthy if many responses derived from it are true.*

HEURISTIC 2: *A response is more likely to be true if it is extracted by many trustworthy systems. A system is more likely to be trustworthy if many responses generated by it are true.*

Based on these two heuristics we design the following credibility propagation approach to mutually reinforce the trustworthiness of linked objects in MTM.

By extension of Co-HITS (Deng et al., 2009), designed for bipartite graphs, we develop a propagation method to handle heterogeneous networks with three types of objects: *source, response* and *system*. Let the weight matrices be W^{rs} (between responses and sources) and W^{rt} (between responses and systems), and their transposes be W^{sr} and W^{tr}. We can obtain the transition probability that vertex s_i in S reaches vertex r_j in R at the next iteration, which can be formally defined as a normalized weight $p_{ij}^{sr} = \frac{w_{ij}^{sr}}{\sum_k w_{ik}^{sr}}$ such that $\sum_{r_j \in R} p_{ij}^{sr} = 1$. We compute the transition probabilities p_{ji}^{rs}, p_{jk}^{rt} and p_{kj}^{tr} in an analogous fashion.

Given the initial credibility scores $c^0(r)$, $c^0(s)$ and $c^0(t)$, we aim to obtain the refined credibility scores $c(r)$, $c(s)$ and $c(t)$ for responses, sources, and systems, respectively. Starting with sources, the update process considers both the initial score $c^0(s)$ and the propagation from connected responses, which we formulated as:

$$c(s_i) = (1 - \lambda_{rs})c^0(s_i) + \lambda_{rs} \sum_{r_j \in R} p_{ji}^{rs} c(r_j) \tag{1}$$

Similarly, the propagation from responses to systems is formulated as:

$$c(t_k) = (1 - \lambda_{rt})c^0(t_k) + \lambda_{rt} \sum_{r_j \in R} p_{jk}^{rt} c(r_j) \tag{2}$$

Each response's score $c(r_j)$ is influenced by both linked sources and systems:

$$c(r_j) = (1 - \lambda_{sr} - \lambda_{tr})c^0(r_j) + \lambda_{sr} \sum_{s_i \in S} p_{ij}^{sr} c(s_i) + \lambda_{tr} \sum_{t_k \in T} p_{kj}^{tr} c(t_k) \tag{3}$$

where $\lambda_{rs}, \lambda_{rt}, \lambda_{sr}$ and $\lambda_{tr} \in [0, 1]$. These parameters control the preference for the propagated over initial score for every type of random walk link. The larger they are, the more we rely on link structure[2]. The propagation algorithm converges (10 iterations in our experimental settings) and a similar theoretical proof to HITS (Peserico and Pretto, 2009) can be constructed. Algorithm 1 summarizes MTM.

4 Response Credibility Initialization

Each evidence along with a claim is expressed as a few natural language sentences that include the query entity and the slot filler, along with semantic content to support the claim. We analyze the evidence of each response in order to initialize that response's credibility score. This is done using heuristic rules defined in terms of the binary outputs of various *linguistic indicators* (Section 4.1).

4.1 Linguistic Indicators

We encode linguistic indicators based on deep linguistic knowledge acquisition and use them to determine whether responses provide supporting clues or carry negative indications (Section 4.3). These indicators make use of linguistic features on varying levels - surface form, sentential syntax, semantics, and pragmatics - and are defined in terms of knowledge graphs (Section 4.2). We define a heuristic rule for each slot type in terms of the binary-valued linguistic indicator outputs to yield a single binary value (1 or 0) for each response. If a response's linguistic indicator value is 1, the credibility score of a response is initialized at 1.0, and 0.5 otherwise.

[2]We set $\lambda_{rs} = 0.9$, $\lambda_{sr} = 0.1$, $\lambda_{rt} = 0.3$ and $\lambda_{tr} = 0.2$, optimized from a development set. See Section 5.1.

Input: A set of responses (R), sources (S) and systems (T).
Output: Credibility scores ($c(r)$) for R.

Initialize the credibility scores $c^0(s)$ for S as $c^0(s_i) = \frac{1}{|S|}$;
Use TextRank to compute initial credibility scores $c^0(t)$ for T;
Initialize the credibility scores $c^0(r)$ using linguistic indicators (Section 4);
Construct heterogeneous networks across R, S and T;
$k \leftarrow 0$, diff $\leftarrow 10e6$;
while $k <$ MaxIteration and diff $>$ MinThreshold **do**
    Use Eq. (1) to compute $c^{k+1}(s)$;
    Use Eq. (2) to compute $c^{k+1}(t)$;
    Use Eq. (3) to compute $c^{k+1}(r)$;
    Normalize $c^{k+1}(s)$, $c^{k+1}(t)$, and $c^{k+1}(r)$;
    diff $\leftarrow \sum(|c^{k+1}(r) - c^k(r)|)$;
    $k \leftarrow k + 1$
end while

Algorithm 1: Multi-dimensional Truth-Finding.

4.2 Knowledge Graph Construction

A semantically rich knowledge graph is constructed that links a query entity, all of its relevant slot filler nodes, and nodes for other intermediate elements excerpted from evidence sentences. There is one knowledge graph per sentence.

Fig. 2 shows a subregion of the knowledge graph built from the sentence: "***Mays***, ***50***, *died in his sleep at his* ***Tampa*** *home the morning of* ***June 28***.". It supports 3 claims: [*Mays, per: city_of_death, Tampa*], [*Mays, per: date_of_death, 06/28/2009*] and [*Mays, per: age, 50*].

Formally, a knowledge graph is an annotated graph of entity mentions, phrases and their links. It must contain one query entity node and one or more slot filler nodes. The annotation of a node includes its entity type, subtype, mention type, referent entities, and semantic category (though not every node has each type of annotation). The annotation of a link includes a dependency label and/or a semantic relation between the two linked nodes.

The knowledge graph is constructed using the following procedure. First, we annotate the evidence text using dependency parsing (Marneffe et al., 2006) and Information Extraction (entity, relation and event) (Li et al., 2013; Li and Ji, 2014). Two nodes are linked if they are deemed related by one of the annotation methods (*e.g.*, [*Mays, 50*] is labeled with the dependency type *amod*, and [*home, Tampa*] is labeled with the semantic relation *located_in*). The annotation output is often in terms of syntactic heads. Thus, we extend the boundaries of entity, time, and value mentions (*e.g.*, people's titles) to include an entire phrase where possible. We then enrich each node with annotation for entity type, subtype and mention type. Entity type and subtype refer to the role played by the entity in the world, the latter being more fine-grained, whereas mention type is syntactic in nature (it may be pronoun, nominal, or proper name). For example, "*Tampa*" in Fig. 2 is annotated as a *Geopolitical (entity type) Population-Center (subtype) Name (mention type)* mention. Every time expression node is annotated with its normalized reference date (*e.g.*, "*June, 28*" in Fig. 2 is normalized as "*06/28/2009*").

Second, we perform co-reference resolution, which introduces implicit links between nodes that refer to the same entity. Thus, an entity mention that is a nominal or pronoun will often be co-referentially linked to a mention of a proper name. This is important because many queries and slot fillers are expressed only as nominal mentions or pronouns in evidence sentences, their canonical form appearing elsewhere in the document.

Finally, we address the fact that a given relation type may be expressed in a variety of ways. For example, "*the face of*" indicates the membership relation in the following sentence: "*Jennifer Dunn was the face of the Washington state Republican Party for more than two decades*." We mined a large

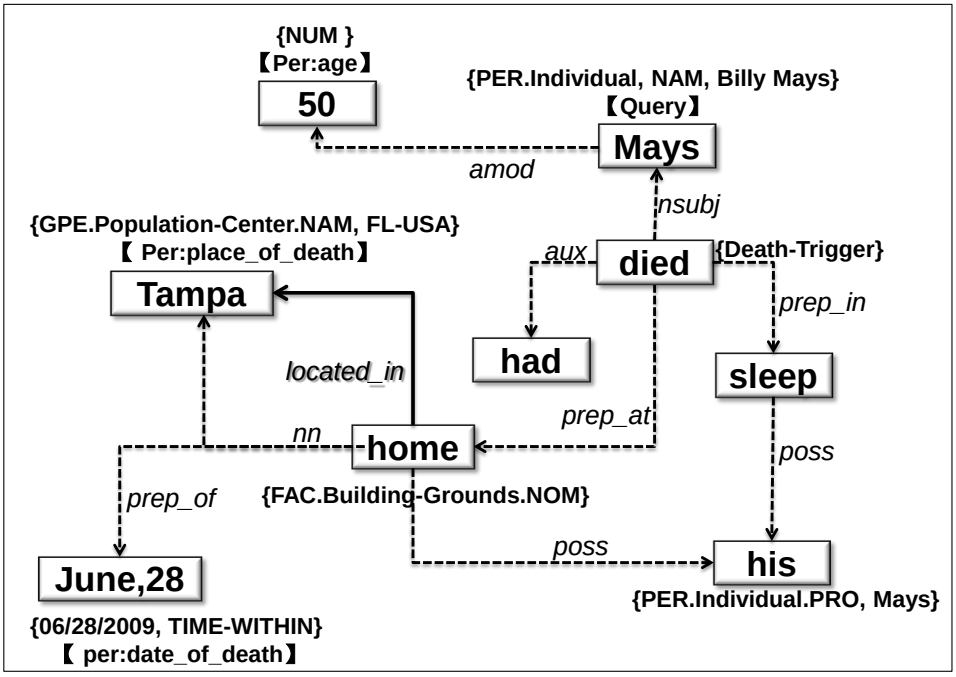


Figure 2: Knowledge Graph Example.

number of trigger phrases for each slot type by mapping various knowledge bases, including Wikipedia Infoboxes, Freebase (Bollacker et al., 2008), DBPedia (Auer et al., 2007) and YAGO (Suchanek et al., 2007), into the Gigaword corpus[3] and Wikipedia articles via distant supervision (Mintz et al., 2009)[4]. Each intermediate node in the knowledge graph that matches a trigger phrase is then assigned a corresponding semantic category. For example, "*died*" in Fig. 2 is labeled a *Death-Trigger*.

4.3 Knowledge Graph-Based Verification

We design linguistic indicators in terms of the properties of nodes and paths that are likely to be bear on the response's veracity. Formally, a *path* consists of the list of nodes and links that must be traversed along a route from a query node to a slot filler node.

Node indicators contribute information about a query entity or slot filler node in isolation, that may bear on the trustworthiness of the containing evidence sentence. For instance, a slot filler for the ***per:date_of_birth*** slot type must be a time expression.

Node Indicators

1. *Surface*: Whether the slot filler includes stop words; whether it is lower cased but appears in news. These serve as negative indicators.

2. *Entity type, subtype and mention type*: For example, the slot fillers for "*org:top_employees*" must be person names; and fillers for "*org:website*" must match the url format. Besides the entity extraction system, we also exploited the entity attributes mined by the NELL system (Carlson et al., 2010) from the KBP source corpus.

Each path contains syntactic and/or semantic relational information that may shed light on the manner in which the query entity and slot filler are related, based on dependency parser output, IE output, and trigger phrase labeling. Path indicators are used to define properties of the context in which which query-entity and slot-filler are related in an evidence sentence. For example, whether the path

[3]http://catalog.ldc.upenn.edu/LDC2011T07

[4]Under the distant supervision assumption, sentences that appear to mention both entities in a binary relation contained in the knowledge base were assumed to express that relation.

associated with a claim about an organization's top employee includes a title commonly associated with decision-making power can be roughly represented using the *trigger phrases* indicator.

Path Indicators

1. *Trigger phrases*: Whether the path includes any trigger phrases as described in Section 4.2.

2. *Relations and events*: Whether the path includes semantic relations or events indicative of the slot type. For example, a "*Start-Position*" event indicates a person becomes a "*member*" or "*employee*" of an organization.

3. *Path length*: Usually the length of the dependency path connecting a query node and a slot filler node is within a certain range for a given slot type. For example, the path for "*per:title*" is usually no longer than 1. A long dependency path between the query entity and slot filler indicates a lack of a relationship. In the following evidence sentence, which does not entail the "*per:religion*" relation between "*His*" and the religion "*Muslim*", there is a long path ("*his-poss-moment-nsubj-came-advcl-seized-militant-acmod-Muslim*"): "*His most noticeable moment in the public eye came in 1979, when Muslim militants in Iran seized the U.S. Embassy and took the Americans stationed there hostage.*".

Detecting and making use of interdependencies among various claims is another unique challenge in SFV. After initial response credibility scores are calculated by combining linguistic indicator values, we identify responses that have potentially conflicting or potentially supporting slot-filler candidates. For such responses, their credibility scores are changed in accordance with the binary values returned by the following indicators.

Interdependent Claims Indicators

1. *Conflicting slot fillers*: When fillers for two claims with the same query entity and slot type appear in the same evidence sentence, we apply an additional heuristic rule designed for the slot type in question. For example, the following evidence sentence indicates that compared to "*Cathleen P. Black*", "*Susan K. Reed*" is more likely to be in a "*org:top_employees/members*" relation with "*The Oprah Magazine*" due to the latter pair's shorter dependency path: "*Hearst Magazine's President Cathleen P. Black has appointed Susan K. Reed as editor-in-chief of the U.S. edition of The Oprah Magazine.*". The credibility scores are accordingly changed (or kept at) 0.5 for responses associated with the former claim, and 1.0 for those associated with the latter.

2. *Inter-dependent slot types*: Many slot types are inter-dependent, such as "*per:title*" and "*per:employee_of*", and various family slots. After determining initial credibility scores for each response, we check whether evidence exists for any implied claims. For example, given initial credibility scores of 1.0 for two responses supporting the claims that (1)"*David*" is "*per:children*" of "*Carolyn Goodman*" and (2)"*Andrew*" is "*per:sibling*" of "*David*", we check for any responses supporting the claim that (3)"*Andrew*" is "*per:children*" of "*Carolyn Goodman*", and set their credibility scores to 1.0. For example, a response supporting this claim included the evidence sentence, "*Dr. Carolyn Goodman, her husband, Robert, and their son, David, said goodbye to David's brother, Andrew.*".

5 Experimental Results

This section presents the experiment results and analysis of our approach.

5.1 Data

The data set we use is from the TAC-KBP2013 Slot Filling Validation (SFV) task, which consists of the merged responses returned by 52 runs (regarded as systems in MTM) from 18 teams submitted to the Slot

Methods	Precision	Recall	F-measure	Accuracy	Mean Average Precision
1.Random	28.64%	50.48%	36.54%	50.54%	34%
2.Voting	42.16%	70.18%	52.68%	62.54%	62%
3.Linguistic Indicators	50.24%	70.69%	58.73%	72.29%	60%
4.SVM (3 + System + Source)	56.59%	48.72%	52.36%	75.86%	56%
5.MTM (3 + System + Source)	53.94%	**72.11**%	**61.72**%	**81.57**%	**70**%

Table 2: Overall Performance Comparison.

Filling (SF) task. The source collection has 1,000,257 newswire documents, 999,999 web documents and 99,063 discussion forum posts, which results in 10 different sources (combinations of publication venues and genres) in our experiment. There are 100 queries: 50 person and 50 organization entities. After removing redundant responses within each single system run, we use 45,950 unique responses as the input to truth-finding. Linguistic Data Consortium (LDC) human annotators manually assessed all of these responses and produced 12,844 unique responses as ground truth. In order to compare with state-of-the-art supervised learning methods for SFV (Tamang and Ji, 2011; Li and Grishman, 2013), we trained a SVMs classifier [5] as a counterpart, incorporating the same set of linguistic indicators, sources and systems as features. We picked 10% (every 10th line) to compose the development set for MTM and the training set for the SVMs. The rest is used for blind test.

5.2 Overall Performance

Table 2 shows the overall performance of various truth finding methods on judging each response as true or false. MTM achieves promising results and even outperforms supervised learning approach. Table 3 presents some examples ranked at the top and the bottom based on the credibility scores produced by MTM.

We can see that majority voting across systems performs much better than random assessment, but its accuracy is still low. For example, the true claim *T5* was extracted by only one system because most systems mistakenly identified "*Briton Stuart Rose*" as a person name. In comparison, MTM obtained much better accuracy by also incorporating multiple dimensions of source and evidence information.

Method 3 using linguistic indicators alone, already achieved promising results. For example, many claims are judged as truths through trigger phrases (*T1* and *T5*), event extraction (*T2*), coreference (*T4*), and node type indicators (*T3*). On the other hand, many claims are correctly judged as false because their evidence sentences did not include the slot filler (*F1*, *F4*, *F5*) or valid knowledge paths to connect the query entity and the slot filler (*F2*, *F3*). The performance gain (2.99% F-score) from Method 3 to Method 5 shows the need for incorporating system and source dimensions. For example, most truths are from news while many false claims are from newsgroups and discussion forum posts (*F1*, *F2*, *F5*).

The SVMs model got very low recall because of the following two reasons: (1) It ignored the inter-dependency between multiple dimensions; (2) the negative instances are dominant in the training data, so the model is biased towards labeling responses as false.

5.3 Truth Finding Efficiency

Table 3 shows that some truths (*T1*) are produced from low-ranked systems whereas some false responses from high-ranked systems (*F1*, *F2*). Note that systems are ranked by their performance in KBP SF task. In order to find all the truths, human assessors need to go through all the responses returned by multiple systems. This process was proven very tedious and costly (Ji et al., 2010; Tamang and Ji, 2011).

Our MTM approach can expedite this process by ranking responses based on their credibility scores and asking human to assess the responses with high credibility first. Traditionally, when human assess responses, they follow an alphabetical order or system IDs in a "passive learning" style. This is set as our baseline. For comparison, we also present the results using only linguistic indicators, using voting in which the responses which get more votes across systems are assessed first, and the oracle method assessing all correct responses first. Table 2 shows our model can successfully rank trustworthy responses at high positions compared with other approaches.

[5] We used the LIBSVM toolkit (Chang and Lin, 2011) with Gaussian radial basis function kernel.

	Response Ranked by MTM							System Rank
	Claim				Evidence	Source		
	Query Entity		Slot Type	Slot Filler				
Top Truths	T1	China Banking Regulatory Commission	org:top members/employees	Liu Mingkang	**Liu Mingkang**, the chairman of the China Banking Regulatory Commission	Central News Agency of Taiwan News	News	15
	T2	Galleon Group	org:founded by	Raj Rajaratnam	Galleon Group, founded by billionaire **Raj Rajaratnam**	New York Times	News	9
	T3	Mike Penner	per:age	52	L.A. Times Sportswriter Mike Penner, **52**, Dies	New York Times	News	1
	T4	China Banking Regulatory Commission	org:alternate names	CBRC	...China Banking Regulatory Commission said in the notice. The five banks ... according to **CBRC**.	Xinhua, News	News	5
	T5	Stuart Rose	per:origin	Briton	Bolland, 50, will replace **Briton** Stuart Rose at the start of 2010.	Agence France-Presse	News	3
Bottom False Claims	F1	American Association for the Advancement of Science	org:top members employees	Freedman	erica.html > American Library Association, President: Maurice **Freedman** < http://www.aft.org > American Federation of Teachers ...	Google	Newsgroup	4
	F2	Jade Goody	per:origin	Britain	because Jade Goody's the only person to ever I love **Britain**	Discussion Forum		3
	F3	Don Hewitt	per:spouse	Swap	...whether "Wife **Swap**" on ABC or "Jon & Kate" on TLC	New York Times	News	7
	F4	Council of Mortgage Lenders	org:website	www.cml.org.uk	he purchases in the U.K. jumped by 16 percent in April, suggesting the property market slump may have bottomed out	Associated Press World-stream	News	18
	F5	Don Hewitt	per:alternate names	Hewitt M-chen	US DoMIna THOMPson LACtaTe haVeD [3866 words]	Google	Newsgroup	13

Table 3: Top and Bottom Response Examples Ranked by MTM.

Fig. 3 summarizes the results from the above 6 approaches. The common end point of all curves represents the cost and benefit of assessing all system responses. We can see that the baseline is very inefficient at finding the truths. If we employ linguistic indicators, the process can be dramatically expedited. MTM provides further significant gains, with performance close to the Oracle. With only half the cost of the baseline, MTM can already find 90% truths.

5.4 Enhance Individual SF Systems

Finally, as a by-product, our MTM approach can also be exploited to validate the responses from each individual SF system based on their credibility scores. For fair comparison with the official KBP evaluation, we use the same ground-truth in KBP2013 and standard precision, recall and F-measure metrics as defined in (Ji et al., 2011). To increase the chance of including truths which may be particularly difficult for a system to find, LDC prepared a manual key which was assessed and included in the final ground truth. According to the SF evaluation setting, F-measure is computed based on the number of unique true claims. After removing redundancy across multiple systems, there are 1,468 unique true claims. The cutoff criteria for determining whether a response is true or not was optimized from the development set.

Fig. 4 presents the F-measure scores of the best run from each individual SF system. We can see that our MTM approach consistently improves the performance of almost all SF systems, in an absolute gain range of [-1.22%, 5.70%]. It promotes state-of-the-art SF performance from 33.51% to 35.70%. Our MTM approach provides more gains to SF systems which mainly rely on lexical or syntactic patterns than other systems using distant supervision or logic rules.

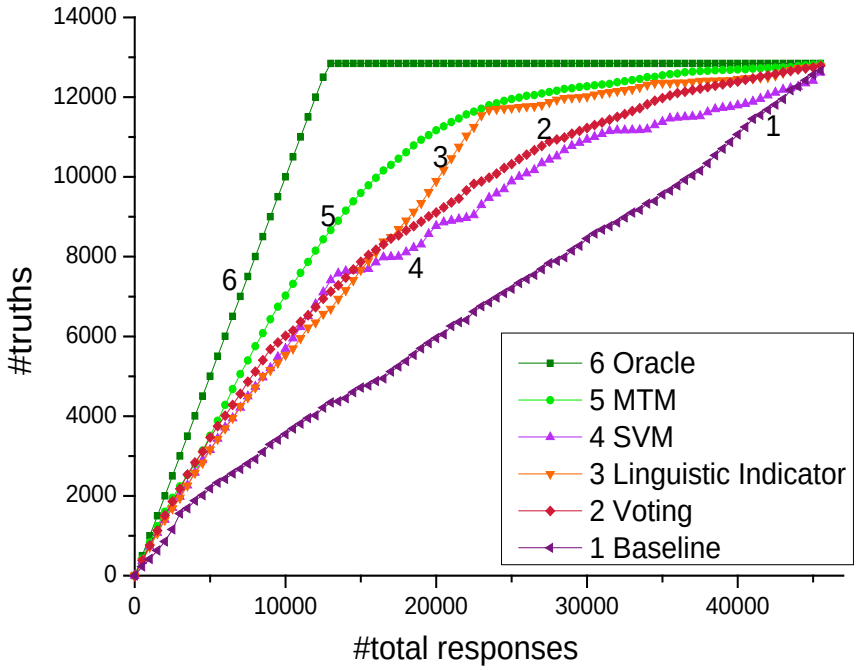


Figure 3: Truth Finding Efficiency.

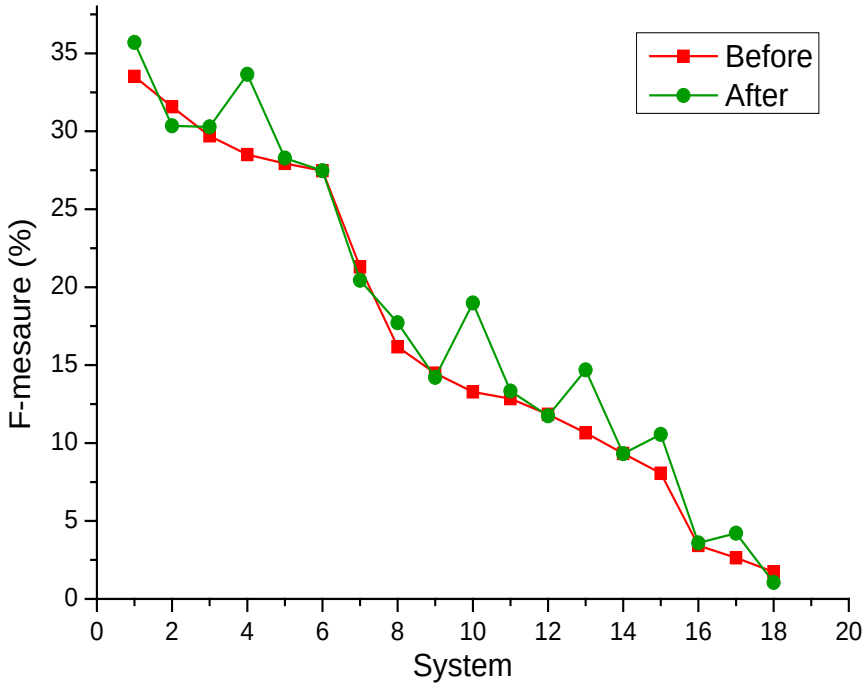


Figure 4: Impact on Individual SF Systems.

6 Conclusions and Future Work

Truth finding has received attention from both Natural Language Processing (NLP) and Data Mining communities. NLP work has mostly explored linguistic analysis of the content, while Data Mining work proposed advanced models in resolving conflict information from multiple sources. They have relative strengths and weaknesses. In this paper we leverage the strengths of these two distinct, but complementary research paradigms and propose a novel unsupervised multi-dimensional truth-finding framework incorporating signals both from multiple sources, multiple systems and multiple evidences based on knowledge graph construction with multi-layer linguistic analysis. Experiments on a challenging SFV task demonstrated that this framework can find high-quality truths efficiently. In the future we will focus on exploring more inter-dependencies among responses such as temporal and causal relations.

Acknowledgments

This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), the U.S. Army Research Office under Cooperative Agreement No. W911NF-13-1-0193, U.S. National Science Foundation grants IIS-0953149, CNS-0931975, IIS-1017362, IIS-1320617, IIS-1354329, U.S. DARPA Award No. FA8750-13-2-0041 in the Deep Exploration and Filtering of Text (DEFT) Program, IBM Faculty Award, Google Research Award, DTRA, DHS and RPI faculty start-up grant. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, and Z. Ives. 2007. Dbpedia: A nucleus for a web of open data. In *Proc. the 6th International Semantic Web Conference*.

L. Blanco, V. Crescenzi, P. Merialdo, and P. Papotti. 2010. Probabilistic models to reconcile complex data from inaccurate data sources. In *Proc. Int. Conf. on Advanced Information Systems Engineering (CAiSE'10)*, Hammamet, Tunisia, June.

K. Bollacker, R. Cook, and P. Tufts. 2008. Freebase: A shared database of structured general human knowledge. In *Proc. National Conference on Artificial Intelligence*.

A. Carlson, J. Betteridge, B. Kisiel, B. Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.

C. Chang and C. Lin. 2011. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

H. Deng, M. R. Lyu, and I. King. 2009. A generalized co-hits algorithm and its application to bipartite graphs. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 239–248, New York, NY, USA. ACM.

X. L. Dong, L. Berti-Equille, and D. Srivastavas. 2009a. Integrating conflicting data: The role of source dependence. In *Proc. 2009 Int. Conf. Very Large Data Bases (VLDB'09)*, Lyon, France, Aug.

X. L. Dong, L. Berti-Equille, and D. Srivastavas. 2009b. Truth discovery and copying detection in a dynamic world. In *Proc. 2009 Int. Conf. Very Large Data Bases (VLDB'09)*, Lyon, France, Aug.

A. Galland, S. Abiteboul, A. Marian, and P. Senellart. 2010. Corroborating information from disagreeing views. In *Proc. ACM Int. Conf. on Web Search and Data Mining (WSDM'10)*, New York, NY, Feb.

L. Ge, J. Gao, X. Yu, W. Fan, and A. Zhang. 2012. Estimating local information trustworthiness via multi-source joint matrix factorization. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 876–881. IEEE.

H. Ji, R. Grishman, H. T. Dang, K. Griffitt, and J. Ellis. 2010. An overview of the tac2010 knowledge base population track. In *Proc. Text Analytics Conf. (TAC'10)*, Gaithersburg, Maryland, Nov.

H. Ji, R. Grishman, and H.T. Dang. 2011. Overview of the tac 2011 knowledge base population track. In *Text Analysis Conf. (TAC) 2011*.

X. Li and R. Grishman. 2013. Confidence estimation for knowledge base population. In *Proc. Recent Advances in Natural Language Processing (RANLP)*.

Q. Li and H. Ji. 2014. Incremental joint extraction of entity mentions and relations.

Q. Li, H. Ji, and L. Huang. 2013. Joint event extraction via structured prediction with global features.

M. D. Marneffe, B. Maccartney, and C. D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*, pages 449,454.

R. Mihalcea. 2004. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proc. ACL2004*.

M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proc. ACL2009*.

J. Pasternack and D. Roth. 2010. Knowing what to believe (when you already know something). In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 877–885. Association for Computational Linguistics.

J. Pasternack and D. Roth. 2011. Making better informed trust decisions with generalized fact-finding. In *Proc. 2011 Int. Joint Conf. on Artificial Intelligence (IJCAI'11)*, Barcelona, Spain, July.

J. Pasternack and D. Roth. 2013. Latent credibility analysis. In *Proc. WWW 2013*.

E. Peserico and L. Pretto. 2009. Score and rank convergence of hits. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 770–771. ACM.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *16th international World Wide Web conference (WWW 2007)*, New York, NY, USA. ACM Press.

S. Tamang and H. Ji. 2011. Adding smarter systems instead of human annotators: Re-ranking for slot filling system combination. In *Proc. CIKM2011 Workshop on Search & Mining Entity-Relationship data*, Glasgow, Scotland, UK, Oct.

VG Vydiswaran, C.X. Zhai, and D. Roth. 2011. Content-driven trust propagation framework. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 974–982. ACM.

D. Wang, L. Kaplan, H. Le, and T. Abdelzaher. 2012. On truth discovery in social sensing: A maximum likelihood estimation approach. In *Proc. ACM/IEEE Int. Conf. on Information Processing in Sensor Networks (IPSN'12)*, pages 233–244, Beijing, China, April.

X. Yin and W. Tan. 2011. Semi-supervised truth discovery. In *Proc. 2011 Int. World Wide Web Conf. (WWW'11)*, Hyderabad, India, March.

X. Yin, J. Han, and P. S. Yu. 2008. Truth discovery with multiple conflicting information providers on the Web. *IEEE Trans. Knowledge and Data Engineering*, 20:796–808.

B. Zhao, B. I. P. Rubinstein, J. Gemmell, and J. Han. 2012. A Bayesian approach to discovering truth from conflicting sources for data integration. In *Proc. 2012 Int. Conf. Very Large Data Bases (VLDB'12)*, Istanbul, Turkey, Aug.

Common Space Embedding of Primal-Dual Relation Semantic Spaces

Hidekazu Oiwa*
The University of Tokyo
Tokyo, Japan
hidekazu.oiwa@gmail.com

Jun'ichi Tsujii
Microsoft Research
Beijing, China
jtsujii@microsoft.com

Abstract

Explicit continuous vector representation such as vector representation of words, phrases, etc. has been proven effective for various NLP tasks. This paper proposes a novel method of constructing such vector representation for both entity-pairs and relation expressions which link them in text. Based on the insight of the duality of relations, the representation is constructed by embedding of two separately constructed semantic spaces, one for entity-pairs and the other for relation expressions, into a common semantic space. By representing the two different types of objects (i.e. entity-pairs and relation expressions) in the same semantic space, we can treat the two tasks, relation mining and relation expression mining (a.k.a. pattern mining), systematically and in a unified manner. The approach is the first attempt to construct a continuous vector representation for expressions whose validity can be explicitly checked by their proximities to known sets of entity-pairs. We also experimentally validate the effectiveness of the common space for relation mining and relation expression mining.

1 Introduction

Learning continuous vector representation for expressions which consist of more than one word has gained attention in recent years. Various representations have been constructed and used to measure semantic similarities between expressions in various tasks, such as analogical reasoning (Turney et al., 2003; Mikolov et al., 2013) and sentiment analysis (Turney and Littman, 2003; Socher et al., 2012). Many algorithms have been proposed to construct such continuous representations, depending on specific tasks in mind. In this paper, we propose a method for constructing a vector representation for binary relations, i.e., relations with two arguments. We demonstrate the effectiveness of the representation for relation mining and relation expression mining.

The method exploits the duality of a relation (Bollegala et al., 2010). While Bollegala et al. (2010) uses the duality in their co-clustering algorithm, we construct an explicit semantic space which reflects the two aspects of a given relation. We first construct two separate semantic spaces, one for pairs of named entities and another for relation expressions in text which link an entity-pair. A relation is supposed to correspond to a subset in each of these two spaces. The subset of entity-pairs is a set of pairs between which the relation holds. The subset is called the extension set of the relation. The subset of relation expressions consists a set of expressions which are used to link entity-pairs in the extension set.

The two semantic spaces are then embedded into a single common space. Figure 1 illustrates a brief summary of constructing a common semantic space. While the subsets which correspond to a specific relation are supposed to constitute natural clusters in the two original spaces, objects in the two spaces exchange useful information to each other and form a tighter cluster in the common space. Exchange of information takes place through common space embedding.

Since both entity-pairs and relation expressions have their vector representations in the common semantic space, one can easily enumerate relation expressions specific to a certain set of entity-pairs (re-

*This project was conducted while the first author stayed at Microsoft Research Asia.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1579–1590, Dublin, Ireland, August 23-29 2014.

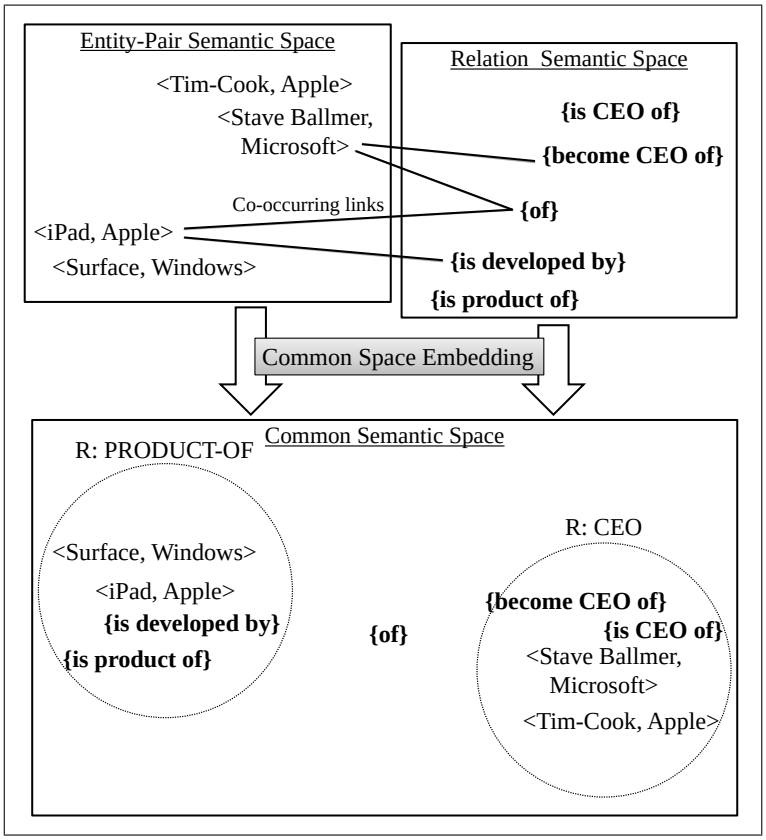


Figure 1: Overview of our framework to construct common semantic space.

lation expression mining). Furthermore, unlike the conventional pattern-based relation mining, one can perform relation mining in the common space without explicit reference to relation expressions.

2 Basic Framework

2.1 Duality of Relation and a Common Space

A binary relation is defined either extensionally by a set of pairs in the relation or intensionally by a set of conditions which a pair in the relation should satisfy. However, in actual applications of text mining, either of these definitions is given in a complete form. We are only given a subset of the whole set of pairs and have to complete the set (i.e. relation mining). Instead of an explicit intensional definition, we only have a set of observations in text where pairs in a relation are linked by certain linguistic expressions. Based on such observations, we have to judge whether a given pair holds the relation or not. Though some observed expressions are non-ambiguous and explicit for a relation (for example, "the birth place of A is B"), most of expressions are not (such as"A comes from B").

We call a set of pairs which define a relation as Extension set of a relation, while we call their observed expressions in text as Manifestation set. While these two sets are only partially given, they define relations which we are interested in. Such duality of a relation has been recognized by many previous work and has been exploited in relation mining and relation expression mining. (Bollegala et al., 2010), for example, used the duality in their work on co-clustering of entity-pairs and relation expressions. (Baroni and Lenci, 2010) presented a more general approach which defines a tensor associating a triplet $< e_1, l, e_2 >$ with a weight. e_1 and e_2 are entity pairs, while l is a linking expression in text. By projecting the tensor to matrices, they showed that diverse concepts used in distributional semantics could be captured in a unified manner. In particular, their tensors capture directly the duality of entity pairs and their linking expressions (i.e. relation expressions).

These previous works implicitly assume that the semantic space of entity pairs and that of relation ex-

pressions are tightly coupled. That is, the space of entity pairs is defined in terms of their co-occurrences with linking expressions (or the weights in a tensor between them) and vice versa. However, such tight coupling between the semantic spaces of entity pairs and relation expressions is not a logical necessity, and harmful in the sense that it restricts available information only to their co-occurrences.

An entity pair and a linking expression are complex objects by themselves, and their semantic spaces can be defined independently of each other. Two entities in a sentence, for example, are linked not only by single verbs or predicates but by a long sequence of words. This means that we can define a semantic space of linking expressions independently of entity pairs which they link. For example, one can use sequence similarities of words among relation expressions. Since knowledge resources of large scale have become available of late, we can define a semantic space of entity pairs by using paths in these knowledge graphs, regardless of their textual occurrences with relation expressions.

In this paper, we first define two separate semantic spaces (i.e. dual primal spaces) for entity pairs and relation expressions, and then use their textual co-occurrences to construct a common space consistent with the two primal spaces. In this approach, the co-occurrences of entity pairs with relation expressions play only an auxiliary role to project the two spaces into a common space.

The approach allows us to integrate information richer than mere co-occurrences of two objects (i.e. entity pairs and relation expressions). Furthermore, the common space provides us with direct means by which one can grasp finer grained relationships between two objects. Given a set of seed pairs of entities, one can gather a set of relation expressions in their nearest neighbor in the common space. Another set of seed pairs, even though conceptually they belong to the same relation, one may get a different set of relation expressions. The previous approaches, in which the semantics of the two objects are captured in two separate spaces, can capture only indirectly the hierarchical nature of natural relations, and how such a hierarchy is mapped on association of extension sets with manifestation sets.

2.2 Extension set and Manifestation Set

Let E be a set of named entities. Let $\langle e_i, e_j \rangle$ denote a pair of entities ($e_i, e_j \in E$) and E^2 a set of all entity-pairs. Then, a relation, R, is extensionally defined as a set of entity-pairs $E_R \subset E^2$, such as CEO = { $\langle$Tim-Cook, Apple$\rangle$, $\langle$Ballmer, Microsoft$\rangle$, ...}, COMPETE = {$\langle$Apple, Samsung$\rangle$, $\langle$Google, Microsoft$\rangle$, ...} between which the relation holds. We call such a set of entity-pairs the extension set of a relation R.

On the other hand, a relation R is manifested in text in various forms of expressions. For example, "is the CEO of" in "Tim-Cook is the CEO of Apple" is a direct manifestation of the relation CEO. While "overtook" in "Samsung overtook Apple in the smartphone market in China" can be a manifestation of the relation COMPETE, this manifestation is rather indirect, based on inference. We denote a relation expression by r_i and the whole set of relation expressions by D. We call a subset of relation expressions which manifest, directly or indirectly, a relation R, as the manifestation set of R.

2.3 Primal-dual semantic spaces

A relation, R, is characterized by the two sets, the extension set and the manifestation set. In other words, the two sets are implicitly associated with each other via the relation R. This association between the two sets constitutes the foundation of the common semantic space to be constructed in this paper.

We first construct primal-dual semantic spaces, one for entity-pairs and another for relation expressions. A sentence where two entities appear can be seen from two different perspectives. One view is to see the sentence as characterization of the entity-pair, while the other takes the sentence as characterization of the relation expression which links the two entities. Based on these two views, we construct two semantic spaces from a given set of sentences (corpus). One space is for a set of entity-pairs (E^2) and the other for a set of relation expressions (D). $e^2 \in E^2$ and $r \in D$ are represented by vectors $\mathbf{e}^2 \in \mathbf{E}^2$ and $\mathbf{r} \in \mathbf{D}$ in the corresponding spaces. We assume that the two spaces are vector spaces, i.e., $\mathbf{E}^2$ and $\mathbf{D}$ are an n-dimensional vector space and an m-dimensional one, respectively.

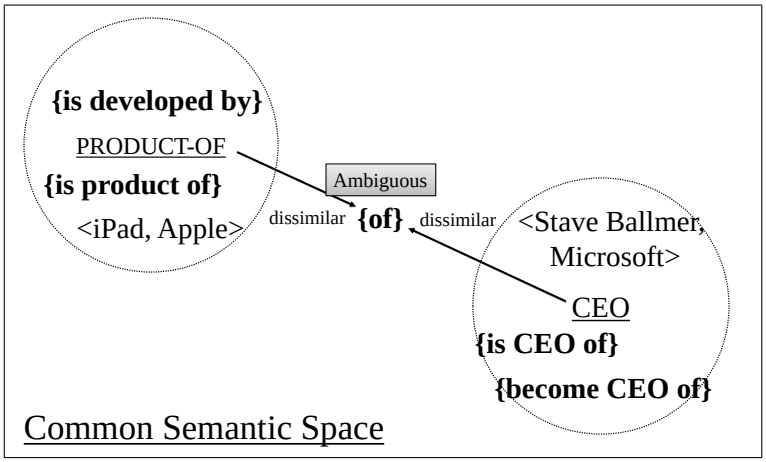


Figure 2: Illustration of common semantic space defined by our approach.

2.4 Triplets

The two objects, entity-pairs and relation expressions, whose spaces are separate, are linked through their co-occurrences in text. Co-occurrence of a relation expression (r) and an entity-pair ($e^2 = \langle e_1^2, e_2^2 \rangle$) means that r links in a sentence the entities of e_1^2 and e_2^2. A triplet represents such a co-occurrence with its frequency ($f \in \mathcal{R}$) in text. An instance of triplets is denoted as $\langle \mathbf{e}^2, \mathbf{r}, f \rangle \in T$. T indicates a set of triplets. These co-occurrence frequencies between entity-pairs and relation expressions play a critical role in common space embedding as linkage clues.

2.5 Common space embedding from $\mathbf{E}^2$ and $\mathbf{D}$

We use Multi-View Partial Least Squares (MVPLS) (Wu et al., 2013) as the basic framework to construct a common space from $\mathbf{E}^2 \subset \mathcal{R}^m$ and $\mathbf{D} \subset \mathcal{R}^n$. MVPLS was originally developed for web search and has been proven to be effective for embedding the semantic space of queries and that of documents into a common space. This framework is an extension of the conventional well-used approach, Partial Least Square. The framework is general enough to be used for our purpose.

Let k be the dimension of common latent space such that $k \leq m$ and $k \leq n$. $\mathbf{e}_i^2 \in \mathbf{E}^2$ is a i-th entity-pair feature vector in the entity-pair space and $\mathbf{r}_i \in \mathbf{D}$ is a i-th phrase feature vector. $\mathbf{L}_e$, $\mathbf{L}_r$ are linear projection matrices for embedding the original feature vector space into the common latent space. $\mathbf{L}_e$ is $m \times k$ and $\mathbf{L}_r$ is $n \times k$ size matrices.

MVPLS learns these two projection matrices for generating a well-constructed common space from the two separated spaces. Construction of latent common space can be formulated as an optimization problem which maximizes the sum of the similarities between entity-pairs and relation expressions in the common space when they co-occur. This optimization problem is as follows:

$$\underset{\mathbf{L}_e, \mathbf{L}_r}{\operatorname{argmax}} \sum_{(\mathbf{e}_i^2, \mathbf{r}_i, f_i) \in T} \log(f_i) \mathbf{r}_i^T \mathbf{L}_r \mathbf{L}_e^T \mathbf{e}_i^2 \quad s.t. \quad \mathbf{L}_e^T \mathbf{L}_e = \mathbf{I}, \quad \mathbf{L}_r^T \mathbf{L}_r = \mathbf{I} \,. \tag{1}$$

Note that the similarity score is weighted by the logarithmic scale of the co-occurrence counts. The outputs of this optimization problem are $\mathbf{L}_e$ and $\mathbf{L}_r$ which maximize the objective value where the orthogonal constraints on these matrices are satisfied. We do not necessarily solve (1) again when the system receives a new instance because the derived matrices can be applied not only for the existing entity-pairs and relation expressions but new ones. The problem is not convex, but Wu et al. (2013) proved that the global optimal solution can be obtained by SVD of $\sum_T \log(f_i) \mathbf{e}_i^2 \mathbf{r}_i^T$. $\mathbf{L}_e$ corresponds to left singular vectors and $\mathbf{L}_r$ consists of right singular vectors.

2.6 Ambiguity of Relation Expressions in the Common Space

Due to the ambiguity of relation expressions, the assumption that the manifestation set of the same R cluster around in proximity does not hold in reality. "of" in "Steve Ballmer of Microsoft" belongs to

the manifestation set of CEO, while "of" in "iPad of Apple" belongs to the set of a different relation, PRODUCT-OF. Indirect manifestation such as "overtake" is another cause of ambiguity. Inference involved here is abductive in nature and not always valid. We may be able to infer COMPETE relation from "X overtake Y", but "X overtake Y" can be a consequence of another relation such as COOPERATE.

Such an ambiguous expression belongs to the manifestation sets of more than one relation and thus would be located in a rather neutral position in the space. Since the common space reflects how frequently certain expressions are used to link entity-pairs, their positions in the space reflect the relative specificity to each relation cluster. Figure 2 illustrates how the ambiguity of a relation expression captured in the common space.

3 Relation Mining and Relation Expression Mining

In an actual situation, both the extension set and the manifestation set of a relation R are only partially known. To produce more comprehensive sets of these objects from large corpora is generally called mining. Two mining tasks have been studied so far, which are different, though mutually related.

We define relation mining as a task which, given a relation R, enumerates entity-pairs in the extension set. Another mining task (i.e. relation expression mining which is often performed as an auxiliary task of relation mining) is to gather a set of relation expressions which are manifestations of a given R.

3.1 Relation Mining

Relation mining is the task to enumerate entity-pairs of a relation R from a small given set of objects of a relation R. For example, if a set of relation expressions as the manifestation set of a relation R are given, one can produce a set of entity-pairs simply by identifying occurrences of relation expressions in text and producing the entity-pairs which are linked by them. Alternatively, if a small set of entity-pairs as a subset of the extension set of a relation R are given, one can produce a set of entity-pairs simply by gathering similar entity-pairs measured by relation expression co-occurrence vectors. These ideas have been shared by many mining systems called pattern-based relation mining systems.

The recall and precision of such a system are determined by the quality and quantity of the given set. If the given set is small, a system suffers low recall. On the other hand, if the set is large but contains many 'ambiguous' or 'weak' objects, a system suffers low precision.

Therefore, one of the keys for success of relation mining is how to gather a large initial set, which are effective, i.e. objects less ambiguous with high frequency. The common semantic space can be used not only to generate a comprehensive set but to measure the specificity of objects in terms of a given R, it also provides refined semantic measures between entity-pairs.

3.2 Relation Expression Mining

We have discussed semantic spaces of relation expressions and the common semantic space as if to define what constitutes a relation expression is straightforward. However, it is not trivial to define what constitutes a relation expression.

In the previous section, we treat "overtake" in "Apple overtook Samsung in the smart phone market" as a relation expression which manifests the relation "COMPETE". However, one may argue that a pattern such as "X overtake Y in ... market" should be treated as a basic unit of manifestation of the relation COMPETE. This longer expression is less ambiguous and thus more effective than the shorter pattern of "overtake". On the other hand, the frequency of this pattern would be much less and thus less effective, compared with the shorter version. Mining of effective relation expressions (sometimes called "pattern mining") has to address the problem of balancing the specificity and generality of relation expressions. Furthermore, one would like to identify the same relation expression in "Apple announced yesterday that it had overtaken Samsung which ..." as in "Apple overtook Samsung in the smart phone market".

In the experiments, we do not treat the process of pattern mining seriously. Instead, we used two conventional methods. The first method is to enumerate subsequences of words in the intervening part in a sentence between two entities, and use them as relation expressions. We expect less effective expressions as manifestation to be recognized in the common space. Another method is to use the shortest paths in

dependency structures of sentences as relation expressions. Shortest paths can generalize surface variants of essentially the same relation expressions and reduce unnecessary proliferation of relation expressions.

4 Experiments

This section empirically evaluates our approach of embedding the two original spaces into a common space. We show that the common space provides a continuous vector space for relation expressions, in which not only similarities among expressions but also their ambiguities are properly captured.

4.1 Experiment Settings

4.1.1 Dataset

We use the ENT benchmark dataset (Bollegala et al., 2009) for our experiments. The dataset consists of 661,502 snippets, which are brief summaries provided by Web search engines. Most web search engines provide links to webpages and snippets as search results and snippets contains a subset of texts including the query words derived from the webpages. Table 1 shows how many distinct entity pairs, relation expressions and triplets were extracted as results of NER and expression extraction (See Section 4.1.2 and 4.1.3). The dataset is accompanied with 100 entity-pairs that are classified into five semantic categories: ACQUISITION, HEADQUARTERS, FIELD, CEO, and BIRTHPLACE. We use the ENT dataset not only for evaluation of relation mining but also for examining the characteristics of the common space for relation expression mining. Note that, due to the nature of snippets, the dataset is very noisy. It contains many non-sentences and even non-English texts, which may adversely affect the performance of mining systems.

	Entity-pair	Relation	Triplet
Enumeration	12,174	12,185	521,454
Shortest Path	10,251	92,797	130,897

Table 1: The specifications of the ENT dataset: Sizes of distinct entity-pairs, relation expressions, and triplets. "Enumeration" indicates the results of pattern mining based on word subsequences. "Shortest Path" shows that of shortest path extraction.

4.1.2 Entity and Entity-Pair Extraction

We first extracted entities from the ENT dataset. After splitting snippets into sentences, we applied named entity recognizer (NER) (Finkel et al., 2005) to recognize entities in sentences. We used Stanford Core NLP tools 2[1] for sentence splitting and NER. As relevant semantic classes for the ENT dataset, entities which are recognized as ORGANIZATION, LOCATION, or PERSON are treated as entities in the further process. We only used sentences in which at least two entities of these three classes appear.

4.1.3 Extraction of Relation Expressions

The definition of relation expressions which link two entities in text is not trivial. We adopt two methods of extracting candidates of relation expressions, and compare them in experiments.

The first method is to use, as relation expressions, subsequences of words which appear between two entities. We assume that two entities which appear apart in a sentence by more than 10 words are not explicitly linked in the sentence. From the word sequence whose length is less than 10, we enumerate all possible subsequences whose length is less than 6 words. Since a set of such subsequences include many noises as relation expressions, we use only subsequences the frequency of which is higher than 100.

This shallow approach can be run very fast, thanks to the advances of sequential pattern mining (Pei et al., 2004). Although the method is similar to that used in Bollegala et al. (2010) , we do not use any further constraints based on part-of-speech tags, lexical-syntactic information, etc. Our contention is that such ad-hoc constraints unnecessarily restrict a set of relation expressions. Our method treats ambiguous expressions (e.g. "of", "in", "with", etc.) as relation expressions. Instead, the effectiveness or the degree of ambiguities of a relation expression is captured in the common space after embedding.

[1] http://nlp.stanford.edu/software/corenlp.shtml

The second method is based on dependency parsing. We obtain the dependency tree of a sentence by a publicly available deep parser, Enju3[2] (Miyao and Tsujii, 2005; Miyao and Tsujii, 2008), and then extract shortest paths between two entities. Unlike the first method, this method uses linguistic information to extract the skeleton of a relation expression.

Each node in shortest paths consists of a base form (e.g., "like", "player"), syntactic category (e.g., "verb", "noun"), and predicate-argument links. The length of shortest paths was restricted to the range from 1 to 6. Compared with the first method, a set of shortest paths contains much less noises, so that we do not filter out those with low frequency. In the same way as the first method, a set of shortest paths contains highly ambiguous paths (e.g. the path of "of").

4.1.4 Generation of the Space for Entity-Pairs

The primal semantic space for entity pairs can be constructed in several ways. The co-training method constructed a space of entity pairs based on their co-occurrences with relation expressions. Their method requires the two spaces of entity pairs and relation expressions have to be tightly coupled.

On the other hand, our approach allows us to design the two spaces independently. In addition to the tightly coupled spaces, we design a new space for entity pairs based on the distributional hypothesis (Harris, 1954). We used the point-wise mutual information (PMI) score of each word with an entity-pair. PMI score is defined as $\mathrm{PMI} = \log_e p(w_a|\langle e_i, e_j\rangle)/p(w_a)$ where $p(w_a)$ is an occurrence probability of a word w_a and $p(w_a|\langle e_i, e_j\rangle)$ is a conditional probability with respect to an entity-pair $\langle e_i, e_j\rangle$. We filtered words whose PMI scores were below 1.0 and all the rest were used as the features.

To maximize the effectiveness of the space, we performed preliminary experiments by changing parameters in the definition of context in the distributional hypothesis, such as how the context around entities is distinguished, whether the whole of a sentence or limited windows around entities are used as context, etc. As a result, we chose the settings in which right, left, and intervening contexts are distinguished. We used three different window sizes as the context (e.g. 4, 5 and 6 words). That is, when we set the window size to 4, we used the four words in the left side of the first entity as the left context, those in the right side of the second entity as the right context, and the words in the intervening part as the intervening context. If the intervening part consists of more than 8 words, the four words in the neighborhood of the two entities are used as the intervening context.

4.1.5 Generation of the Space for Relation Expressions

Following the work (Lin and Pantel, 2001), we constructed a simple space, in which a relation expression is characterized by the entities which it links. We counted the entities in the left-hand side and the right hand side of a relation expression. The same as the vector of an entity-pair, we used the PMI score as the feature value. As for feature selection, we chose the entities whose PMI scores are no less than 1.0[3].

4.1.6 Dimension Reduction

After generating vectors for entity-pairs and relation expressions, we applied a dimension reduction. Since both of the primary semantic spaces use surface words or entities, their vectors tend to have a very large dimension (i.e. about $100,000$ for entity pairs and about $2,500$ for relation expressions). Since the cardinalities of the two sets of distinct entity pairs and relations expressions are also very high (See Table 1 of the specification of the ENT dataset), the high dimensions of the two spaces would make the computation cost of MVPLS embedding in terms of time and space prohibitively high.

To take advantage of the sparseness of both spaces, we used Randomized SVD (Halko et al., 2011) which can produce low-dimensional feature vectors from a large-scale sparse feature matrix efficiently. We produced spaces with $3,000$-dimensions for entity-pairs and $1,000$ for relation expressions.

4.1.7 Common Space Embedding

Lastly, we applied MVPLS (1) to construct common space projection matrices. We set the dimension of common space as $1,000$. We verified that the dimension does not affect much the evaluation results,

[2] `http://www.nactem.ac.uk/enju/`

[3] Other than context-based characterization methods, we have applied path kernel method (Reichartz et al., 2009; Reichartz et al., 2010) to shortest path relations as preliminary works, however, their performances were definitely worse.

Window Size	4	5	6
VSM (Turney, 2005)		0.68	
LRA (Turney, 2005)		0.68	
(Bollegala et al., 2010)		0.76	
Relation $(1,000)$		0.82	
Original $(1,000)$	0.88	0.88	0.88
Embedded $(1,000)$	**0.90**	**0.89**	**0.90**

Table 2: Entity-Pair space evaluation results (Enumeration) : Each figure shows the average precision. The best figures in each window size are written in **bold**. Figures in parentheses denote the number of dimensions.

Window Size	4	5	6
VSM (Turney, 2005)		0.68	
LRA (Turney, 2005)		0.68	
(Bollegala et al., 2010)		0.76	
Relation $(1,000)$		0.62	
Original $(1,000)$	**0.91**	0.90	**0.91**
Embedded $(1,000)$	**0.91**	**0.91**	**0.91**

Table 3: Entity-Pair space evaluation results (Shortest Path). Each figure shows the average precision. The best figures in each window size are written in **bold**. Figures in parentheses denote the number of dimensions.

when we set it to larger than 300. So we used a common space with $1,000$ dimensions for the sake of comparison with the original spaces.

4.2 Relation Mining Evaluation

We evaluated the embedding approach by a quantitative analysis on the relation mining task used in (Bollegala et al., 2010). The experimental setting is the same as the previous work. The objective is to assess whether the derived common semantic space provides a good space for measuring semantic distances among entity-pairs. We expected that in a good semantic space, entity-pairs which belong to the same semantic category would be clustered in proximity.

We used the ENT dataset (Bollegala et al., 2009). We used the same evaluation measures used in (Bollegala et al., 2010). The measure assumes that a semantic space would be judged as appropriate if it assigned higher similarity scores to entity-pairs the relationships of which belong to the same category. Therefore, the measure evaluated the top 10 similar pairs to each entity-pair and calculated average precision defined as $\sum_{t=1}^{10} \mathrm{Rel}(t) \cdot \mathrm{Pre}(t)/10$. Here, $\mathrm{Rel}(t)$ is a binary valued function that returns 1 if the entity-pair at rank t and $\langle e_i, e_j \rangle$ have the same semantic category. $\mathrm{Pre}(t)$ is the precision at rank t, which is defined by the percentage of correct objects in top t pairs.

For the sake of comparison, we prepared several models, which used different semantic spaces for entity pairs. One space (called Relation) is to characterize an entity pair by the relation expressions which it co-occur. Another space (called Original) is to characterize an entity pair by the context vector discussed in Section 4.1.4. There are three Original spaces which use different window sizes (4, 5 and 6 words). Then, the final space is the common space obtained by embedding (called Embedded).

Table 2 and 3 correspond to the experiment results using the two definitions of relation expressions, one by enumerated word sequences and the other by shortest paths. We note that the previous works only use co-occurrences information and cannot use any context information. The previous work and Relation have no ways of changing the size of windows. Therefore, these results are independent of the window size. These tables show the limitation of co-training which can only use tightly coupled vector spaces for entity pairs and relation expressions. Both the original and the common embedded space outperform significantly the performance obtained by previous works, regardless of the definitions of relation expressions (i.e. enumerated subsequence and shortest paths). Since the space for relation expressions is simple and poor, we expected that it would hardly add extra information to the space of entity pairs. However, the common space embedded from the two spaces improve the performance.

4.3 Relation Expression Mining

While the primary space for relation expressions is rather poor, vector representations of relation expressions are much richer in the common space. This is because they receive extra information from the rich space of entity-pairs through their co-occurrences. For evaluation, we first chose representative relation expressions, and then gathered relations that are close to them in the primary space of relation

{announce acquisition}		{president ,}	
Embedded	Original	Embedded	Original
{announce that have acquire}	{announce that have acquire}	{chairman ,}	{'s president be}
{complete acquisition}	{acquire}	{, ceo &}	{would say}
{say have it buy}	{pay}	{'s president ,}	{would that say}
{acquire}	{buy}	{ceo &}	{'s blue and}
{pay}	{compra}	{chief ,}	{'s chairman ,}
{'s acquisition}	{buy company}	{, ceo)}	{chairman ,}
{'s out_of}	{say that it buy}	{chief ,}	{palmisano}
{'s purchase}	{nor}	{would that say}	{,}
{acquisition}	{do}	{executive ,}	{, reader ,}
{'s takeover}	{announce be buy}	{ceo become}	{palmisano include door '}

Table 4: Evaluation of similarity measure between relation expressions. This table shows the top-10 ranked relation expressions that are closest to two representative relation expressions.

expressions and in the common space. If our expectation was correct, the list of expressions close to the chosen expression in the common space should be more appropriate than that in the primary space.

We show the result of the experiment in which we use shortest paths as relation expressions. We used the same dataset as the previous experiment. We removed shortest paths with frequency less than 10. As for the primary space for entity pairs, we use the one with the window size of 5. We used {announce acquisition} and {president ,} as two representatives.

Table 4 shows the lists of relation expressions closest to the chosen representatives in the common space and the primary space. For the ease of interpretation, we do not show syntactic categories and predicates attached to the shortest paths. One can easily see that the common space successfully moved down many ambiguous expressions such as {compra} and {nor} in {announce acquisition}, and {would say} and{,} in {president ,}. On the other hand, some relation expressions which are specific and semantically similar to the chosen ones moved up in the rank, for example {'s purchase} and {chief ,}.

We have also conducted the same experiment for relation expressions produced by the enumeration method. While the enumeration method improves the relation mining which gathering similar entity-pairs, it gave much poorer results to expressing mining than the shortest paths. This is because the enumeration method generated a large amount of non-meaningful relation expressions. For example, to generate a complex relation expression such as {say have it buy} appeared in Table 4, the enumeration method has to generate a large variety of noisy ones that co-occur with a complex relation expression.

4.4 Similarity measure between entity-pair and relation expressions

The major advantage of embedding over co-training is that it produces where the two different types of objects, entity-pairs and relation expressions, are treated in the exactly the same vector space. Therefore, we can easily gather a set of relation expressions relevant to a given prototype entity pair of a relation. In this experiment, instead of representative relation expressions, we gave entity pairs which are prototypical examples of certain relations. As in the previous experiment, we used the shortest paths as relation expressions, and ignored relation expressions with frequency under 10.

Table 5 shows the list of relation expressions for two prototypical entity-pairs used in the ENT dataset, ⟨charlie chaplin, london⟩ as a representative entity-pair for BIRTHPLACE and ⟨facebook inc, mark zuckerberg⟩ as CEO relation semantics. The table shows that the top-10 frequently co-occurring relation expressions. While many noisy relation expressions (i.e. ambiguous expressions) appear by extracting expressions based on their co-occurrence frequency with ⟨charlie chaplin, london⟩, these ambiguous expressions disappear in the proximity of the entity-pair in the common space. Moreover, the result of ⟨facebook inc. mark zuckerberg⟩ shows that some relation expressions that do not co-occur with the prototype entity-pair were successfully extracted, such as {'s executive ,}.

5 Related Work

Bollegala et al. (2010) proposed a simple sequential co-clustering framework of entity-pairs and relation expressions for objects sharing the same semantic relation to be clustered. Our definition of primal-dual

⟨charlie chaplin, london⟩		⟨facebook inc, mark zuckerberg⟩	
Embedded	Co-occurrence	Embedded	Co-occurrence
{bear walworth}	{bear}	{'s executive ,}	{, ceo}
{bear april}	{'s " arrangement while lay orchestra}	{ceo be}	{, ceo (}
{play}	{,}	{ceo}	{founder and}
{bear}	{reception}	{,}	{everything , ceo}
{bear}	{'s}	{'s president ,}	{andceo}
{bear woolsthorpe ,}	{'s}	{have say}	{ceo ,}
{bear woolthrope}	{be when}	{, ceo ,}	{-}
{be member parliament}	{and}	{, ceo}	N/A
{bear woolsthorpe}	{bear april street , walworth ,}	{ceo become}	N/A
{'s}	{walk ,}	{buy}	N/A

Table 5: Relation expressions gathered by prototype entity-pairs on the ENT dataset. This table shows the top-10 ranked relation expressions that are closest to the representative entity-pairs ⟨charlie chaplin, london⟩ as BIRTHPLACE and ⟨facebook inc, mark zuckerberg⟩ as CEO. ⟨facebook inc, mark zuckerberg⟩ co-occurred with only seven discrete relation expressions.

semantic space and common space embedding approach can be viewed as extensions of their work by introducing feature spaces as characterizations. This extension enables to utilize each space's characterizations and calculate similarity between different types of objects. Baroni and Lenci (2010) proposed a framework that analyze triplets as a third-order tensor, called "distributional memory". By matricizing the tensor to second-order tensors, that is matrices, this framework can utilize the relationship between entity-pairs and relation expressions. They also propose the procedure for generating continuous vector representations of entities and relation expressions through the tensor decomposition techniques. However, this framework cannot use semantic spaces independently defined, therefore it is difficult to incorporate the similarity information between entity-pairs or similarities between relation expressions into the decomposition procedure in contract to our framework based on MVPLS. Lin and Pantel (2001) proposed a weakly supervised framework of mining paraphrases based on shortest paths as basic units to be mined. Our work can be viewed as an extension by mixing entity-pair characterizations with the extended distributional hypothesis by embedding.

Many other previous work have been proposed to construct a knowledge base, including relation expressions (Carlson et al., 2010; Fader et al., 2011; Nakashole et al., 2012). However, they cannot interactively predict semantic meanings of objects through labeled objects of the other space.

As for treatment of ambiguity, some previous work has focused on triplet clustering to disambiguate each triplet object known as relation extraction. Unlike other mining tasks, this task requires a system to disambiguate the meaning of a relation expression r in $\langle r, e_1, e_2 \rangle$ which appears in a specific context. We did not treat this task in this paper, however, our framework would discharge the burden by showing the insight of ambiguities of each relation expression and entity-pair. Yao et al. (2011; 2012) proposed a new triplet clustering method through a generative probabilistic model. The model used surrounding contexts as features in both a sentence and document level to identify the meaning of each triplet. They demonstrated the effectiveness of their models compared with USP (Poon and Domingos, 2009) or DIRT (Lin and Pantel, 2001). Min et al. (2012) provided a simple and scalable triplet clustering algorithm in an unsupervised way and enables to incorporate various resources about entity and relation expressions. Chen et al. (2006) proposed a label propagation algorithm for relation extraction as a semi-supervised learning method by utilizing the information of parsing.

6 Conclusion

We propose a common space embedding framework which constructs a semantic space in which both entity-pairs and relation expressions are represented. We showed that our framework is effective to construct the extension set and the manifestation set of a relation R in this space. The results of experiments showed that the common space is further refined for tasks such as relation and relation expression mining, compared with the original two spaces. Moreover, we showed relation expressions collected from a small set of entity-pairs through the common space, which share the same semantics as being relevant.

There are several interesting future topics:

- how to iteratively collect objects from a dual object, like bootstrapping
- how to reduce surface diversities of relation expressions which are not abstracted away by simple method or shortest paths (by using methods such as SOL Pattern Model (Nakashole et al., 2012))
- How to combine a ground truth and non-textual knowledge stored in knowledge bases for characterizing entity-pairs with our framework
- How to extend the framework in order to deal with n-ary relations

References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

Danushka T. Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2009. Measuring the similarity between implicit semantic relations from the web. In *Proc. of WWW*.

Danushka T. Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2010. Relational duality: unsupervised extraction of semantic relations between entities on the web. In *Proc. of WWW*.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proc. of AAAI*.

Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. 2006. Relation extraction using label propagation based semi-supervised learning. In *Proc. of ACL*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proc. of EMNLP*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of ACL*.

Nathan Halko, Per G. Martinsson, and Joel A. Tropp. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Dekang Lin and Patrick Pantel. 2001. Dirt - discovery of inference rules from text. In *Proc. of KDD*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

Bonan Min, Shuming Shi, Ralph Grishman, and Chin-Yew Lin. 2012. Ensemble semantics for large-scale unsupervised relation extraction. In *Proc. of EMNLP-CoNLL*.

Yusuke Miyao and Jun'ichi Tsujii. 2005. Probabilistic disambiguation models for wide-coverage hpsg parsing. In *Proc. of ACL*.

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature forest models for probabilistic hpsg parsing. *Computational Linguistics*, 34(1):35–80.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. Patty: A taxonomy of relational patterns with semantic types. In *Proc. of EMNLP-CoNLL*.

Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Jianyong Wang, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. 2004. Mining sequential patterns by pattern-growth: The prefixspan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1424–1440.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proc. of EMNLP*.

Frank Reichartz, Hannes Korte, and Gerhard Paass. 2009. Dependency tree kernels for relation extraction from natural language text. In *Proc. of ECML/PKDD (2)*.

Frank Reichartz, Hannes Korte, and Gerhard Paass. 2010. Semantic relation extraction with kernels over typed dependency trees. In *Proc. of KDD*.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP-CoNLL*, pages 1201–1211.

Peter D. Turney and Michael L. Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4):315–346.

Peter D. Turney, Michael L. Littman, Jeffrey Bigham, and Victor Shnayder. 2003. Combining independent modules to solve multiple-choice synonym and analogy problems. In *RANLP*, pages 482–489.

Peter D. Turney. 2005. Measuring semantic similarity by latent relational analysis. In *IJCAI*, pages 1136–1141.

Wei Wu, Hang Li, and Jun Xu. 2013. Learning query and document similarities from click-through bipartite graph with metadata. In *Proc. of WSDM*.

Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proc. of EMNLP*.

Limin Yao, Sebastian Riedel, and Andrew McCallum. 2012. Unsupervised relation discovery with sense disambiguation. In *Proc. of ACL*.

An Enhanced Lesk Word Sense Disambiguation Algorithm through a Distributional Semantic Model

Pierpaolo Basile **Annalina Caputo** **Giovanni Semeraro**
Department of Computer Science, University of Bari Aldo Moro
Via E. Orabona, 4, Bari - 70125 Italy
{pierpaolo.basile,annalina.caputo,giovanni.semeraro}@uniba.it

Abstract

This paper describes a new Word Sense Disambiguation (WSD) algorithm which extends two well-known variations of the Lesk WSD method. Given a word and its context, Lesk algorithm exploits the idea of maximum number of shared words (maximum overlaps) between the context of a word and each definition of its senses (gloss) in order to select the proper meaning. The main contribution of our approach relies on the use of a word similarity function defined on a distributional semantic space to compute the gloss-context overlap. As sense inventory we adopt BabelNet, a large multilingual semantic network built exploiting both WordNet and Wikipedia. Besides linguistic knowledge, BabelNet also represents encyclopedic concepts coming from Wikipedia. The evaluation performed on SemEval-2013 Multilingual Word Sense Disambiguation shows that our algorithm goes beyond the most frequent sense baseline and the simplified version of the Lesk algorithm. Moreover, when compared with the other participants in SemEval-2013 task, our approach is able to outperform the best system for English.

1 Introduction

Unsupervised Word Sense Disambiguation (WSD) algorithms aim at resolving word ambiguity without the use of annotated corpora. Among these, two categories of knowledge-based algorithms gained popularity: overlap- and graph-based methods. The former owes its success to the simple intuition underlying that family of algorithms, while the diffusion of the latter started growing after the development of semantic networks.

The overlap-based algorithms stem from the Lesk (1986) one, which inspired a whole family of methods that exploit the number of common words in two sense definitions (*glosses*) to select the proper meaning in a context. Glosses play a key role in Lesk algorithm, which exploits only two types of information: 1) the set of dictionary entries, one for each possible word meaning, and 2) the information about the context in which the word occurs. The idea is simple: given two words, the algorithm selects those senses whose definitions have the maximum overlap, i.e. the highest number of common words in the definition of the senses. In order to extract definitions, Lesk adopted the *Oxford Advanced Learner's* dictionary. This approach suffers from two problems: 1) complexity, the number of comparisons increases combinatorially with the number of words in a text; and 2) definition expressiveness, the overlap is based only on word co-occurrences in glosses. The first problem was tackled by a "simplified" version of Lesk algorithm (Kilgarriff and Rosenzweig, 2000), which disambiguated each word separately. Given a word, the meaning whose gloss shows the maximum overlap with the current context, represented by the surrounding words, is selected. The simplified Lesk significantly outperforms the original Lesk algorithm as proved by Vasilescu et al. (2004). The second problem was faced by Banerjee and Pedersen (2002), who proposed an "adapted" Lesk algorithm. The adapted variation exploits relationships among meanings: each gloss is extended by the definitions of semantically related meanings. Banerjee and Pedersen adopt WordNet as semantic network and their algorithm takes into account several relations:

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1591–1600, Dublin, Ireland, August 23-29 2014.

hypernym, hyponym, holonym, meronym, troponym and attribute relation. The adapted algorithm outperforms the Lesk one in disambiguating nouns in SensEval-2 English task. Despite these improvements, overlap-based algorithms failed to stand the pace with figures achieved by graph approaches. Their ability to disambiguate all words in a sequence at once, meanwhile exploiting the existing interconnections (edges) between senses (nodes), has made these algorithms more principled than methods that use the sense definition overlaps. Moreover, the success of PageRank in web search has inspired a new vein of algorithms for sense disambiguation that blossomed during the past years. Although graph-based algorithms have taken advantage of the rich set of relationships available in dictionaries like WordNet, they completely neglected the role of glosses in the disambiguation process.

From our standpoint, glosses are an important piece of information since they extensionally define a word meaning. In this paper we propose a revised version of the simplified and adapted Lesk variations that overcomes limits due to the definition expressiveness. Our method replaces the concept of overlap with that of similarity. Similarity is computed on a Distributional Semantic Space (DSS) in order to account for semantic relationships between words occurring in the definition and context, for as they emerge from the use in the language. Indeed, Distributional Semantics Models (DSM) exploit the geometric metaphor of meanings, which are represented through points into a space where distance is a measure of semantic similarity. The point representation inherits information about all co-occurring context words, and then it is suitable for computing the overlap where no exact word matching can occur. In addition, we introduce two functions: the former assigns an inverse gloss frequency (IGF) score to each term occurring in the extended gloss, the latter exploits information about sense frequencies extracted from an annotated corpus.

We choose BabelNet (Navigli and Ponzetto, 2012) as sense inventory. BabelNet is a very large semantic network built up exploiting both WordNet and Wikipedia. Besides linguistic knowledge, it also represents encyclopedic concepts coming from Wikipedia and information about named entities. This makes our approach inherently multilingual and suitable for tasks such as named entity disambiguation. The evaluation on SemEval-2013 Multilingual Word Sense Disambiguation (Navigli et al., 2013) proves that our method is able to outperform both baselines (simplified Lesk and most frequent sense) and, for English language, also the best SemEval-2013 participant.

The paper is structured as follows. Section 2 provides a brief introduction to Distributional Semantic Models, while Section 3 describes the proposed methodology. Evaluation and details about the algorithm implementation are reported in Section 4, while related work is described in Section 5. Finally, conclusions close the paper.

2 Distributional Semantic Models

Semantic (or *Word*) *Spaces* are geometrical spaces of words where vectors express concepts, and their proximity is a measure of the semantic relatedness. One of their greatest virtues is that they can be built using entirely unsupervised analysis of free text. Moreover, they make few language-specific assumptions since only tokenized text is needed. *WordSpace*s are inspired by the *distributional hypothesis* (Harris, 1968) whereby the meaning of a word is determined by the rules of its use in the context of ordinary and concrete language behaviour. This means that words are semantically similar if they share *contexts* (surrounding words). Building a *WordSpace* involves the definition of a distributional model, that is a quadruple (Lowe, 2001) consisting of: the space basis (word vectors) and dimension; the function that takes into account word co-occurrences and how these are represented in the final vector; a similarity function defined over vectors; and eventually a map that transforms the vector space.

Our idea is to apply DSMs to WSD for computing the overlap between the gloss of the meaning and the context as a similarity measure between their corresponding vector representations in a *SemanticSpace*. In this paper we build a *SemanticSpace* (co-occurrences matrix M) by analysing the distribution of words in a large corpus, then M is reduced using Latent Semantic Analysis (LSA) (Landauer and Dumais, 1997). LSA collects the text data in a co-occurrence matrix, which is then decomposed into smaller matrices with Singular-Value Decomposition (SVD). Hence, LSA represents high-dimensional vectors in a lower-dimensional space while capturing latent semantic structures in the text data.

Given the vector representation of two words, DSMs usually compute their similarity as the cosine of the angle between them. In our case, the gloss and the context are composed by several terms, so in order to compute their similarity we need a method to compose the words occurring in these sentences. It is possible to combine words through vector addition (+) that corresponds to the point-wise sum of vector components. For each set of terms, phrase or sentence, we build its vector representation by adding the vectors associated to the words it is composed of. Then, the similarity measure is computed as the cosine similarity between the two phrases/sentences. More formally, if $g = g_1 g_2 ... g_n$ and $c = c_1 c_2 ... c_m$ are the gloss and the context respectively, we build their vector representation $\mathbf{g}$ and $\mathbf{c}$ in the *SemanticSpace* through addition of word vectors belonging to them:

$$\begin{aligned} \mathbf{g} &= \mathbf{g_1} + \mathbf{g_2} + \ldots + \mathbf{g_n} \\ \mathbf{c} &= \mathbf{c_1} + \mathbf{c_2} \ldots + \mathbf{c_m} \end{aligned} \tag{1}$$

The cosine similarity between $\mathbf{g}$ and $\mathbf{c}$ is a measure of the similarity of the two sentences that we consider as a score associated to the candidate meaning with respect to the context.

3 Methodology

At the heart of our approach there is the simplified Lesk algorithm. Given a text $w_1 w_2 ... w_n$ of n words, we disambiguate one at a time taking into account the similarity between the gloss associated to each sense of the target word w_i and the context. The meaning whose gloss has the highest similarity is selected. The context could be represented by a subset of surrounding words or the whole text where the word occurs. Moreover, taking into account the idea of the Banerjee's adaptation, we expand each gloss with those of related meanings.

Our sense inventory is BabelNet, a very large multilingual semantic network built relying on both WordNet and Wikipedia. In BabelNet linguistic knowledge is enriched with encyclopedic concepts coming from Wikipedia. WordNet synsets and Wikipedia concepts (pages) are connected in an automatic way. We choose BabelNet for three reasons: 1) glosses are richer and contain text from Wikipedia, 2) it is multilingual, thus the proposed algorithm can be applied to several languages, and 3) it also contains information about named entities, thus an algorithm using BabelNet could be potentially used to disambiguate entities.

Our algorithm consists of five steps:

1. **Look-up**. For each word w_i, the set of possible word meanings is retrieved from BabelNet. First, we look for senses coming from WordNet (or WordNet translated into languages different from English). If no sense is found, we retrieve senses from Wikipedia. We adopt this strategy because mixing up all senses from Wikipedia and WordNet results in worse performance. Conversely, if a word does not occur in WordNet it is probably a named entity, thus Wikipedia could provide useful information to disambiguate it.

2. **Building the context**. The context C is represented by the l words to the left and to the right of w_i. We also adopt a particular configuration in which the context is represented by all the words that occur in the text.

3. **Gloss expansion**. We indicate with s_{ij} the j-th sense associated to the target word w_i. We expand the gloss g_{ij} that describes the j-th sense using the function "getRelatedMap" provided by BabelNet API. This method returns all the meanings related to a particular sense. For each related meaning, we retrieve its gloss and concatenate it to the original gloss g_{ij} of s_{ij}. During this step we remove glosses belonging to synsets related by the "antonym" relationship. The result of this step is an extended gloss denoted by g^*_{ij}. In order to give more importance to terms occurring in the original gloss, the words in the expanded gloss are weighed taking into account both the distance between s_{ij} and the related synsets and the word frequencies. More details about term scoring are reported in Subsection 3.2.

4. **Building semantic vectors**. Exploiting the DSM described in Section 2, we build the vector representation for each gloss g^*_{ij} associated with the senses of w_i and the context C.

5. **Selecting the correct meaning**. For each gloss g^*_{ij}, the algorithm computes the cosine similarity between its vector representation and context vector C. The similarity is linearly combined with the probability $p(s_{ij}|w_i)$ that takes into account the sense distribution of s_{ij} given the word w_i; details are reported in Subsection 3.1. The sense with the highest similarity is chosen.

In order to compare our approach to the simplified Lesk algorithm, we developed a variation of our method in which, rather than building the semantic vectors, we count the common words between each extended gloss g^*_{ij} and the context C. In this case, we apply stemming to maximize the overlap.

3.1 Sense Distribution

The selection of the correct meaning takes also into account the senses distribution of the word w_i. We retrieve information about sense occurrences from WordNet (Fellbaum, 1998), which reports for each word w_i its sense inventory S_i with the number of times that the word w_i was tagged with s_{ij} in SemCor. SemCor is a collection of 352 documents manually annotated with WordNet synsets. We introduce the sense distribution factor in order to consider the probability that a word w_i can be tagged with the sense s_{ij}. Moreover, since some synsets do not occur in SemCor and can cause zero probabilities, we adopt an additive smoothing (also called Laplace smoothing). Finally the probability is computed as follow:

$$p(s_{ij}|w_i) = \frac{t(w_i, s_{ij}) + 1}{\#w_i + |S_i|} \tag{2}$$

where $t(w_i, s_{ij})$ is the number of times the word w_i is tagged with s_{ij} and $\#w_i$ is the number of occurrences of w_i in SemCor.

3.2 Gloss Term Scoring

The extended gloss conflates words from the gloss directly associated with the synset s_{ij} with those of the glosses appearing in the related synsets. When we add words to the extended gloss, we weigh them by a factor inversely proportional to the distance in the graph (number of edges) between s_{ij} and the related synsets so to reflect their different origin. Let d be that distance, then the weight is computed as $\frac{1}{1+d}$. Finally, we re-weigh words using a strategy similar to the inverse document frequency (IDF) that we call inverse gloss frequency (IGF). The idea is that if a word occurs in all the extended glosses associated with a word, then it poorly characterizes the meaning description. Let gf^*_k be the number of extended glosses that contain a word w_k, then IGF is computed as follow:

$$IGF_k = 1 + log_2 \frac{|S_i|}{gf^*_k} \tag{3}$$

This approach is similar to the idea proposed by Vasilescu et al. (2004), where TF-IDF of terms is computed taking into account the glosses in the whole WordNet, while we compute IGF considering only the glosses associated to each word. Finally, the weight for the word w_k appearing h times in the extended gloss g^*_{ij} is given by:

$$weight(w_k, g^*_{ij}) = h \times IGF_k \times \frac{1}{1+d} \tag{4}$$

4 Evaluation

The evaluation is performed using the dataset provided by the organizers of the Task-12 of SemEval-2013. The task concerns Multilingual Word Sense Disambiguation, a traditional WSD "all-words" experiment in which systems are expected to assign the correct BabelNet synset to all occurrences of noun phrases (which refer to both disambiguated nouns and named entities) within arbitrary texts in different languages.

The goal of our evaluation is twofold: to prove that our strategy outperforms the simplified Lesk approach, and then to compare our system with respect to the other task participants.

In both the experiments we consider two languages, English and Italian, to evaluate performance in a multilingual setting. It is important to underline that in our approach only stemming and the corpus used to build the distributional model are language dependent.

4.1 System Setup

Our method[1] is completely developed in JAVA using BabelNet API 1.1.1 provided by the authors[2]. We adopt the standard Lucene analyzer to tokenize both glosses and the context, while Snowball library[3] is used for stemming in several European languages. The *SemanticSpace*s for the two languages are built using proprietary code relying on two Lucene indexes, which contain documents from British National Corpus (BNC) for English, and from Wikipedia dump for Italian, respectively. For each language, the co-occurrences matrix M contains information about the top 100,000 most frequent words in the corpus. M is reduced by LSA using the SVDLIBC tool[4] and setting a reduced dimension equal to 200. The result of the SVD decomposition is stored in a binary format used by our algorithm implementation.

It is important to underline that BabelNet Italian glosses are taken from MultiWordNet, which does not contain glosses for all the synsets. Then, we replace each missing gloss by the words that belong to the synset.

We evaluate our system by setting two parameters: 1) the context size (3, 5, 10, 20 and the whole text); 2) the use of information about sense distribution (see Formula (2) in Subsection 3.1).

The gloss term scoring function is always applied, since it provides better results. The synset distance d used to expand the gloss is fixed to 1; we experimented with a distance d set to 2 without any improvement. The sense distribution is linearly combined with the cosine similarity score through a coefficient set to 0.5.

Some notes about sense frequency: by using only sense distribution to select a sense we obtain an algorithm that performs like the most frequent sense (MFS). In other words, the algorithm always assigns the most probable meaning. It is well known that this approach obtains very good performance and it is hard to be outperformed especially by unsupervised approaches.

4.2 English Evaluation

Table 1 reports results of our algorithm (DSM) compared with the best simplified Lesk approaches (LESK) in terms of precision (P), recall (R), F-measure (F) and attempt (A). Attempt is the percentage of words disambiguated by the algorithm. $SenseDistr.$ column reports the information about when the sense distribution formula (see Subsection 3.1) is used (Y) or not (N); it is also important to point out that MSF produces the same results of using only sense distribution i.e. the first sense is the most likely one. We have experimented different context sizes also for the Lesk algorithm, although for the sake of readability we report only the best Lesk with and without sense distribution.

All our runs always obtain an attempt of 100%; thus the precision and recall values are always the same. The run **EN.DSM.10** obtains the best result using both sense distribution information and the whole text (W) as context. Another important outcome is the result obtained by the run **EN.DSM.5** that, without information about sense distribution, is able to overcome the MFS baseline. To the best of our knowledge, this is the first completely unsupervised system able to overcome the MFS baseline without using sense frequency. Both results (**EN.DSM.10** and **EN.DSM.5**) suggest that the vector representation of the whole text helps the system to achieve the best performance.

Considering the Lesk method, generally, the best size for the context is 3, then a larger set of surrounding words results in worse performance, differently to what happens in the distributional approach. This is probably due to the fact that words distant from the target one match some incorrect glosses. It is important to note that no simplified Lesk run is able to overcome the MFS baseline.

[1]Available on line: `https://github.com/pippokill/lesk-wsd-dsm`
[2]Available on line: `http://lcl.uniroma1.it/babelnet/download.jsp`
[3]Available on line: `http://snowball.tartarus.org/`
[4]Available on line: `http://tedlab.mit.edu/\textasciitildedr/SVDLIBC/`

Run	*ContextSize*	*SenseDistr.*	*P*	*R*	*F*	*A*
MFS	-	-	0.656	0.656	0.656	100%
EN.LESK.1	3	N	0.525	0.525	0.525	100%
EN.LESK.6	3	Y	0.633	0.633	0.633	100%
EN.DSM.1	3	N	0.536	0.536	0.536	100%
EN.DSM.2	5	N	0.605	0.605	0.605	100%
EN.DSM.3	10	N	0.633	0.633	0.633	100%
EN.DSM.4	20	N	0.650	0.650	0.650	100%
EN.DSM.5	W	N	0.687	0.687	**0.687**	100%
EN.DSM.6	3	Y	0.669	0.669	0.669	100%
EN.DSM.7	5	Y	0.677	0.677	0.677	100%
EN.DSM.8	10	Y	0.689	0.689	0.689	100%
EN.DSM.9	20	Y	0.696	0.696	0.696	100%
EN.DSM.10	W	Y	0.715	0.715	**0.715**	100%

Table 1: Results of the English evaluation.

Comparing DSM-based with simplified Lesk approaches, the former consistently outperform Lesk-based algorithms when considering the use (or not) of sense distribution.

4.3 Italian Evaluation

Table 2 reports results of our algorithm for the Italian language. In this case we still obtain the best result (**IT.DSM.10**) using DSM and sense distribution. As for English, the systems without sense distribution overcome the MFS baseline. However, in this case simplified Lesk with sense distribution is able to outperform the MFS. We ascribe this different behaviour to the problem of missing glosses that we solved by adding the words in the synset.

Run	*ContextSize*	*SenseDistr.*	*P*	*R*	*F*	*A*
MFS	-	-	0.572	0.572	0.572	100%
IT.LESK.2	5	N	0.531	0.530	0.530	99.71%
IT.LESK.10	W	Y	0.608	0.606	0.607	99.71%
IT.DSM.1	3	N	0.611	0.609	0.610	99.71%
IT.DSM.2	5	N	0.608	0.607	0.607	99.71%
IT.DSM.3	10	N	0.627	0.625	0.626	99.71%
IT.DSM.4	20	N	0.629	0.627	0.628	99.71%
IT.DSM.5	W	N	0.634	0.632	**0.633**	99.71%
IT.DSM.6	3	Y	0.632	0.630	0.631	99.71%
IT.DSM.7	5	Y	0.631	0.629	0.630	99.71%
IT.DSM.8	10	Y	0.636	0.634	0.635	99.71%
IT.DSM.9	20	Y	0.640	0.638	0.639	99.71%
IT.DSM.10	W	Y	0.642	0.640	**0.641**	99.71%

Table 2: Results of the Italian evaluation.

4.4 Task Results

In this subsection, we report our best performance (Table 3) with respect to the other participants in the SemEval-2013 Task-12 on multilingual Word Sense Disambiguation, for both English (Table 3a) and Italian (Table 3b).

Regarding the English language, our best methods are able to outperform all the systems. It is important to underline that our method without knowledge about sense distribution (**EN.DSM.5**) outperforms both the MFS and all the task participants. This is a very important outcome because generally

System	F
EN.DSM.10	**0.715**
EN.DSM.5	**0.687**
UMCC-DLSI-2	0.685
UMCC-DLSI-3	0.680
UMCC-DLSI-1	0.677
MFS	*0.656*
DAEBAK	0.604
GETALP-BN-1	0.263
GETALP-BN-2	0.266

(a) English

System	F
UMCC-DLSI-2	0.658
UMCC-DLSI-1	0.657
IT.DSM.10	**0.641**
IT.DSM.5	**0.633**
DAEBAK	0.613
MFS	*0.572*
GETALP-BN-2	0.325
GETALP-BN-1	0.324

(b) Italian

Table 3: Results of our best systems with respect to the Semeval-2013 participants.

knowledge-base approaches without information about sense frequencies obtain low results. For example, the UMCC-DLSI system (Gutiérrez et al., 2013) exploits sense frequency to modify prior probability of synset nodes in the PageRank, and DAEBAK system (Manion and Sainudiin, 2013) uses MFS when it is not able to select a meaning. Our experiments show that a dictionary-based approach and the adoption of a distributional semantic model for computing the similarity are enough to obtain good results. Moreover, by adding information about sense frequencies we are able to boost our performance and obtain over 70% of F-measure.

For Italian, our systems are not able to reach the same performance as for English, although they still outperform the MFS and two task participants. We think that these results are due to the same problem observed for the Italian evaluation (Subsection 4.3), that is to say, the poor quality of glosses.

5 Related Work

WSD has been an active area of NLP whose roots stem from early work in Machine Translation. Ambiguity resolution has been pursued as a way to improve retrieval systems, and generally to get better information access. Despite its ancient roots and perceived importance, this task is still far from being resolved.

Our WSD method relies on both the Lesk algorithm and its two variants: simplified (Kilgarriff and Rosenzweig, 2000) and adapted (Banerjee and Pedersen, 2002). Several approaches have modified the Lesk algorithm to reduce is exponential complexity, like the one based on Simulated Annealing (Cowie et al., 1992). Basile et al. (2007) adopted the simplified Lesk algorithm to disambiguate adjectives and adverbs, combining it with other two methods for nouns and verbs: the combination of different approaches for each part-of-speech resulted in better performance with respect to the use of a single strategy. More recently, Schwab et al. (2013) proposed GETALP, another unsupervised WSD algorithm inspired by Lesk. Their approach computes a local similarity using the classical Lesk measure (overlap between glosses), and then the local similarity is propagated to the whole text (global similarity) using an algorithm inspired by the Ant Colony. This approach got the lowest results during the SemEval-2013 Task 12 evaluation due to a bug in the system. However, the correct implementation achieves 0.583 of F-measure for English and 0.528 for Italian.

Another problem with the Lesk-based approaches is to maximize the chances of overlap between glosses or between the gloss and the context. To solve this problem, Ponzetto and Navigli (2010) extended WordNet with Wikipedia pages (WordNet++) in order to produce a richer lexical resource, obtaining the English portion of BabelNet. The simple Lesk algorithm built over WordNet++ outperformed the WordNet-base version, although it was not successful in overtaking the MFS baseline. Our approach tries to extend glosses using related synsets and adopts distributional semantics to address the problem of data sparsity. A different perspective has been recently proposed by Wang and Hirst (2014), where the limits of the exact string matching between glosses and context are overcome by a Naive Bayes-based similarity measure.

Other unsupervised approaches rely on graph algorithms that exploit the graph generated by a semantic network where the senses are connected through semantic relations. For example, DAEBAK (Manion and Sainudiin, 2013) adopts a sub-graph of BabelNet generated taking into account the surrounding words of the target word. A measure of connectivity computed on the sub-graph is used to extract the most probable sense. The MFS is used when the algorithm is not able to choose any sense. Also Navigli and Lapata (2010) exploit the idea of graph connectivity measures for identifying the most important node (sense) in the graph. Experiments conducted on SemCor show that the Degree Centrality provides best results compared to other well known techniques, such as PageRank, HITS, Key Player Problem and Betweenness Centrality. Graph-based methods also showed their validity during the SemEval 2013 Multilingual Word Sense Disambiguation task. The best system, UMCC-DLSI (Gutiérrez et al., 2013), builds a graph using several resources: WordNet, WordNet Domains and the eXtended WordNet. Then, the best sense is selected using the PageRank algorithm where the a priori probability of senses is estimated according to the sense frequencies. This is an extension of UBK algorithm (Agirre and Soroa, 2009), the first application of personalized PageRank to the WSD problem.

On the distributional side, Brody and Lapata (2008) use distributional similarity to automatically annotate a corpus for training a supervised method. Each target word in the corpus is paired with a list of neighbours selected via distributional similarity. A neighbour is linked to a sense in WordNet and then it is used for the annotation. Differently from our approach, distributional similarity is used to automatically annotate a training corpus rather than directly disambiguate terms. Miller et al. (2012) exploit a distributional thesaurus to expand both glosses and the context, then they apply the classical word overlap adopted in the simplified Lesk. This approach is strongly related to our, although our approach directly computes the overlap in the geometric space that implements the distributional semantic model. In particular, we build a vector representation for both the gloss and the context. In recent years, other approaches have tried to solve unsupervised WSD relying on distributional information. Gliozzo et al. (2005) build a matrix taking into account words and WordNet domains. The matrix is reduced using LSA and then it is combined in a kernel exploited in a supervised approach. Martinez et al. (2008) propose a method based on topic signatures automatically constructed using the Web, while Li et al. (2010) adopt Latent Dirichlet Allocation (LDA) to compute a conditional probability between a sense and the context. In this model, a sense is represented by its paraphrases used to build the LDA model.

6 Conclusions and Future Work

In this paper we describe an unsupervised WSD approach which selects the best sense according to the distributional similarity with respect to the context. In particular, the similarity is computed representing both the gloss and the context as vectors in a geometric space generated by a distributional semantic model based on LSA. The evaluation, conducted on the Task-12 of SemEval-2013, shows promising results: our method is able to overcome both the most frequent sense baseline and, for English, also the other task participants. We provide two implementations of our approach, with and without exploiting sense frequencies. For English, both implementations outperform the SemEval-2013 participants and the MFS. Differently, for Italian both implementations do not reach the figures of the best participant, but they are able to defeat the MFS. As future work, we plan to extend our evaluation to other languages, and to investigate how to adapt our approach to a specific domain. In particular, distributional models built upon a domain corpus, and sense frequencies extracted from the same corpus, could result in a domain adaptation of our algorithm.

Acknowledgements

This work fulfils the research objectives of the projects PON 01_00850 ASK-Health (Advanced System for the interpretation and sharing of knowledge in health care) and PON 02_00563_3470993 project "VINCENTE - A Virtual collective INtelligenCe ENvironment to develop sustainable Technology Entrepreneurship ecosystems" funded by the Italian Ministry of University and Research (MIUR).

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 33–41, Athens, Greece, March. Association for Computational Linguistics.

Satanjeev Banerjee and Ted Pedersen. 2002. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 2276 of *Lecture Notes in Computer Science*, pages 136–145. Springer Berlin Heidelberg.

Pierpaolo Basile, Marco de Gemmis, Anna Lisa Gentile, Pasquale Lops, and Giovanni Semeraro. 2007. UNIBA: JIGSAW algorithm for Word Sense Disambiguation. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 398–401, Prague, Czech Republic, June. Association for Computational Linguistics.

Samuel Brody and Mirella Lapata. 2008. Good Neighbors Make Good Senses: Exploiting Distributional Similarity for Unsupervised WSD. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008) - Volume 1*, pages 65–72, Manchester, United Kingdom. The Coling 2008 Organizing Committee.

Jim Cowie, Joe Guthrie, and Louise Guthrie. 1992. Lexical Disambiguation Using Simulated Annealing. In *Proceedings of the 15th Conference on Computational Linguistics (Coling 1992) - Volume 1*, pages 359–365, Nantes, France. The COLING 1992 Organizing Committee.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Alfio Gliozzo, Claudio Giuliano, and Carlo Strapparava. 2005. Domain Kernels for Word Sense Disambiguation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 403–410, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yoan Gutiérrez, Yenier Castañeda, Andy González, Rainel Estrada, Dennys D. Piug, Jose I. Abreu, Roger Pérez, Antonio Fernández Orquín, Andrés Montoyo, Rafael Muñoz, and Franc Camara. 2013. UMCC_DLSI: Reinforcing a Ranking Algorithm with Sense Frequencies and Multidimensional Semantic Resources to solve Multilingual Word Sense Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 241–249, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Zellig Harris. 1968. *Mathematical Structures of Language*. New York: Interscience Publishers John Wiley & Sons.

Adam Kilgarriff and Joseph Rosenzweig. 2000. Framework and Results for English SENSEVAL. *Computers and the Humanities*, 34(1-2):15–48.

Thomas K Landauer and Susan T Dumais. 1997. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, 104(2):211–240.

Michael Lesk. 1986. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, SIGDOC '86, pages 24–26, New York, NY, USA. ACM.

Linlin Li, Benjamin Roth, and Caroline Sporleder. 2010. Topic Models for Word Sense Disambiguation and Token-based Idiom Detection. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1138–1147, Stroudsburg, PA, USA. Association for Computational Linguistics.

Will Lowe. 2001. Towards a Theory of Semantic Space. In Johanna T. Moore and Keith Stenning, editors, *Proceedings of the 23rd Conference of the Cognitive Science Society*, pages 576–581.

Steve L. Manion and Raazesh Sainudiin. 2013. DAEBAK!: Peripheral Diversity for Multilingual Word Sense Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 250–254, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

David Martinez, Oier Lopez de Lacalle, and Eneko Agirre. 2008. On the Use of Automatically Acquired Examples for All-nouns Word Sense Disambiguation. *Journal of Artificial Intelligence Research*, 33(1):79–107, September.

Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. 2012. Using Distributional Similarity for Lexical Expansion in Knowledge-based Word Sense Disambiguation. In *Proceedings of the 24th International Conference on Computational Linguistics (Coling 2012)*, pages 1781–1796, Mumbai, India, December. The COLING 2012 Organizing Committee.

Roberto Navigli and Mirella Lapata. 2010. An experimental study of graph connectivity for unsupervised word sense disambiguation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(4):678–692.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 Task 12: Multilingual Word Sense Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Simone Paolo Ponzetto and Roberto Navigli. 2010. Knowledge-rich Word Sense Disambiguation Rivaling Supervised Systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1522–1531, Stroudsburg, PA, USA. Association for Computational Linguistics.

Didier Schwab, Andon Tchechmedjiev, Jérôme Goulian, Mohammad Nasiruddin, Gilles Sérasset, and Hervé Blanchon. 2013. GETALP System : Propagation of a Lesk Measure through an Ant Colony Algorithm. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 232–240, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Florentina Vasilescu, Philippe Langlais, and Guy Lapalme. 2004. Evaluating Variants of the Lesk Approach for Disambiguating Words. In *Proceedings of the 4th Conference on Language Resources and Evaluation (LREC '04)*, pages 633–636.

Tong Wang and Graeme Hirst. 2014. Applying a Naive Bayes Similarity Measure to Word Sense Disambiguation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, MD, USA, June.

Word Sense Induction Using Lexical Chain based Hypergraph Model

Tao Qian[1,3], **Donghong Ji**[*1], **Mingyao Zhang**[2], **Chong Teng**[1], **and Congling Xia**[1]
(1) Computer School, Wuhan University, Wuhan, China
（2）College of Foreign Languages and Literature, Wuhan University, Wuhan, China
(3) College of Computer Science and Technology, Hubei University of Science and Technology,XianNing, China
{taoqian, dhji, myzhang, tengchong, clxia}@whu.edu.cn

Abstract

Word Sense Induction is a task of automatically finding word senses from large scale texts. It is generally considered as an unsupervised clustering problem. This paper introduces a hypergraph model in which nodes represent instances of contexts where a target word occurs and hyperedges represent higher-order semantic relatedness among instances. A lexical chain based method is used for discovering the hyperedges, and hypergraph clustering methods are used for finding word senses among the context instances. Experiments show that this model outperforms other methods in supervised evaluation and achieves comparable performance with other methods in unsupervised evaluation.

1 Introduction

Word sense induction (WSI) aims to automatically find senses of a given target word (Yarowsky, 1995) from large scale texts. Compared with existing manual word sense resources, WSI techniques use clustering algorithms to determine the possible senses for a word.

Word sense induction is generally considered as an unsupervised clustering problem. The input for the clustering algorithm is context instances of a target word, represented by word bags or co-occurrence vectors, and the output is a grouping of these instances into classes, each corresponding to an induced sense.

Traditional methods in WSI tend to adopt the vector space model, in which the context of each instance of a target word is represented as a vector of features based on frequency statistics and probability distributions, e.g., first-order or second-order vector (Schütze, 1998; Purandare and Pedersen, 2004; Cruys et al., 2011). These vectors are clustered and the resulting clusters represent the induced senses. Another family of employed approach is graph-based methods (Widdows and Dorow, 2002; Véronis, 2004; Agirre et al., 2006; Klapaftis and Manandhar, 2007; Di Marco and Navigli, 2011; Hope and Keller, 2013), which have been recently explored successfully to some extent. Graph-based methods are considering the notion of a co-occurrence graph, assuming a binary relatedness between co-occurring words.

One of the key challenges in WSI is learning the higher-order semantic relatedness among multiple context instances. Previous approaches (Klapaftis and Manandhar. 2007; Bordag, 2006) for WSI are used to construct higher-order relatedness by counting co-occurrence frequency or collocation of mutiwords, regardless of global semantic similarity.

Lexical chain (Morris and Hirst, 1991) is defined as a sequence of semantically related words in text and provides important clues about the text structure and topic. It can be viewed as a global counterpart of the measures of semantic similarity (Navigli, 2009). For example, Figure 1 gives three context instances containing Apple.

* Corresponding author E-mail: dhji@whu.edu.cn.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1601–1611, Dublin, Ireland, August 23-29 2014.

i) **Apple** designs and creates *iPod* and *iTunes*.
ii) The **Apple** support homepage is your starting point for help with **Apple** *hardware and software products*.
iii) Get detailed market's price information for the *products* of **Apple** *Inc*.

Figure 1. Context instances of Apple

Obviously, four Apples in Figure 1 all refer to the Apple Company. We can directly group three instances by the lexical chain: *iPod-iTunes-hardware and software product-Inc*. This lexical chain represents a higher-order semantic relatedness among the three instances.

In this paper, we propose a hypergraph model from a *global* perspective, in which nodes represent instances of contexts where a target word occurs and hyperedges denote higher-order semantic relatedness among instances. A lexical chain based method is used for identifying the hyperedges. This method for lexical chain extraction is a knowledge-free method based on LDA topic model (Remus and Biemann, 2013).

The remainder of this paper is structured as follows. Section 2 presents an overview of the related work. Section 3 describes our model in details. Section 4 provides a quantitative evaluation and comparison with other algorithms in the SemEval-2013 word sense induction task. Finally, section 5 draws conclusions and lays out some future research directions.

2 Related Work

2.1 Word sense induction

A number of diverse approaches to WSI have been proposed so far. Context features are often represented in a variety of forms such as co-occurrence of words within phrases (Pantel and Lin, 2002; Dorow and Widdows, 2003), parts of speeches (Purandare and Pedersen, 2004), and grammatical relations (Pantel and Lin, 2002; Dorow and Widdows, 2003). The size of the context window also varies, such as two words before and after the target word, the sentence or even larger paragraph within which contains the target word.

Most of the work in WSI is the vector space model, such as context-based vector algorithm (Schütze, 1998; Ide et al., 2001; Van de Cruys et al., 2011), substitute-based vector algorithm (Yatbaz et al., 2012; Baskaya et al., 2013). In this model, the context of each instance of a target word is represented as a vector of features based on frequency statistics or probability distributions (e.g., first-order or second-order vector). These vectors are clustered by various algorithms and the resulting clusters represent the induced senses.

Another family of employed approach is graph-based methods, which have been successfully applied in the sense induction task with some better results achieved. In this framework words are represented as nodes in the graph and vertices are drawn between the target word and its co-occurrences. The co-occurrences between words can be obtained on the basis of grammatical (Widdows and Dorow, 2002) or collocational relations (Véronis, 2004). Senses are induced by identifying highly dense sub-graphs (hubs) in the co-occurrence graph.

Klapaftis (2007) uses hypergraph model for WSI, in which co-occurrences of two or more words are represented by using weighted hyperedges. This model fully exploits the existence of collocations or terms consisting of more than two words. In fact, the method converts the sense induction problem to the clustering of the contextual words, and the result relies on local word co-occurrence frequency. Our hypergraph model is constructed from a global perspective, where the whole context instance is regarded as a node.

WSI evaluation also is an important issue in WSI tasks. Previous WSI evaluations in SemEval (Agirre and Soroa, 2007; Manandhar et al., 2010) have approached sense induction in terms of finding the single most salient sense of a target word given its context. However, as shown in Erk and McCarthy (2009), multiple senses of the target word may be perceived by readers from different angles and a graded notion of sense labeling may be considered as the most appropriate. The SemEval-2013 WSI evaluation is designed to explore the possibility of finding all perceived senses of a target word in a single context instance. Our model is evaluated and verified on the SemEval-2013 WSI task.

Algorithm 1. lexical chains extraction algorithm

Input: training set D of target word, hyper-parameters of LDA model; semantic threshold γ.
Output: lexical chain set S

```
θ,φ,Z ← LDA (D)
for each topic z
  lc =""              // lc denotes a lexical chain
  for each doc d
    for each word w in doc d
      if ( z_w = z and p(w,d|z) > γ )
        lc.add (w)
S.add (lc)
return S
```

2.2 Lexical chain extraction

The Lexical chain method is an important technique in natural language processing. A lexical chain is a sequence of semantic related words in text and provides important clues about the text structure and topic. It has formed a theoretically well-founded building block in a lot of applications, such as word sense disambiguation (Manabu and Takeo, 1994), malapropism detection and correction (Hirst and St-Onge, 1998), summarization (Barzilay et al., 1997), topic tracking (Carthy, 2004), text segmentation (Stokes et al., 2004), and others.

There are mainly two approaches for lexical chain extraction. One focuses on the use of knowledge resources like WordNet (Hirst and St-Onge, 1998) or thesauri (Morris and Hirst, 1991) as background information in order to quantify semantic relations between words. A major disadvantage of this strategy is that it relies on the resource, which has a direct impact on the quality of lexical chains. Another approach is based on statistical methods (Remus and Biemann, 2013). In this paper, we follow Remus and Biemann (2013) to automatically extract lexical chain by using LDA topic model.

3 Hypergraph model

In general, lexical chain based hypergraph model contains the following steps:

i) Automatically extracting lexical chains based on topic model;
ii) Constructing hypergraph with lexical chains;
iii) Inducing word sense by hypergraph clustering.

3.1 Lexical chain extraction

The extraction technique of lexical chains is based on LDA topic model. LDA topic model (Blei et al., 2003) is a probabilistic model of text generation designed for revealing some hidden structure in large data collections. The key idea is that each document can be represented as a probability distribution over a fixed set of topics where each topic can be represented as a probability distribution over words. We use LDA topic model for estimating the semantic closeness of lexical terms, and explore a way of utilizing LDA's topic information in constructing lexical chains automatically. In our model, document is replaced with context instance of a target word.

We adopt the idea of interpreting lexical chains as topics and placing all word tokens that share the same topic into the same chain. Lexical chains are usually extracted from the same paragraph or text, whose topic distributions are identical. However, in our experiment the context instances of a target word for WSI are derived from different articles, whose topic distributions are varied. Therefore both lexical and contextual topics are modeled. After training the LDA model, we use the information of the per-document topic distribution $\theta_d = p(z|d)$, the per-topic word distribution $\phi_w = p(w|z)$ and the sampling topic of a word z_w.

The key work lies in how to assign a word to a topic in training LDA model. Since single samples of topics per word may exhibit a large variance (Riedl and Biemann, 2012), we sample several times and use the mode (most frequently assigned) topic ID per word as the topic assignment.

d_1. WOMEN are half the population but only 15% of board members at big American firms, and 10% in Europe.

d_2. He referred the dispute to the board of directors.

d_3. The chairman of your board wears a blue shirt.

d_4. Your board should be involved in helping you sell and buy candidates.

d_5. The boy pounded the nails into the board.

d_6. A board is a flat, thin, rectangular piece of wood used for a particular purpose.

d_7. The board should have ropes attached to each end so you can loop it over your neck.

(a)

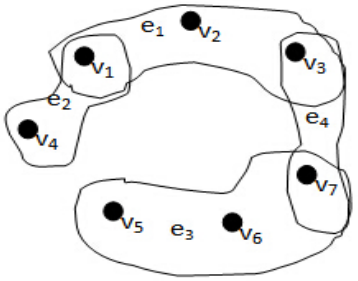
Chain 1: (members#d_1,directors#d_2,chairman#d_3)

Chain 2: (firms#d_1, candidates#d_4)

Chain 3: (nails#d_5, ropes#d_6, wood#d_7)

Chain 4: (skirt#d_3, neck#d_7)

(b)



(c)

Figure 2. An example of hypergraph creation. (a). seven context instances of **board**; (b). four lexical chains extracted; (c). the created hypergraph. The d_i in (a) corresponds to the v_i in (c) and the chain i in (b) corresponds to the e_i in (c).

The extraction algorithm is shown in algorithm 1. In order to improve the quality of identified lexical chains, a threshold γ is set to filter those invalid words whose generating probability of sampling topics in the document is lower than γ.

$$p(w,d \mid z) \approx p(z \mid d)p(w \mid z) > \gamma \qquad (1)$$

The threshold γ is essential for the quality of lexical chains, which directly impacts on the performance of the model. Detailed analysis for the threshold γ will be given in section 4.5.

3.2 Hypergraph creation

A hypergraph $H = (V, E)$ is a generalization of a graph whose edge can connect more than two vertices. Just as graphs represent many kinds of information in mathematical and computer science problems, hypergraphs also arise in important practical problems, including circuit layout, boolean satisfiability, numerical linear algebra, complex network and article co-citation, etc.

Figure 2 shows an example of hypergraph creation in our model. We represent each context instance as a vertex and connect those context instances with a lexical chain across them by a hyperedge. A hyperedge weight equals to the weight of the corresponding lexical chain, defined as follows:

$$w(e) = \frac{\sum_{w_i \in C} p(z \mid \mathrm{d}_i)p(\mathrm{w}_i \mid z)}{|C|} \qquad (2)$$

where lexical chain C corresponds to hyperedge e, $|C|$ is the number of words in C, and z is the sampling topic of C.

3.3 Hypergraph clustering

For hypergraph clustering, the hypergraph is usually transformed into induced graph. There are two transformation strategies: one is vertex expansions (2006; Zhou et al., 2006), i.e., clique expansion or star expansion, in which a hyperedge is transformed into a clique; the other is called hyperedge expansion (Pu and Faltings, 2012) based on a network flow technique, in which hyperedges are projected back to vertices through the adjacency information between hyperedges and vertices.

Hypergraph clustering algorithm can be divided into two classes: one is based on minimal normalized cut, and the other is based on maximal density. We use three general hypergraph clustering algorithms to identify the context instance clusters. The three algorithms are simply shown in figure 3 and described as the follows.

1) Normalized Hypergraph Cut (NHC)

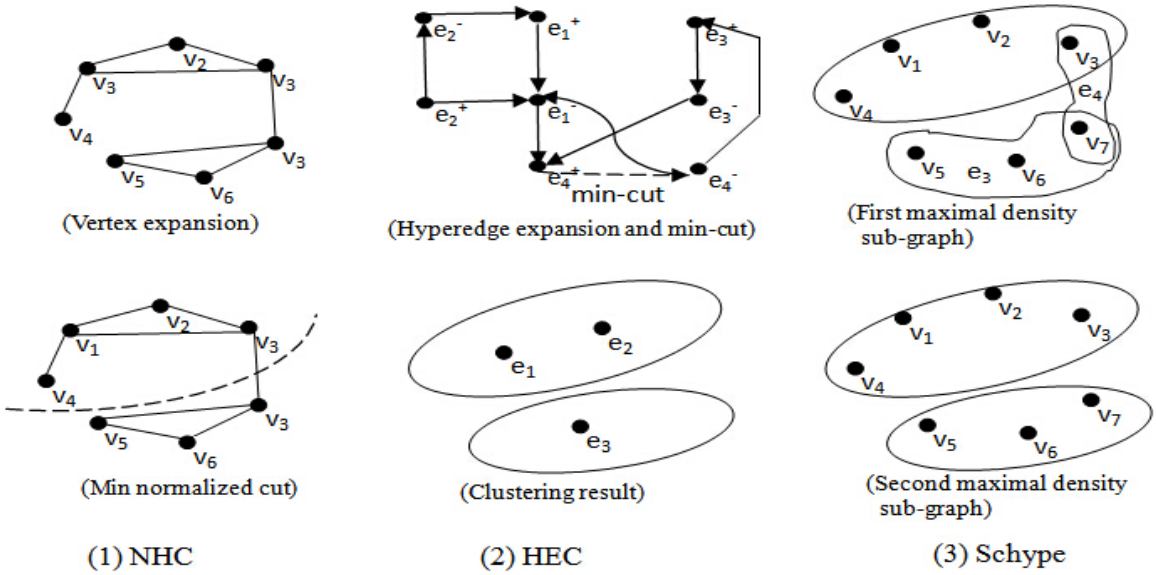


Figure 3. The clustering processes of three algorithms for the hypergraph created in figure 2.

The NHC algorithm (Zhou et al., 2006) is a typical approach based on vertex expansion. The objective is to obtain a partition in which the connection among the vertices in the same cluster is dense while the connection between two clusters is sparse. The main steps include transforming the hypergraph into an induced graph first, and then adopting the normalized Laplacian to spectral partitioning.

2) Hyperedge Expansion Clustering (HEC)

Some works (e.g., Shashua et al., 2006; Bulo and Pelillo, 2012) have shown that the pairwise affinity relations after the projection to the induced graph would introduce information loss, and working directly on the hypergraph could produce better performance.

The hyperedge expansion works as follows. It constructs a directed graph $G = (V, E)$ that includes two vertices $e+$ and $e-$ for each hyperedge e in the original hypergraph. Note that the vertices in G correspond to the hyperedges, but not the vertices in the original hypergraph. A directed edge is placed from $e+$ to $e-$ with weight $w(e)$ where w is the weighting function in the hypergraph. For every pair of overlapping hyperedges e_1 and e_2, two directed edges (e_1^-, e_2^+) and (e_2^-, e_1^+) are added to G with weights $w(e_2)$ and $w(e_1)$. After hypergraph expansion, it adopts spectral method for clustering.

3) Schype

The Schype (Michoel and Nachtergaele, 2012) is a maximal density cluster algorithm. According to the generalization of the Perron-Frobenius theorem, there exists a unique, positive vector, called the dominant eigenvector, over the set of vertices of the hypergraph, which produces a maximal density sub-graph with linear time. The procedure is as follow:

i) Finding maximal density sub-graph by computing the dominant eigenvector.
ii) Removing all vertices and hyperedges of the sub-graph from hypergraph.
iii) Repeating above steps until no vertex in hypergraph occurs.

This algorithm tends to generate many fine-grained clusters. We follow Tan and Kumar (2006) to merge clusters using two measures: cohesion and separation. The cohesion of a cluster C_i is defined as:

$$cohesion(\mathrm{C}_i) = \frac{\sum_{x \in C_i, y \in C_i} \#(e \mid x, y \in e)}{|\mathrm{C}_i|} \qquad (3)$$

where $\#(e \mid x, y \in e)$ is the number of hyperedges containing nodes x and y in C and $|C_i|$ is the number of vertices in C_i. Separation between two clusters C_i, C_j is defined as:

$$separation(\mathrm{C}_i, \mathrm{C}_j) = 1 - \left(\frac{\sum_{x \in C_i, y \in C_j} \#(e \mid x, y \in e)}{|\mathrm{C}_i| \times |\mathrm{C}_j|}\right) \qquad (4)$$

We merge cluster pairs with high cohesion and low separation. The intuition is that context instances in such pairs will maintain a relatively high degree of semantic similarity. High cohesion is defined as greater than average cohesion of all clusters. Low separation is defined as a reciprocal relationship between two clusters: if a cluster C_i has the lowest separations to a cluster C_j and C_j has the lowest separation to C_i, then the two (high cohesion) clusters are merged. This merging process is iterated until it converges.

4 Experiment and Evaluation

4.1 Dataset

Our WSI evaluation is based on the dataset provided by the SemEval-2013 shared 13th task. Test data was drawn from the Open American National Corpus (OANC) (Ide and Suderman, 2004) across a variety of genres and from both the spoken and written portions of the corpus. It consists of 4,806 instances of 50 target words: 20 verbs, 20 nouns and 10 adjectives. Due to the unsupervised nature of the task, participants were not provided with sense-labeled training data. However, WSI systems were provided with the ukWac corpus (Baroni et al., 2009) to use in inducing senses. Additionally, we used the SemEval-2013 lexical trial data sets as development sets to tune parameters.

4.2 Implementation details

The training data is extracted from uKWac corpus. For each target word, we extracted 10K context instances, and each instance is a sentence window containing the target word. Additionally, we randomly selected 10K sentences as common auxiliary corpus, including none of the target word. The training data are tagged with POS tags and lemmatized with TreeTagger (Schmid, 1994). Removing stop words, nouns are taken as features. Meanwhile, we also removed words that co-occur with the target of word less than 50 times over the whole ukWac data.

The training data in the model contains 20K instances: 10K instances of target word, 10K auxiliary instances. Specifically, we used the JGibbLDA[1] framework for topic model estimation and inference, and examined the following LDA parameters: number of topics K, dirichlet hyperparameters for document-topic distribution α and topic-term distribution β. We tested combinations in the ranges K=1000, 1500, 2000, α=5/K..50/K and β=0.001..0.1. The highest performance of the WSI system was found for K = 2000, α =0.025, β = 0.001. Similar to tuning the dirichlet hyperparameters of LDA, the best parameter γ in lexical chain extraction is 0.0001 in the ranges γ =0.01.. 0.000001.

We adopt the three clustering algorithms to cluster hypergraph[2]. The number of clusters is set as 10 for NHC and HEC, while Schype algorithm generates the number of the clusters (but requiring the edge-vertex ratio to be pre-defined), whose average number of senses is 31.8 after clusters are merged. Additionally, For Schype algorithm, we used the default values of parameters, except that the "min-clustscore" parameter, a minimal score to output a cluster, being tuned to 0.1.

The sense inventory acquired from the induction step can be used for disambiguation of individual instances. Each sense is represented as a vector, whose element is a word and the value of element is co-occurrence frequency with target word in the training set. Each test instance is also represented as a vector. The similarity between the instance and the induced sense is computed by using cosine function. For each test instance, it is compared with each sense separately, and finally the sense is selected if the cosine value is greater than a certain threshold λ. In experiment, λ is 0.04 for NHC and HEC, and is 0.1 for schype.

4.3 Evaluation measures

Evaluation in the SemEval-2013 WSI task can be divided into two categories:

1. A traditional WSD task for unsupervised WSD and WSI systems,
2. A clustering comparison setting that evaluates the similarity of the sense inventories for WSI systems.

[1] http://sourceforge.net/projects/jgibblda/

[2] The Hypergraph Analysis Toolbox (HAT) for NHC and HEC: http://lia.epfl.ch/index.php/research/relational-learning and the Schype's code: http://www.roslin.ed.ac.uk/tom-michoel/software/

In the first evaluation, we adopt a WSD task with three objectives: (1) detecting which senses are applicable, (2) ranking senses by their applicability, and (3) measuring agreement in applicability ratings with human annotators. Each objective uses a specific measurement:

i): Jaccard Index: given two sets of sense labels for an instance, X and Y, the Jaccard Index is used to measure the agreement: $\frac{X \cap Y}{X \cup Y}$. The Jaccard Index is maximized when X and Y use identical labels, and is minimized when the sets of sense labels are disjoint.

ii): Positionally-weighted Kendall's τ similarity: for graded sense evaluation, we consider an ranking scoring based on Kumar and Vassilvitskii(2010), which weights the penalty of reordering the lower positions less than the penalty of reordering the first ranks.

iii): Weighted Normalized Discounted Cumulative Gain (WNDCG): NDCG (Moffat and Zobel, 2008) normally compares the rankings of two lists. It is extended to weighting the DCG by considering the relative difference in the two weights.

Because the induced senses will likely vary in number and nature between systems, the WSD evaluation has to incorporate a sense alignment step, in which it performs by splitting the test in-stances into two sets: a mapping set and an evaluation set. The optimal mapping from induced senses to gold-standard senses is learned from the mapping set, and the sense alignment is used to map the predictions of the WSI system to pre-defined senses for the evaluation set. The particular split we use to calculate WSD effectiveness in this paper is 80%/20% (mapping/test), averaged across 5 random splits.

In the clustering evaluation, similarity between participant's clusters and the gold standard clusters is measured by way of two metrics.

i): Fuzzy Normalised Mutual information (NMI): it extends the method of (Lancichinetti et al., 2009) to compute NMI between overlapping clusters. Fuzzy NMI captures the alignment of the two clusters independent of the cluster sizes and therefore serves as an effective measure of the ability of an approach to accurately model rare senses.

ii): Fuzzy B-Cubed: it adapts the overlapping B-Cubed measured defined in Amigo et al. (2009) to the fuzzy clustering setting, and provides an item-based evaluation that is sensitive to the cluster size skew and effectively captures the expected performance of the system on a dataset where the cluster distribution would be equivalent.

4.4 Results

We compared our models with four baselines and three benchmark systems from the SemEval-2013 task. Four baselines are described as follows.

- Baseline MFS — most frequent sense baseline, assigning all test instances to the MFS in the test data (regardless of what applicability rating it was given).
- One sense — labels each instance with the same induced sense.
- One sense per instance (1clinst) — labels each instance with its own induced.
- Baseline Random-n — randomly assigns each test instance to one of n randomly selected induced senses, where n is the number of senses for the target word in WordNet 3.1.

Three benchmark systems as the following are those which achieved better results in the original SemEval-2013 task.

- AI-KU is based on a lexical substitution method.
- UoS uses dependency-parsed features from the corpus, which are then clustered into senses using the MaxMax algorithm (Hope and Keller, 2013).
- Unimelb is a non-parameter topic model which uses a Hierarchical Dirichlet Process (Teh et al., 2006) to automatically infer the number of senses from contextual and positional features.

4.4.1 Supervised evaluation

In the supervised evaluation, the automatically induced clusters are mapped to gold standard senses, using one part of the test set. The obtained mapping is used to label the other part of test set with gold standard tags, which means that the methods are evaluated in a standard WSD task. In experiment, we follow the 80/20 setup: 80% for mapping and 20% for test.

Table 1 shows the results of our systems on test data using all instances (including verbs, nouns and adjectives) for the WSD evaluation. As in previous WSD tasks, the MFS baseline on Jaccard Index measures is quite competitive, outperforming all systems in detecting which senses are applicable.

System	JI-F1	WKT-F1	WNDCG-F1
NHC	0.325	0.692	0.375
HEC	0.327	0.693	**0.376**
Schype	**0.376**	**0.753**	0.345
AI-KU	0.244	0.642	0.332
UNIMELB	0.213	0.62	0.371
UoS	0.232	0.625	0.374
MFS	0.455	0.465	0.339
One sense	0.192	0.609	0.288
1c1inst	0.0	0.095	0.0
Random-n	0.29	0.638	0.286

Table 1. The supervised results over the SemEval-2013 dataset.

System	FNMI	FBC
NHC	0.046	0.406
HEC	0.037	0.400
Schype	0.042	0.377
AI-KU	0.039	0.451
UNIMELB	**0.056**	**0.459**
UoS	0.045	0.448
MFS	-	-
One sense	0	0.632
1c1inst	0.071	0
Random-n	0.016	0.245

Table 2. The unsupervised results over the SemEval-2013 dataset.

However, most systems in this task were able to outperform the MFS baseline in ranking senses and quantifying their applicability.

On the other hand, it also indicates that our three systems achieve better or comparable scores And the Schype gets the highest scores in detecting senses and ranking senses over all systems.

4.4.2 Unsupervised evaluation

Unsupervised evaluation aims to measure the similarity of the induced sense inventories for WSI systems. Unlike supervised metrics, it avoids potential loss of sense information since this setting does not require any sense mapping procedure to convert induced senses to WordNet senses.

Table 2 shows the performance of our systems, benchmarks and baselines. It shows that the NMI-measure is biased towards the **one sense per instance** baseline and the FBC-measure **one sense** baseline. However, systems are capable of performing well in both the Fuzzy NMI and Fuzzy B-Cubed measures, thereby avoiding the extreme performance of either baseline. Generally, the performance of our model gets balanced scores.

4.5 Discussion

Topic models, such as LDA and HDP (Brody and Lapata, 2009; Lau et al., 2012), have been successfully adopted for WSI, in which one topic is viewed as one sense. Our work is motivated by lexical chain that represents the intrinsic semantic relatedness among context instances on the viewpoint of linguistics. Topic model is used to find lexical chains which are interpreted as topics. We have compared the Unimelb (Lau et al., 2013), a HDP topic model, with our model in the experiments. Additionally, we also follow Lau et al. (2012) to train a LDA model with a fixed number of topics based on our training data for WSI[3]. Table 3 shows the supervised result compared to the Schype.

These experiments show promising performance for our model, which captures richer semantic relatedness by using lexical chains. Lexical chains play a key role for the performance of our model. Intuitively, when lexical chains are too long, the higher-order relatedness would be mixed with some noises, while when lexical chains are too short, some higher-order relatedness will be missed. In order to verify the effectiveness of lexical chains, we tune the parameter γ in lexical chain extraction procedure and the results are shown in Figure 4.

Another issue is the impact of POS labels of word for WSI task. The test data in SemeVal-2013 WSI task contain nouns, verbs and adjectives. We also test the performance based on different POS labels. Table 4 gives the supervised evaluation performance of our three systems on adjectives, verbs and nouns respectively. We found that the performance for adjectives in sense detection is the best, verbs followed and nouns worst, whereas it's reversed in sense ranking. The probable interpretation is

[3] The LDA model parameters are set as follows: K=10, α =0.025, β = 0.001.

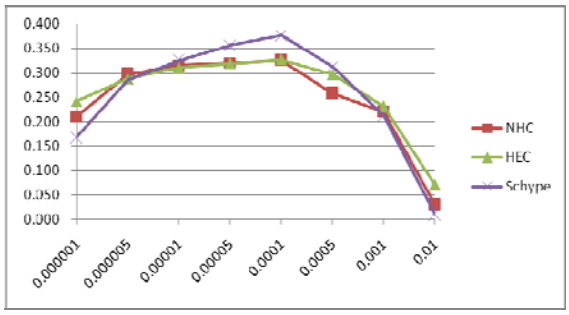


Figure 4. Performance analysis on Jaccard index measure for different threshold parameter γ in lexical chains extraction procedure.

Type	$LDA_{K=10}$	Schype
JI-F1	0.318	0.376
WKT-F1	0.692	0.753
WNDCG-F1	0.334	0.345

Table 3. The supervised results over the SemEval-2013 dataset between $LDA_{k=10}$ and Schype for WSI.

	NHC			HEC			Schype		
POS	**JI**	**WKT**	**WNDCG**	**JI**	**WKT**	**WNDCG**	**JI**	**WKT**	**WNDCG**
nouns	0.306	0.697	0.370	0.313	0.702	0.367	0.363	0.767	0.246
verbs	0.336	0.686	0.374	0.336	0.690	0.380	0.384	0.749	0.245
adjs	0.347	0.697	0.396	0.342	0.678	0.390	0.394	0.733	0.248

Table 4. The supervised performance of three algorithms respectively on nouns, verbs and adjectives.

that adjective's average sense number is the lowest, and the sense granularity is greater than verbs and nouns over the test data[4].

5 Conclusions and future work

In this paper, we present a hypergraph model in which a node represents an instance and a hyperedge represents higher-order semantic relatedness among instances. Compared with other strategies based on binary local comparison, the model captures complex semantic relatedness among the instances from a global perspective.

The evaluation results indicate that our model outperforms or reaches competitive performance comparable to other systems for the SemEval-2013 word sense induction task. Additionally, the experiments also show that both sense number and sense granularity of a target word affect the performance of WSI.

For future work, we would like to explore better ways to extract and evaluate lexical chain for WSI task. In addition, for the three clustering algorithms, they generally require the number of clusters or edge-vertex ratio to be pre-defined, so we will seek more effective hypergraph clustering algorithms to automatically determine the parameters. Finally, the hypergraph model proposed in this work is not specific to the sense induction task, and can be adapted for other applications, such as document classification and clustering, information retrieval, etc.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 61173062, 61373108, 61133012), the major program of the National Social Science Foundation of China (No. 11&ZD189), and the High Performance Computing Center of Computer School, Wuhan University.

Reference

Eneko Agirre and Aitor Soroa. 2007. SemEval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 7–12.

Eneko Agirre, David Martnez, Oier López de Lacalle, and Aitor Soroa. 2006. Two graph-based algorithms for state-of-the-art WSD. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language*

[4] In the test data, the average number of senses of nouns, verbs, adjectives respectively is 7.15, 6.85, 5.9 respectively.

Processing, pages 585–593.

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.

Regina Barzilay, Michael Elhadad, et al. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL workshop on Intelligent scalable text summarization*, volume 17, pages 10–17.

Osman Baskaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. AI-kU: Using substitute vectors and co-occurrence modeling for word sense induction and disambiguation. In *Proceedings of SemEval-2013*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of Machine learning research*, 3:993–1022.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*.

Stefan Bordag. 2006. Word sense induction: Triplet-based clustering and automatic evaluation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*

Joe Carthy. 2004. Lexical chains versus keywords for topic tracking. In *Computational Linguistics and Intelligent Text Processing*, pages 507–510.

Antonio Di Marco and Roberto Navigli. 2011. Clustering web search results with maximum spanning trees. In *AI* IA 2011: Artificial Intelligence Around Man and Beyond*, pages 201–212.

Beate Dorow and Dominic Widdows. 2003. Discovering corpus-specific word senses. In *Proceedings of the tenth Conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 79–82.

Katrin Erk, Diana McCarthy, and Nicholas Gaylord. 2009. Investigations on word senses and word usages. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 10–18.

Graeme Hirst and David St-Onge. 1998. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, 305:305–332.

David Hope and Bill Keller. 2013. Maxmax: a graph-based soft clustering algorithm applied to word sense induction. In *Computational Linguistics and Intelligent Text Processing*, pages 368–381.

Nancy Ide and Keith Suderman. 2004. The american national corpus first release. In *LREC*.

Nancy Ide, Tomaz Erjavec, and Dan Tufis. 2001. Automatic sense tagging using parallel corpora. In *NLPRS*, pages 83–90.

David Jurgens and Ioannis Klapaftis. 2013. SemEval-2013 Task 13: Word Sense Induction for Graded and Non-Graded Senses. In *Proceedings of the Seventh International Workshop on Semantic Evaluation* (SemEval 2013).

Ioannis P Klapaftis and Suresh Manandhar. 2007. Uoy: a hypergraph model for word sense induction & disambiguation. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 414–417.

Andrea Lancichinetti, Santo Fortunato, and János Kertész. 2009. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015.

Jey Han Lau, Paul Cook, and Timothy Baldwin. 2013. unimelb: Topic Modelling-based Word Sense Induction. In *Proceedings of the Seventh International Workshop on Semantic Evaluation* (SemEval 2013).

Jey Han Lau, Paul Cook and Diana McCarthy, David Newman and Timothy Baldwin. 2012. Word sense induction for novel sense detection. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*.

Okumura Manabu and Honda Takeo. 1994. Word sense disambiguation and text segmentation based on lexical cohesion. In *Proceedings of the 15th conference on Computational linguistics-Volume 2*, pages 755–761.

Suresh Manandhar, Ioannis P Klapaftis, Dmitriy Dligach, and Sameer S Pradhan. 2010. SemEval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic*

Evaluation, pages 63–68.

Tom Michoel and Bruno Nachtergaele. 2012. Alignment and integration of complex networks by hypergraph-based spectral clustering. *Physical Review E*, 86(5):056111.

Alistair Moffat and Justin Zobel. 2008. Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems (TOIS)*, 27(1):2.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):21–48.

Roberto Navigli. 2009. Word sense disambiguation: A survey. ACM Computing Surveys (CSUR),41(2):10.

Michael Steinbach Pang-Ning Tan and Vipin Kumar. 2006. *Introduction to Data Mining*. Pearson Addison Wesley.

Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD*, pages 613–619.

Li Pu and Boi Faltings. 2012. Hypergraph learning with hyperedge expansion. In *Machine Learning and Knowledge Discovery in Databases*, pages 410–425.

Amruta Purandare and Ted Pedersen. 2004. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the Conference on Computational Natural Language Learning*, pages 41–48. Boston.

Kumar, Ravi and Vassilvitskii, Sergei. 2010. Generalized distances between rankings. In *Proceedings of the 19th international conference on World wide web*, pages 571-580.

Steffen Remus and Chris Biemann. 2013. Three knowledge-free methods for automatic lexical chain extraction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 989–999, Atlanta, Georgia, June.

Martin Riedl and Chris Biemann. 2012. Sweeping through the topic space: bad luck? roll again! In *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, pages 19–27.

Samuel Rota Bulo and Marcello Pelillo. 2012. A game-theoretic approach to hypergraph clustering.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of international conference on new methods in language processing*, volume 12, pages 44–49.

Hinrich Schütze. 1998. Automatic word sense discrimination. Computational linguistics, 24(1):97–123.

Amnon Shashua, Ron Zass, and Tamir Hazan. 2006. Multi-way clustering using super-symmetric non-negative tensor factorization. In Computer Vision–ECCV 2006, pages 595–608.

Nicola Stokes, Joe Carthy, and Alan F Smeaton. 2004. Select: a lexical cohesion based news story segmentation system. AI Communications, 17(1):3–12.

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical dirichlet processes. Journal of the american statistical association, 101(476).

Tim Van de Cruys, Marianna Apidianaki, et al. 2011. Latent semantic word sense induction and disambiguation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*: Human Language Technologies (ACL/HLT), pages 1476–1485.

Jean Véronis. 2004. Hyperlex: lexical cartography for information retrieval. Computer Speech & Language, 18(3):223–252.

Dominic Widdows and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th International Conference on Computational linguistics-Volume 1*, pages 1–7.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196.

Mehmet Ali Yatbaz, Enis Sert, and Deniz Yuret. 2012. Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 940–951.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2006. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems*, pages 1601–1608.

Minimally Supervised Classification to Semantic Categories using Automatically Acquired Symmetric Patterns

Roy Schwartz[1] **Roi Reichart**[2] **Ari Rappoport**[1]

[1]Institute of Computer Science, The Hebrew University
{roys02|arir}@cs.huji.ac.il

[2]Technion IIT
roiri@ie.technion.ac.il

Abstract

Classifying nouns into semantic categories (e.g., animals, food) is an important line of research in both cognitive science and natural language processing. We present a minimally supervised model for noun classification, which uses symmetric patterns (e.g., "X and Y") and an iterative variant of the k-Nearest Neighbors algorithm. Unlike most previous works, we do not use a predefined set of symmetric patterns, but extract them automatically from plain text, in an unsupervised manner. We experiment with four semantic categories and show that symmetric patterns constitute much better classification features compared to leading word embedding methods. We further demonstrate that our simple k-Nearest Neighbors algorithm outperforms two state-of-the-art label propagation alternatives for this task. In experiments, our model obtains 82%-94% accuracy using as few as four labeled examples per category, emphasizing the effectiveness of simple search and representation techniques for this task.

1 Introduction

The role of language is to express meaning. In the field of NLP, there has been an increasingly growing number of approaches that deal with semantics. Among these are vector space models (Turney and Pantel, 2010; Baroni and Lenci, 2010), lexical acquisition (Hearst, 1992; Dorow et al., 2005; Davidov and Rappoport, 2006), universal cognitive conceptual annotation (Abend and Rappoport, 2013) and automatic induction of feature representations (Collobert et al., 2011). In this paper, we utilize extremely weak supervision to classify words into fundamental cognitive semantic categories.

There are several types of semantic categories expressed by languages, e.g., objects, actions, and properties. We follow human development, acquiring coarse-grained categories and distinctions before detailed ones (Mandler, 2004). Specifically, we focus on the major class of concrete "*things*" (Langacker, 2008, Ch. 4), roughly corresponding to nouns – the main participants in linguistic clauses – that are universally present in the semantics of virtually all languages (Dixon, 2005).

Most works on noun classification to semantic categories require large amounts of human annotation to build training corpora for supervised algorithms (Bowman and Chopra, 2012; Moore et al., 2013) or rely on language-specific resources such as WordNet (Evans and Orăsan, 2000; Orăsan and Evans, 2007). Such heavy supervision is labor intensive and makes these models domain and language dependent.

Our reasoning is that weak supervision is highly valuable for semantic categorization, as it can compensate for the lack of input from the senses in text corpora. Our model therefore performs semantic category classification using only a small number of labeled seed words per category. The experiments we conduct show that such weak supervision is sufficient to construct a high quality classifier.

A key component of our model is the application of symmetric patterns. We define patterns to be sequences of words and wildcards (e.g., "X is a dog", "both X and Y", etc.). Accordingly, *symmetric* patterns are patterns that contain exactly two wildcards, where both wildcards are interchangeable. Examples of symmetric patterns include "X and Y", "X as well as Y" and "neither X nor Y".

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1612–1623, Dublin, Ireland, August 23-29 2014.

Works that apply symmetric patterns in their model generally require expert knowledge in the form of a pre-compiled set of patterns (Widdows and Dorow, 2002; Kozareva et al., 2008). In this work, we extract symmetric patterns in an unsupervised manner using the (Davidov and Rappoport, 2006) algorithm. This algorithm automatically extracts a set of symmetric patterns from plain text using simple statistics about high and low frequency word co-occurrences. The unsupervised nature of our approach makes it domain and language independent.

Our model addresses semantic classification in a transductive setup. It takes advantage of word similarity scores that are computed based on symmetric pattern features, and propagates information from concepts with known classes to the rest of the concepts. For this aim we apply an iterative variant of the k-Nearest Neighbors algorithm (denoted with I-k-NN) to a graph in which vertices correspond to nouns and word pairs are connected with edges based on their participation in symmetric patterns.

We experiment with a subset of 450 nouns from the CSLB dataset (Devereux et al., 2013), which were annotated with semantic categories by thirty human subjects. From the set of semantic categories in this dataset, we select categories that are both frequent and have a high inter-annotator agreement (Section 2). This results in a set of four semantic categories – *animacy, edibility, is_a_tool* and *is_worn*.

Our experiments show that our model performs very well even when only a small number of labeled seed words are available. For example, on the task of binary classification with respect to a single category, when using as few as four labeled seed words, classification accuracy reaches 82%-94%.

Furthermore, our model outperforms several strong baselines for this task. First, we compare our model against a model that uses a deep neural network word embedding baseline (Collobert et al., 2011) instead of our symmetric pattern based features, and applies the exact same I-k-NN algorithm. In recent years, deep networks word embeddings obtained state-of-the-art results in several NLP tasks (Collobert and Weston, 2008; Socher et al., 2013). However, in our task, features based on simple, intuitive and easy to compute symmetric patterns, lead to substantially better performance (average improvement of 0.15 F1 points). Second, our model outperforms two baseline models that utilize the same symmetric pattern classification features as in our model, but replace our simple I-k-NN algorithm with two leading label propagation alternatives (the normalized graph cut (N-Cut) algorithm (Yu and Shi, 2003) and the Modified Adsorption (MAD) algorithm (Talukdar and Crammer, 2009)). The average improvement over these two baselines is 0.21 and 0.03 F1 points .

The rest of the paper is organized as follows. Section 2 describes our semantic classification task and, particularly, the semantic classes that we aim to learn. Section 3 presents our method for automatic symmetric patterns acquisition. Sections 4, 5 and 6 describe our model, experimental setup and results, respectively. Related work is finally surveyed in Section 7.

2 Task Definition

The task we tackle in this paper is the classification of nouns into semantic categories. This section defines the categories we address and the dataset we use.

Semantic Categorization of Concrete Nouns. We focus on concrete "*things*" (Langacker, 2008), which correspond to *noun* categories. Nouns are interesting because they are the most basic lexical semantic categories. Specifically, children acquire nouns before any other category (Clark, 2009). Moreover, noun categories are generally not subjective. For example, it is hard to argue that a dog is not an animal, or that an apple is inedible, in most reasonable contexts. The context independent nature of nouns makes them appropriate for a type level classification task, such as the one we tackle. In order to provide a better description of the categories we aim to predict, we now turn to discuss the CSLB dataset, with which we experiment.

Dataset. We experiment with the CSLB property norms dataset (Devereux et al., 2013). In order to prepare this data set, thirty human subjects were presented with 638 concrete nouns and were asked to write the categories associated with each concept. Table 1 presents the top five categories for the nouns *apple* and *horse*.

Noun	Categories
Apple	is_a_fruit, does_grow_on_trees, is_green, is_red, has_pips_seeds
Horse	is_ridden, is_an_animal, has_four_legs, has_legs, has_hooves

Table 1: Five most frequent semantic categories for the words *apple* and *horse* in the CSLB dataset.

Category Selection. The CSLB dataset consists of a total of 2725 semantic categories. We apply a selection mechanism that provides us with a dataset in which (1) only noun categories (*things*) are included; and (2) only semantic categories that are prominent across humans are considered. For this, we apply the following filtering stages. First, since the vast majority of annotated categories are rare (for example, 1691 categories are assigned to a single noun only), we set a minimum threshold of 35 nouns per category (5% of the nouns). After removing highly infrequent categories, 28 are left. We then apply an inter-annotator agreement criterion: for each semantic category c, we compute the average number of human annotators that associated this category with a given noun, across the nouns annotated with c. We select the category c only if the value of this statistic is higher than 10 subjects ($1/3$ of the subjects), which results in a semantic category set of size 18. Finally, we discard categories, such as *color* and *size*, that do not correspond to *things*. We are left with four noun semantic categories: *animacy* (animals), *edibility* (food items), *is_a_tool* (tools), and *is_worn* (clothes).

Interestingly, the resulting semantic categories can also be justified from a cognitive perspective. There is a large body of work indicating that our categories relate to brain organization principles. For example, Just et al. (2010) showed that food products and tools arouse different brain activation patterns. Moreover, a number of works showed that both animate objects and tools are represented in specific brain regions. These works used neuroimaging methods such as functional magnetic resonance imaging (fMRI) (Naselaris et al., 2012), electroencephalography (EEG) (Chan et al., 2011) and magnetoencephalography (MEG) (Sudre et al., 2012). See (Martin, 2007) for a detailed survey. This parallel evidence to the prominence of our categories provides substance for intriguing future research.

3 Symmetric Patterns

Patterns. In this work, patterns are combinations of words and wildcards, which provide a structural phrase representation. Examples of patterns include "*X and Y*", "*X such as Y*", "*X is a country*", etc. Patterns can be used to extract various relations between words. For example, patterns such as "X of a Y" ("basement *of a* building") can be useful for detecting the meronymy (part-of) relation (Berland and Charniak, 1999). Symmetric patterns (e.g., "X *and* Y", "France *and* Holland"), which we use in this paper, can be used to detect semantic similarity between words (Widdows and Dorow, 2002).

Symmetric Patterns. *Symmetric* patterns are patterns that contain exactly two wildcards, and where these wildcards are interchangeable. Examples of symmetric patterns include "X *and* Y", "X *or* Y" and "X *as well as* Y". Previous works have shown that word pairs that participate in symmetric patterns bare strong semantic resemblance, and consequently, that these patterns can be used to cluster words into semantic categories, where a high precision, but low coverage (recall) solution is good enough (Dorow et al., 2005; Davidov and Rappoport, 2006). A key observation of this paper is that symmetric patterns can be also used for semantic classification, where recall is as important as precision.

Flexible Patterns. It has been shown in previous work (Davidov and Rappoport, 2006; Turney, 2008; Tsur et al., 2010; Schwartz et al., 2013) that patterns can be extracted from plain text in a fully unsupervised manner. The key idea that makes this procedure possible is the concept of "flexible patterns", which are composed of high frequency words (HFW) and content words (CW). Every word in the language is defined as either HFW or CW, based on the number of times this word appears in a large corpus. This clustering procedure is applied by traversing a large corpus, and marking words that appear with corpus frequency higher than a predefined threshold t_1 as HFWs, and words with corpus frequency lower than t_2 as CWs.[1]

[1]We follow (Davidov and Rappoport, 2006) and set $t_1 = 10^{-5}, t_2 = 10^{-3}$. Note that some words are marked both as HFW and as CW. See (Davidov and Rappoport, 2008) for discussion.

The resulting clusters have a desired property: HFWs are comprised mostly of function words (prepositions, determiners, etc.) while CWs are comprised mostly of content words (nouns, verbs, adjectives and adverbs). This coarse grained clustering is useful for pattern extraction from plain text, since language patterns tend to use fixed function words, while content words change from one instance of the pattern to another (Davidov and Rappoport, 2006).

Flexible patterns are extracted by traversing a large corpus and, based on the clustering of words to CWs and HFWs, extracting all pattern instances. An extracted pattern instance consists of CW wildcards and the actual words replacing the HFWs in the pattern type. Consider the sentence "*The boy is happy and joyful*". Replacing the content words with the CW wildcard results in "*The* CW *is* CW *and* CW". From this intermediate representation, we extract word sequences of a given length constraint and denote them as flexible patterns.[2] The flexible patterns of length 5 extracted from this sentence are "*The* CW *is* CW *and*" and "CW *is* CW *and* CW". The reader is referred to (Davidov and Rappoport, 2006) for more details.

Automatically Extracted Symmetric Patterns. Most models that incorporate symmetric patterns use a predefined set of patterns (Widdows and Dorow, 2002; Kozareva et al., 2008). In this work, we apply an automatic, completely unsupervised procedure for symmetric pattern extraction. This procedure, described in Algorithm 1, is adopted from (Davidov and Rappoport, 2006).

The procedure first extracts flexible patterns that contain exactly two CW wildcards. It then selects those flexible patterns in which both CWs are interchangeable. That is, it selects a pattern p if every word pair CW_1, CW_2 that participates in p indicates with high probability that the word pair C_2, C_1 also participates in p. For example, for the symmetric pattern "CW *and* CW", both "cats *and* dogs" and "dogs *and* cats" are semantically plausible expressions, and are therefore likely to appear in a large corpus. On the other hand, the flexible pattern "CW *such as* CW" is asymmetric, as exemplified in expressions like "countries *such as* France", where replacing the CWs does not result in a semantically plausible expression (# "France *such as* countries"). The selection process is done by computing the proportion of CW_1, CW_2 pairs that participate in p for which CW_2, CW_1 also participates in p. Patterns for which this proportion exceed a certain threshold are selected.

We apply Algorithm 1 on the google books 5-gram corpus (Michel et al., 2011)[3] and extract 20 symmetric patterns. Some of the more interesting symmetric patterns extracted using this algorithm include "CW *and the* CW", "*from* CW *to* CW", "CW *rather than* CW" and "CW *versus* CW". In the next section we present our approach to semantic classification, which makes use of automatically acquired symmetric patterns for word similarity computations.

4 Model

In this section we present our model for binary word classification according to a single semantic category in a minimally-supervised, transductive setup. Given a set of words, we label a small number of words with their correct label according to the category at hand (+1 for words that belong to the category, -1 for words that do not belong to it). Our model is based on an undirected weighted graph, in which vertices correspond to words, and edges correspond to relations between words. Our goal is to classify the unlabeled words (vertices) in the graph through a label propagation process. We now turn to describe our model in detail.

Graph Construction. We construct our graph such that an edge is added between two words (vertices) if both words participate in a symmetric pattern. The edge generation process is performed as follows. We first apply our symmetric pattern extraction procedure (Algorithm 1), and denote the set of selected symmetric patterns with P. We then traverse a large corpus[4] and extract all word pairs that participate in any pattern $p \in P$. We denote the number of occurrences of a word pair (w_1, w_2) in such patterns with f_{w_1,w_2}. Finally, we select all word pairs (w_1, w_2) for which $min(f_{w_1,w_2}, f_{w_2,w_1}) > \alpha$. Each such

[2] We set the maximal flexible pattern length to be 5.

[3] `https://books.google.com/ngrams`

[4] We use google books 5-grams (Michel et al., 2011).

Algorithm 1 Symmetric pattern extraction

procedure EXTRACT_SYMMETRIC_PATTERNS(C, W)
  ▷ C is a large corpus, W is a lexicon
  ▷ Traverse C and extract all flexible patterns of length 3-5 that appear in C and contain exactly two content words
  $P \leftarrow$ extract_flexible_patterns(C, W)
  for $p \in P$ **do**
    if p appears in $<10^{-6}$ of the sentences in C **then**
      Discard p and continue
    end if
    $G_p \leftarrow$ a directed graph s.t. $V(G_p) \leftarrow W$, $E(G_p) \leftarrow \{(w_1, w_2) \in W^2 : w_1, w_2$ participate in at least one instance of $p\}$
    ▷ An undirected graph based on the bidirectional edges of the G_p
    $symG_p \leftarrow$ an undirected graph: $\{(w_1), (w_1, w_2) : (w_1, w_2) \in E(G_p) \wedge (w_2, w_1) \in E(G_p)\}$
    ▷ Two measures of symmetry
    $M_1 \leftarrow \frac{|V(symG_p)|}{|V(G_p)|}, M_2 \leftarrow \frac{|E(symG_p)|}{|E(G_p)|}$
    ▷ Symmetric pattern candidates are those with high M_1 and M_2 values
    if $\min(M_1, M_2) < 0.05$ **then**
      Discard p
    end if
  end for
  for $p \in P$ **do**
    ▷ E.g., "CW and CW" is contained in "both **CW and CW**"
    if $\exists p' \in P$ s.t. p' is contained in p **then**
      Discard p
    end if
  end for
  return The top 20 members of P with the highest M_1 value
end procedure

pair is connected with an edge e_{w_1,w_2} in the graph, where the edge weight (denoted with w_{w_1,w_2}) is the geometric mean between f_{w_1,w_2} and f_{w_2,w_1}.

Label Propagation. Given a small number of annotated words (vertices), our goal is to propagate the information these words convey to other words in the graph. To do so, we apply an iterative variant of the k-Nearest Neighbors algorithm (I-k-NN). This iterative variant is required due to graph sparsity; when starting with a small set of labeled vertices, most unlabeled vertices do not have any labeled neighbor, and thus running the standard k-NN algorithm would result in classifying a very small number of vertices. Our approach is to run iterations of the k-NN algorithm, and thus propagate information to additional vertices at each iteration. At each k-NN step, the algorithm selects words that have at least one labeled neighbor. From this set, only the words that have the highest ratio of neighbors with the same label are selected, and are assigned with this label.

Consider a simple example. Say we have three candidate vertices a, b and c, where a has one neighbor with label +1 ($ratio(a) = 1/1 = 1.0$), b has two neighbors with label -1 ($ratio(b) = 2/2 = 1.0$) and c has three neighbors with label +1 and one neighbor with label -1 ($ratio(c) = max(3, 1)/4 = 3/4$). Then, a and b are selected and are assigned with $+1$ and -1, respectively.

Seed Expansion. In minimally supervised setups like ours, the model is initialized with a small set of labeled seed examples. A natural approach in such settings is to apply a seed expansion step, in order to obtain a larger set of labeled seeds. Our method uses the same graph construction procedure described above, but uses a larger edge generation threshold $\beta >> \alpha$.[5] We then apply an iterative procedure that labels a vertex v with a label l if either (a) v is directly connected to γ of the vertices labeled with l or (b) v is connected to δ_l of the neighbors of vertices labeled with l.[6] This procedure is run iteratively until no more vertices meet any of the criteria (a) or (b).

[5] Using a larger threshold results in a sparser graph. Nevertheless, each edge in this graph is more likely to represent a real semantic relation.

[6] γ and δ_l are hyperparameters tuned on our development set (see Section 5.2).

5 Experimental Setup

5.1 Baselines

We compare our model to two types of baselines. The first (Classification Features Baselines) utilizes the I-k-NN algorithm, along with a different set of classification features. The second (Label Propagation Baselines) utilizes the same classification features as we do, but replaces I-k-NN with a more sophisticated label propagation algorithm.

5.1.1 Classification Features Baselines

In this set of baselines, we use different methods for building our graph. Concretely, instead of adding edges for pairs of words that appear in the same symmetric pattern, we use word similarity measures based on different feature sets as described below. The process of building the graph using the baseline word similarity measures is described in Section 5.2.

SENNA. Deep neural networks have gained recognition as leading feature extraction methods for word representation (Collobert and Weston, 2008; Socher et al., 2013). We use SENNA,[7] a deep network based word embedding method, which has been used to produce state-of-the-art results in several NLP tasks, including POS tagging, chunking, NER, parsing and SRL (Collobert et al., 2011). We use the cosine similarity between two word embeddings as a word similarity measure.

Brown. This baseline is derived from the clustering induced by the Brown algorithm (Brown et al., 1992).[8] This clustering, in which words share a cluster if they tend to appear in the same lexical context, has shown useful for several NLP tasks, including POS tagging (Clark, 2000), NER (Miller et al., 2004) and dependency parsing (Koo et al., 2008). We use it in order to control for the possibility that a simple contextual preference similarity correlates with similarity in semantic categorization better than symmetric pattern features.

The Brown algorithm builds a binary tree, where words are located at leaf nodes. We use the graph distance between two words u, v (i.e., the shortest path length between u, v in the tree) as a word similarity measure for building our graph.

5.1.2 Label Propagation Baselines

In this type of baselines, we replace I-k-NN with a different label propagation algorithm, while still using the symmetric pattern features for word similarity computations.

N-Cut. This baseline applies the normalized graph cut algorithm (Yu and Shi, 2003)[9] for label propagation. Given a graph $G = (V, E)$ and two sets of vertices $A, B \subseteq V$, this algorithm defines $links(A, B)$ to be the sum of edge weights between A and B. The objective of the algorithm is to find the clusters $A, V \setminus A$ that minimize $\frac{links(A,V \setminus A)}{links(A,V)}$. The algorithm of (Yu and Shi, 2003) is particularly efficient for this problem as it avoids eigenvector computations which may become computationally prohibitive for large graphs (for more details, see their paper). In order to encode information about our labeled seed words, we hard-code a large negative value (-100000) to the weights of edges between seed words with different labels (positive and negative).

MAD. The Modified Adsorption (MAD) algorithm (Talukdar and Crammer, 2009)[10] is an extension of the Adsorption algorithm (Baluja et al., 2008). MAD is a stochastic graph-based label propagation algorithm which has shown to have a number of attractive theoretical properties and demonstrated good experimental results.

[7] The word embeddings were downloaded from `http://ml.nec-labs.com/senna/`

[8] We use the clusters induced by (Koo et al., 2008), who applied the Brown algorithm implementation of (Liang, 2005) to the BLLIP corpus (Charniak et al., 2000). `http://www.people.csail.mit.edu/maestro/papers/bllip-clusters.gz`

[9] `http://www.cis.upenn.edu/~jshi/software/Ncut_9.zip`

[10] `http://github.com/parthatalukdar/junto`

5.2 Experiments

Graph Construction. We experiment with the CSLB dataset (Devereux et al., 2013), consisting of 638 nouns, annotated with their semantic categories by thirty human subjects. We first omit all nouns that are annotated as having more than one sense, and use the remaining 603 nouns to build our graph. From these nouns, 146 nouns are annotated as animate, 115 as edible, 50 as wearable and 35 as tools.[11] We then discard nouns that have less than two neighbors, which results in a final set of 450 nouns (vertices).

The graphs used in the classification features baselines are different than those used by the models that use our symmetric pattern classification features, since the features define the graph structure (Section 4). In order to provide a meaningful comparison, we build graphs with the same number of vertices for each of these baselines. We do so by selecting the n edges with the highest weight, together with the set of vertices connected by these edges, such that the resulting graph has 450 vertices. Working with these sets of vertices is the optimal setting for these baselines, as the resulting graphs are the ones with the highest possible edge weights for graphs with 450 vertices.[12]

Parameter Tuning. In order to avoid adding additional labeled examples for the sake of parameter tuning, we set the hyperparameter values to the ones for which each model performs best on an auxiliary semantic classification task. Concretely, we experiment with a fifth semantic category (*audibility*),[13] which is not part of our evaluation setting, for parameter tuning. Note that this results in our model having the same hyperparamter values for all four classification tasks.

In order to ensure that the models assign all participating words with labels, we set α=3, where α is the minimal number of times a word pair should appear in the same symmetric pattern in order to have an edge in our graph (See Section 4). In our seed expansion procedure, where we search for seeds whose label is predicted with high confidence, only word pairs that appear at least β=50 times in the same symmetric pattern are assigned an edge in the graph. We set the seed expansion procedure parameters to be $\gamma = 0.6$, $\delta_{+1} = 0.5, \delta_{-1} = 0.2$.

Evaluation. For each classification task, we run experiments with 4, 10, 20 and 40 labeled seed words. In each setting, half of the labeled seed words are assigned a positive label and the other half are assigned a negative label. For each semantic category and labeled seed set size, we repeat our experiment 1000 times, each of which with a different set of randomly selected labeled seed examples, and report the average results. We report both accuracy (number of correct labels divided by number of vertices in the graph) and F1 score, which is the harmonic mean of p (the average precision across labels) and r (average recall across labels).

These two measures represent complementary aspects of our results. On the one hand, accuracy is the most natural classification performance measure. On the other hand, the number of positive labels is substantially smaller than the number of negative labels,[14] and thus this measure can be manipulated: a dummy model that always assigns the negative label gets a high accuracy. The F1 score controls against such models by assigning them low scores.

6 Results

Our experiments are designed to explore two main questions: (a) the value of symmetric patterns as semantic classification features, compared to state-of-the-art word clustering and embedding methods; and (b) the required complexity of an algorithm that can propagate information about semantic similarity. Particularly, we test the value of our simple I-k-NN algorithm compared to more sophisticated alternatives.

A Minimally Supervised Setting. Our first set of experiments is in a minimally supervised setting where only two positive and two negative examples are available for each binary classification task. This

[11] Some words are classified as belonging to more than one category (e.g., "chicken" is both animate and edible).

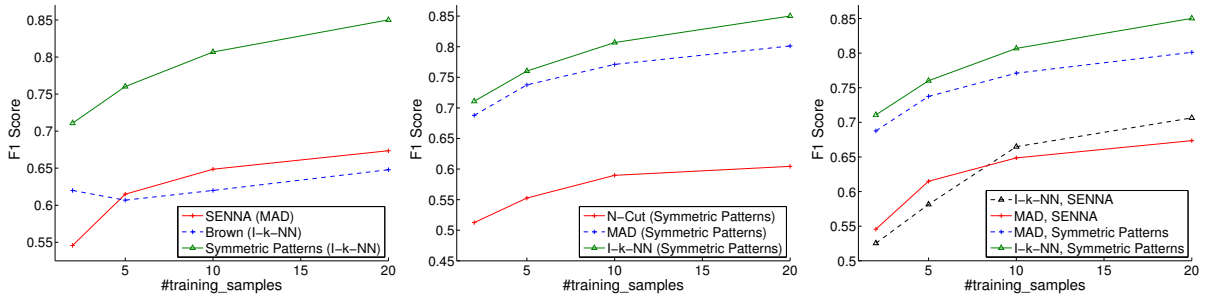
[12] The resulting graphs are actually denser than the symmetric patterns-based graph: 14K and 9K edges for the Brown and SENNA graphs, respectively, compared to $<$ 5K edges in the symmetric patterns graph.

[13] We used four labeled seed words in these experiments.

[14] Only 6-25% of the nouns have a positive label.

		Animacy			Edibility			is_worn			is_a_tool		
		SP	SENNA	Brown	SP	SENNA	Brown	SP	SENNA	Brown	SP	SENNA	Brown
Acc.	MAD	80.4%	77.7%	12.0%	75.0%	56.5%	14.8%	82.7%	66.8%	14.7%	73.3%	67.7%	12.2%
	N-Cut	71.4%	60.4%	51.2%	75.5%	59.4%	50.9%	83.3%	71.5%	51.4%	**82.7%**	77.1%	52.0%
	I-k-NN	**85.1%**	76.0%	55.5%	**82.2%**	56.8%	68.0%	**94.1%**	70.9%	66.7%	82.0%	75.7%	65.0%
F1	MAD	0.77	0.76	0.18	0.69	0.55	0.24	0.71	0.56	0.22	0.58	0.47	0.17
	N-Cut	0.49	0.45	0.46	0.51	0.44	0.45	0.61	0.56	0.41	0.56	0.50	0.38
	I-k-NN	**0.78**	0.70	0.48	**0.71**	0.53	0.62	**0.86**	0.59	0.55	**0.64**	0.52	0.51

Table 2: Accuracy and F1 score comparison between our model and the baselines. The columns correspond to the type of classification features used by the model: SP – symmetric patterns, SENNA – word embeddings extracted using deep networks (Collobert et al., 2011), Brown – Brown word clustering (Brown et al., 1992). The rows correspond to the algorithms applied by the model: N-Cut – the normalized graph cut algorithm (Yu and Shi, 2003), MAD – the modified adsorption algorithm (Talukdar and Crammer, 2009), I-k-NN – our iterative k-NN algorithm. Our model (I-k-NN + SP) is superior in all cases, except for the accuracy of the "is_a_tool" semantic category, where it is second only to N-Cut+SP.



(a) Classification Features Comparison (b) Algorithm Comparison (c) Top four Best Models

Figure 1: (a) Comparison of the different classification features. The figure shows the F1 scores of the best model that uses each of the feature sets (the label propagation algorithm used in each model appears in parentheses). (b) Comparison of the different label propagation algorithms. The figure shows the F1 scores of the best model that uses each of the algorithms (the classification feature sets used in each model appears in parentheses. It is always symmetric patterns). (c) The four best overall models (algorithm + classification feature set). The figures show that the symmetric pattern feature set is superior to the other feature sets, and that I-k-NN is superior or comparable to the other label propagation algorithms.

setup enables us to explore the performance of our model when the amounts of labeled training data is taken to the possible minimum.

Table 2 presents our results. With respect to objective (a), the table clearly demonstrates that symmetric patterns lead to much better results compared to the alternatives. Particularly, for all four semantic categories, and across both evaluation measures, it is a model that utilizes symmetric pattern classification features that achieves the best results. The average difference between the best model that uses symmetric patterns and the best model that does not is 12.5% accuracy and 0.13 F1 points. The dominance of symmetric pattern classification features is further demonstrated by the fact that a model that uses these features always performs better than a model that uses the same algorithm but different features.

With respect to objective (b) the table shows that I-k-NN provides a large improvement in seven out of eight (*category* $\times$ *evaluation measure*) settings. The average difference between the best model that utilizes I-k-NN and the best model that applies a different algorithm is 5.4% accuracy and 0.06 F1 points.

Analysis of Labeled Seed Set Size. In order to get a wider perspective on our model, we repeated our experiments with various sizes of the labeled seed set: 5,10 and 20 positive and negative labeled examples per semantic category. For brevity, only the F1 score results of the edibility category are presented. The trends observed on the other semantic categories (as well as when using the accuracy measure) are very similar.

Figure 1a compares the different classification features. For each feature f, results of the best performing model that uses f are shown. The results reveal that symmetric patterns clearly outperform the other features. The average differences between the best symmetric patterns-based model and the best

models that use the other features are 0.15 (SENNA) and 0.16 (Brown) F1 points.

Figure 1b compares the different label propagation algorithms. For each algorithm a, results for the best performing model that uses a are presented. The results reveal that the I-k-NN algorithm outperforms both algorithms by 0.03 (MAD) and 0.21 (N-Cut) F1 points. The results also show that for all algorithms, the best performing model uses symmetric patterns classification features, which further demonstrates the dominance of these features.

Finally, Figure 1c presents the four top performing models (algorithm + classification feature). In accordance with the other findings presented in this section, the top two models, which outperform the other models by a large margin, apply symmetric pattern classification features.

Seed Expansion Effect. Our model uses a seed expansion procedure in order to expand a small set of labeled seed words to a larger set (see Section 4). In order to assess the quality of this procedure we compute, for each semantic category, the average size of the expanded set and the accuracy of the new seeds (i.e., the proportion of new seeds that are labeled correctly). Results show that the initial set is increased from four seeds (two positive + two negative) to 48-52, and that the accuracy of the new seeds is as high as 88-99%. Our experiments also show that this procedure provides a substantial performance boost to our I-k-NN algorithm, which obtains a 7.2% accuracy and 0.05 F1 points improvement (averaged over the four semantic categories) when applied with the expanded set of labeled seed words compared to the original set of size four.

7 Related Work

Classification into Semantic Categories. Several works tackled the task of semantic classification, mostly focusing on animacy, concreteness and countability. The vast majority of these works are either supervised (Hatzivassiloglou and McKeown, 1997; Baldwin and Bond, 2003; Peng and Araki, 2005; Øvrelid, 2005; Nagata et al., 2006; Xing et al., 2010; Kwong, 2011; Bowman and Chopra, 2012) or make use of external, language-specific resources such as WordNet (Orăsan and Evans, 2001; Orăsan and Evans, 2007; Moore et al., 2013). Our work, in contrast, is minimally supervised, requiring only a small set of labeled seed words.

Ji and Lin (2009) classified words into the gender and animacy categories, based on their occurrences in instances of hand-crafted patterns such as "X *who* Y" and "X *and his* Y". While their model uses patterns that are tailored to the animacy and gender categories, our model uses automatically induced patterns and is thus applicable to a range of semantic categories.

Finally, Turney et al. (2011) built a label propagation model that utilizes LSA (Landauer and Dumais, 1997) based classification features. They used their model to classify nouns into the concrete/abstract category using 40 labeled seed words . Unlike our model, which requires only a small set of labeled seeds, their algorithm is actually heavily supervised, requiring thousands of labeled examples for selecting the seed set of labeled words that are used for propagation. Our model, on the other hand, does not require any seed selection procedure, and utilizes a randomly selected set of labeled seed words.

Lexical Acquisition. Another line of work focused on the acquisition of semantic categories. In this setup, a model aims to find a core seed of words belonging to a given category, sacrificing recall for precision. Our model tackles a different task, namely the classification of words according to a given category where both recall and precision are to be optimized.

Lexical acquisition models are either supervised (Snow et al., 2006), unsupervised, making use of symmetric patterns (Davidov and Rappoport, 2006), or lightly supervised, requiring expert, language specific knowledge for compiling a set of hand-crafted patterns (Widdows and Dorow, 2002; Kozareva et al., 2008; Wang and Cohen, 2009). Other models require syntactic annotation derived from a supervised parser to extract coordination phrases (Riloff and Shepherd, 1997; Dorow et al., 2005). Our model automatically induces symmetric patterns, obtaining high quality results without relying on any type of language specific knowledge or annotation. Moreover, some of the works mentioned above (Riloff and Shepherd, 1997; Widdows and Dorow, 2002; Kozareva et al., 2008) also require manually selected label

seeds to achieve good performance; in contrast, our work performs very well with a randomly selected set of labeled seed words.

8 Conclusion

We presented a minimally supervised model for noun classification into coarse grained semantic categories. Our model obtains 82%-94% accuracy on four semantic categories even when using only four labeled seed words per category. We showed that our modeling decisions – using symmetric patterns as classification features and a simple iterative k-NN algorithm for label propagation – lead to a substantial performance gain compared to state-of-the-art, more sophisticated, alternatives. Our results demonstrate the applicability of minimally supervised methods for semantic classification tasks. Future work will include modifying our model to support other, more fine-grained types of semantic categories, including adjectival categories (*properties*). We also plan to work on token-level word classification, and thus support multi-sense words, as well as demonstrate the power of unsupervised patterns acquisition for multilingual setups.

Acknowledgments

This research was funded (in part) by the Harry and Sylvia Hoffman leadership and responsibility program (for the first author), the Google Faculty research award (for the second author), the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI) and the Israel Ministry of Science and Technology Center of Knowledge in Machine Learning and Artificial Intelligence (Grant number 3-9243).

References

O. Abend and A. Rappoport. 2013. Universal conceptual cognitive annotation (UCCA). In *Proc. of ACL*.

T. Baldwin and F. Bond. 2003. A plethora of methods for learning English countability. In *Proc. of EMNLP*.

S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly. 2008. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proc. of WWW*, pages 895–904. ACM.

M. Baroni and A. Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

M. Berland and E. Charniak. 1999. Finding parts in very large corpora. In *Proc. of ACL*.

S. R. Bowman and H. Chopra. 2012. Automatic Animacy Classification. In *Proc. of NAACL-HLT Student Research Workshop*.

P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

A. M. Chan, J. M. Baker, E. Eskandar, D. Schomer, I. Ulbert, K. Marinkovic, S. S. Cash, and E. Halgren. 2011. First-pass selectivity for semantic categories in human anteroventral temporal lobe. *The Journal of Neuroscience*, 31(49):18119–18129.

E. Charniak, D. Blaheta, N. Ge, K. Hall, J. Hale, and M. Johnson. 2000. BLLIP 198789 WSJ Corpus Release 1, LDC No. LDC2000T43. Linguistic Data Consortium.

A. Clark. 2000. Inducing syntactic categories by context distribution clustering. In *Proc. of CoNLL*.

E. V. Clark. 2009. *First language acquisition*. Cambridge University Press.

R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proc. of ICML*.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537.

D. Davidov and A. Rappoport. 2006. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *Proc. of ACL-Coling*.

D. Davidov and A. Rappoport. 2008. Unsupervised discovery of generic relationships using pattern clusters and its evaluation by automatically generated SAT analogy questions. In *Proc. of ACL-HLT*.

B. J. Devereux, L. K. Tyler, J. Geertzen, and B. Randall. 2013. The centre for speech, language and the brain (CSLB) concept property norms. *Behavior research methods*, pages 1–9.

R. M. Dixon. 2005. *A semantic approach to English grammar*. Oxford University Press.

B. Dorow, D. Widdows, K. Ling, J. P. Eckmann, D. Sergi, and E. Moses. 2005. Using Curvature and Markov Clustering in Graphs for Lexical Acquisition and Word Sense Discrimination.

R. Evans and C. Orăsan. 2000. Improving anaphora resolution by identifying animate entities in texts. In *Proc. of DAARC*.

V. Hatzivassiloglou and K. R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proc. of ACL*.

M. A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of Coling – Volume 2*.

H. Ji and D. Lin. 2009. Gender and Animacy Knowledge Discovery from Web-Scale N-Grams for Unsupervised Person Mention Detection. In *Proc. of PACLIC*.

M. A. Just, V. L. Cherkassky, S. Aryal, and T. M. Mitchell. 2010. A neurosemantic theory of concrete noun representation based on the underlying brain codes. *PloS one*, 5(1):e8622.

T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proc. of ACL-HLT*.

Z. Kozareva, E. Riloff, and E. Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proc. of ACL-HLT*.

O. Y. Kwong. 2011. Measuring concept concreteness from the lexicographic perspective. In *Proc. of PACLIC*.

T. K. Landauer and S. T. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211.

R. W. Langacker. 2008. *Cognitive grammar: A basic introduction*. Oxford University Press.

P. Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

J. M. Mandler. 2004. *The foundations of mind: Origins of conceptual thought*. Oxford University Press New York.

A. Martin. 2007. The representation of object concepts in the brain. *Annual Review of Psychology*, 58:25–45.

J. B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, et al. 2011. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182.

S. Miller, J. Guinness, and A. Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proc. of NAACL*.

J. L. Moore, C. J. Burges, E. Renshaw, and W.-t. Yih. 2013. Animacy Detection with Voting Models. In *Proc. of EMNLP*.

R. Nagata, A. Kawai, K. Morihiro, and N. Isu. 2006. Reinforcing English countability prediction with one countability per discourse property. *Proc. of ACL-Coling*.

T. Naselaris, D. E. Stansbury, and J. L. Gallant. 2012. Cortical representation of animate and inanimate objects in complex natural scenes. *Journal of Physiology-Paris*, 106(5):239–249.

C. Orăsan and R. Evans. 2001. Learning to identify animate references. In *Proc. of the Workshop on Computational Natural Language*.

C. Orăsan and R. Evans. 2007. NP Animacy Identification for Anaphora Resolution. *JAIR*, 29:79–103.

L. Øvrelid. 2005. Animacy classification based on morphosyntactic corpus frequencies: some experiments with Norwegian nouns. In *Proc. of the Workshop on Exploring Syntactically Annotated Corpora*, pages 1–11.

J. Peng and K. Araki. 2005. Detecting the countability of english compound nouns using web-based models. In *Proc. of IJCNLP*.

E. Riloff and J. Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *Proc. of EMNLP*.

R. Schwartz, O. Tsur, A. Rappoport, and M. Koppel. 2013. Authorship Attribution of Micro-Messages. In *Proc. of EMNLP*.

R. Snow, D. Jurafsky, and A. Y. Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proc. of ACL-Coling*.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*.

G. Sudre, D. Pomerleau, M. Palatucci, L. Wehbe, A. Fyshe, R. Salmelin, and T. Mitchell. 2012. Tracking neural coding of perceptual and semantic features of concrete nouns. *NeuroImage*, 62(1):451–463.

P. P. Talukdar and K. Crammer. 2009. New regularized algorithms for transductive learning. In *ECML-PKDD*, pages 442–457. Springer.

O. Tsur, D. Davidov, and A. Rappoport. 2010. ICWSM – a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *Proc. of ICWSM*.

P. D. Turney and P. Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188.

P. Turney, Y. Neuman, D. Assaf, and Y. Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proc. of EMNLP*.

P. D. Turney. 2008. The latent relation mapping engine: Algorithm and experiments. *Journal of Artificial Intelligence Research*, 33:615–655.

R. C. Wang and W. W. Cohen. 2009. Automatic set instance extraction using the web. In *Proc. of ACL-IJCNLP*.

D. Widdows and B. Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proc. of Coling*.

X. Xing, Y. Zhang, and M. Han. 2010. Query difficulty prediction for contextual image retrieval. In *Advances in Information Retrieval*, pages 581–585. Springer.

S. X. Yu and J. Shi. 2003. Multiclass spectral clustering. In *Proc. of ICCV*.

Novel Word-sense Identification

Paul Cook♣, Jey Han Lau♠, Diana McCarthy♢ and Timothy Baldwin♣
♣ Department of Computing and Information Systems, The University of Melbourne
♠ Department of Philosophy, King's College London
♢ University of Cambridge
paulcook@unimelb.edu.au, jeyhan.lau@gmail.com,
diana@dianamccarthy.co.uk, tb@ldwin.net

Abstract

Automatic lexical acquisition has been an active area of research in computational linguistics for over two decades, but the automatic identification of new word-senses has received attention only very recently. Previous work on this topic has been limited by the availability of appropriate evaluation resources. In this paper we present the largest corpus-based dataset of diachronic sense differences to date, which we believe will encourage further work in this area. We then describe several extensions to a state-of-the-art topic modelling approach for identifying new word-senses. This adapted method shows superior performance on our dataset of two different corpus pairs to that of the original method for both: (a) types having taken on a novel sense over time; and (b) the token instances of such novel senses.

1 Novel word-senses

The meanings of words change over time with, in particular, established words taking on new senses. For example, the usages of *drop*, *wall*, and *blow up* in the following sentences correspond to relatively-recent senses of these words that appear to be quite common in text related to popular culture, but are not listed in many dictionaries; for example, they are all missing from WordNet 3.0 (Fellbaum, 1998).

1. *The reissue album drops March 27 and is an extension of Perry's huge 2010 Teenage Dream.* [*drops* = "comes out", "is released"]

2. *On Facebook, you can plainly see much of the data the site has on you, because it's posted to your wall.* [*wall* = "Facebook wall", "personal electronic noticeboard"]

3. *Why would I give him my number so he can blow up my phone the way he does my inbox.* [*blow up* = "overwhelm with messages"]

Computational lexicons are an essential component of systems for a variety of natural language processing (NLP) tasks. The success of such systems, therefore, depends on the quality of the lexicons they use, and (semi-)automatic techniques for identifying new word-senses could benefit applied NLP by helping to keep lexicons up-to-date. In revising dictionaries, lexicographers must identify new word-senses, in addition to new words themselves; methods which identify new word-senses could therefore also help to keep dictionaries current.

In this paper, because of the need for lexicon maintenance, we focus on relatively-new word-senses. Specifically, we consider the identification of word-senses that are not attested in a *reference corpus*, taken to represent standard usage, but that are attested in a *focus corpus* of newer texts.

Lau et al. (2012) introduced the task of novel sense identification. They presented a method for identifying novel word-senses — described here in Section 4 — and evaluated this method on a very small dataset consisting of just five lemmas having a novel sense in a single corpus pair. Cook et al. (2013) extended the method of Lau et al. to incorporate knowledge of the expected domains of new word-senses, but did not conduct a rigorous empirical evaluation. The remainder of this paper is structured

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1624–1635, Dublin, Ireland, August 23-29 2014.

as follows. After discussing related work in Section 2, we present a substantially-expanded evaluation dataset in Section 3, that is based on a second corpus pair and consists of many more lemmas with a novel sense. We describe the models used by Lau et al. and Cook et al., and our new extensions to them, in Section 4. In Section 5 we analyse the results of novel sense identification, and consider a new baseline for this task. We demonstrate that the extended methods give an improvement over the original method of Lau et al. We conclude by discussing some previously-unexplored variations on novel sense identification, and limitations of the approaches considered.

The primary contributions of this paper are: (1) development of a novel sense detection dataset much larger than has been used in research to date; (2) development and evaluation of a new baseline for novel sense detection, reformulations of the method of Lau et al., and a method that incorporates only the expected domain(s) of novel senses; (3) empirical evaluation of the method of Cook et al.; and (4) extension of the novel sense detection method of Cook et al. to automatically acquire information about the expected domain(s) of novel senses.

2 Related work

Identifying diachronic changes in word-sense is a challenge that has only been considered rather recently in computational linguistics. Sagi et al. (2009) and Cook and Stevenson (2010) propose methods to identify specific types of semantic change — widening and narrowing, and amelioration and pejoration, respectively — based on specific properties of these phenomena. Gulordava and Baroni (2011) identify diachronic sense change in an n-gram database, but using a model that is not restricted to any particular type of semantic change. Cook and Hirst (2011) consider the impact of sense frequency on methods for identifying novel senses. Crucially, all of the aforementioned approaches are type-based: they are able to identify words that have undergone a change in meaning, but not the token instances which give rise to these sense differences.

Bamman and Crane (2011) use a parallel Latin–English corpus to induce word senses and build a WSD system, which they then apply to study diachronic variation in sense frequency. Rohrdantz et al. (2011) present a system for visualizing changes in word usage over time. Crucially, in these token-based approaches there is a clear connection between (induced) word-senses and tokens, making it possible to identify usages of a specific (new) sense.

Other work has focused on sense differences between dialects and domains. Peirsman et al. (2010) consider the identification of words that are typical of Belgian and Netherlandic Dutch, due to either marked frequency or sense. McCarthy et al. (2007) consider the identification of predominant word-senses in corpora, including differences between domains. However, this approach does not identify new senses as it relies on a pre-existing sense inventory. Carpuat et al. (2013) identify words in a domain-specific parallel corpus with novel translations.

The method proposed by Lau et al. (2012), and extended by Cook et al. (2013), identifies novel word-senses using a state-of-the-art word-sense induction (WSI) system. This token-based approach offers a natural account of polysemy and not only identifies word types that have a novel sense, but identifies the token instances of the hypothesized novel senses, without reliance on parallel text or a pre-existing sense inventory. We therefore adopt this method for evaluation on our new dataset, and propose further extensions to this method.

3 Datasets

Evaluating approaches to identifying semantic change is a challenge due to the lack of appropriate evaluation resources (i.e., corpora for the appropriate time periods, known to exhibit particular sense changes); indeed, most previous approaches have used very small datasets (e.g., Sagi et al., 2009; Cook and Stevenson, 2010; Bamman and Crane, 2011). In this study we consider two datasets of relatively newly-coined word-senses: (1) an extended version of the dataset based on the BNC (Burnard, 2000) and ukWaC (Ferraresi et al., 2008) used by Lau et al. (2012); and (2) a new dataset based on the SiBol/Port Corpus.[1] This

[1] http://www3.lingue.unibo.it/blog/clb/?page_id=8

is the largest dataset for evaluating approaches to identifying diachronic semantic change constructed from corpus evidence to be presented to date.

3.1 BNC–ukWaC

Lau et al. (2012) take the written portion of the BNC (approximately 87 million words of British English from the late 20th century) as the reference corpus, and a similarly-sized random sample of documents from the ukWaC (a Web corpus built from the `.uk` domain in 2007) as the focus corpus. They used TreeTagger (Schmid, 1994) to tokenise and lemmatise both corpora.

A set of words that has acquired a new sense between the late 20th and early 21st centuries — the time periods of the reference and focus corpora — is required. The Concise Oxford English Dictionary aims to document contemporary usage, and has been published in numerous editions including Thompson (1995, COD95) and Soanes and Stevenson (2005, COD08), enabling the identification of new senses amongst the entries in COD08 relative to COD95. Manually searching these dictionaries for new senses would be time intensive, but new words often correspond to concepts that are culturally salient (Ayto, 2006), and one can leverage this observation to speed up the process of finding some candidate words with novel senses.[2]

Between the time periods of the reference and focus corpora, computers and the Internet have become much more mainstream in society. Lau et al. therefore extracted all headwords in COD08 whose entries contain the word *computing*. They then carefully annotated these lemmas to identify those that indeed exhibit the novel sense indicated in the dictionary in the corpora. Here, we expand Lau et al.'s dataset by extracting all headwords including any of the following words *code*, *computer*, *internet*, *network*, *online*, *program*, *web*, and *website*. We then follow a similar annotation process to Lau et al.

An annotator read the entries for the selected lexical items in COD95 and COD08, and identified those which have a clear sense related to computers or the Internet in COD08 that is not present in COD95; such senses are referred to as *novel senses*. This process, along with all the annotation in this section (including Section 3.2), is carried out by native English-speaking authors of this paper and graduate students in computational linguistics.

To ensure that the words identified from the dictionaries do in fact have a new sense in the ukWaC sample compared to the BNC, we examine word sketches (Kilgarriff et al., 2004)[3] for each of these lemmas in the BNC and ukWaC for collocates that likely correspond to the novel sense; we exclude any lemma for which we find evidence of the novel sense in the BNC, or fail to find evidence of the novel sense in the ukWaC sample.[4]

We further examine the usage of these words in the corpora. We extract a random sample of 100 usages of each lemma from the BNC and ukWaC sample, and annotate these usages as to whether they correspond to the novel sense or not. This binary distinction is easier than fine-grained sense annotation, and since we do not use these annotations for formal evaluation — only for selecting items for our dataset — we do not carry out an inter-annotator agreement study here. We eliminate any lemma for which we find evidence of the novel sense in the usages from the BNC, or for which we do not find evidence of the novel sense in the ukWaC sample usages.[5]

This process resulted in the identification of two lemmas not in the dataset of Lau et al., with frequency greater than 1000 in the ukWaC sample, and having a novel sense in the ukWaC compared to the BNC (*feed* (n) and *visit* (v)). Combining these new lemmas with the dataset of Lau et al. gives an expanded dataset consisting of seven lemmas. For both of the two new lemmas, a second annotator annotated the sample of 100 usages from the ukWaC. The observed agreement and unweighted Kappa for this annotation task for all seven lemmas is 97.4% and 0.93, respectively, indicating that this is indeed a relatively easy annotation task. The annotators discussed the small number of disagreements to reach

[2]We access the dictionaries in the same way as Lau et al., namely we use COD08 online via `http://oxfordreference.com`, and the paper version of COD95.

[3]`http://www.sketchengine.co.uk/`

[4]We examine word sketches for the full ukWaC because this version of the corpus is available through the Sketch Engine.

[5]We use the IMS Open Corpus Workbench (`http://cwb.sourceforge.net/`) to extract usages of our target lemmas from the corpora. This extraction process fails in a number of cases, and so we also eliminate such items from our dataset.

BNC–ukWaC

Lemma	Frequency	Novel sense definition
domain (n)	41	Internet domain
export (v)	28	export data
feed (n)	23	data feed
mirror (n)	10	mirror website
poster (n)	4	one who posts online
visit (v)	28	access a website
worm (n)	30	malicious program

SiBol/Port

Lemma	Frequency	Novel sense definition
cloud (n)	9	Internet-based computational resources
drag (v)	1	move on a computer screen using a mouse
follower (n)	34	Twitter follower
help (n)	1	displayed instructions, e.g., help menu
hit (n)	2	search hit
platform (n)	22	computing platform
poster (n)	5	one who posts online
reader (n)	3	e-reader
rip (v)	1	copy music
site (n)	39	website
text (n)	39	text message
visit (v)	7	access a website
wall (n)	2	Facebook wall

Table 1: Lemmas in the BNC–ukWaC and SiBol/Port datasets. For each lemma, the frequency of its novel sense in the annotated sample of usages from the focus corpus, and a definition of its novel sense, are shown.

consensus. The seven lemmas in this dataset are shown in Table 1, along with definitions of their novel senses, and the frequencies of their novel senses in the focus corpus.

Lau et al. compared the novelty of the lemmas with a novel sense to that of a same-size set of distractor lemmas not having a novel sense. Here we consider a much larger set of 50 distractors — 25 nouns and 25 verbs — randomly sampled from a similar frequency range as the items with a novel sense.

One shortcoming of this dataset (and indeed the subset of it used by Lau et al.) is that text types are represented to different extents in the BNC and ukWaC, with, for example, texts related to the Internet being much more common in the ukWaC. Such differences in corpus composition are a noted challenge for approaches to identifying lexical semantic differences between corpora (Peirsman et al., 2010). In the following subsection we therefore consider the creation of a new dataset from more-comparable corpora.

3.2 SiBol/Port

The SiBol/Port Corpus consists of texts from several British newspapers for the years 1993, 2005, and 2010; we use the 1993 and 2010 portions of this corpus — referred to as SP1993 and SP2010 — as our reference and focus corpora, respectively. SP1993 and SP2010 contain approximately 93M and 99M words, respectively. In contrast to BNC–ukWaC, our reference and focus corpora are now comparable, in that they both consist of texts from British newspapers but they differ with respect to the specific year.

The novel word-senses in the BNC–ukWaC dataset are all related to computers and the Internet, but there has been recent lexical semantic change unrelated to technology as well (e.g., *sick* can be used to mean "excellent"). In an effort to include such non-technical novel senses in this new dataset, we obtain a list of headwords for which a sense was added to the Macmillan English Dictionary for Advanced

Learners (MEDAL)[6] since its first edition (Rundell and Fox, 2002), courtesy of Macmillan Dictionaries. Beginning with these candidates from MEDAL, and the items extracted from COD from Section 3.1, we discard any lemma whose frequency is less than 1000 in SP1993 or SP2010.

As for the BNC–ukWaC dataset, an annotator examined word sketches for these lemmas. However, it is possible that the novel sense for a lemma is present in a corpus, but that we fail to find evidence for it in that lemma's word sketch. We therefore also obtain judgements from two annotators as to whether each novel sense is expected to be very infrequent (or unattested) in SP2010. To reduce subsequent annotation effort, we discard any lemma for which its novel sense is believed to be infrequent in SP2010 by both judges, and is not found in the word sketch from SP2010.

Annotators then annotate a random sample of 100 usages of each lemma in the reference and focus corpora as before, and again eliminate any lemma for which we find evidence of its novel sense in the reference corpus, or fail to find evidence of that sense in the focus corpus. We identify thirteen lemmas having a novel sense in SP2010 relative to SP1993. These lemmas are also shown in Table 1.

We obtain a second set of annotations for the usages of these lemmas in the sample from SP2010, with each lemma being annotated by a different annotator than before. The observed agreement and unweighted Kappa between the two sets of annotations is 96.2% and 0.81, respectively. In cases of disagreement, a final annotation is again reached through discussion.

We randomly select 164 lemmas (116 nouns and 48 verbs) from a similar frequency range as the lemmas having a novel sense, to serve as distractors.

Both the BNC–ukWaC and SiBol/Port datasets have been made available.[7]

4 The WSI-based approach to novel word-sense detection

In this section we describe the WSI-based method of Lau et al. (2012) for detecting novel senses, and an extension of this method from Cook et al. (2013). We then present new extensions of this method.

The Lau et al. (2012) WSI model is based on a Hierarchical Dirichlet Process (HDP, Teh et al., 2006), which is a non-parametric variant of a topic model that, like the commonly-used Latent Dirichlet Allocation (LDA, Blei et al., 2003), learns topics (in the form of multinomial probability distributions over words) and per-document topic assignments (in the form of multinomial probability distributions over topics) for a collection of documents; unlike LDA, however, it also optimises the number of topics in an unsupervised data-driven manner. In the context of WSI, by creating "documents" that consist of sentences containing a target word, we can view the topics learnt by topic models as the sense representation of the target word. Indeed, topic models have been previously applied to WSI (e.g., Brody and Lapata, 2009; Yao and Van Durme, 2011).

To generate the input for the topic model, the documents are tokenised (in this case, a "document" is a short context, typically 1–3 sentences, containing a target word) into a bag of words. All words are lemmatised, and stopwords and low frequency terms are removed. Positional word features — commonly used in WSI — for each of the three words to the left and right of the target word are also included.

To induce the senses of a target word w from a given set of usages of w, HDP is run on those usages (represented according to the features described above) to induce topics; these topics are then interpreted as representing the senses of w (one topic per sense). To determine the sense assigned to each instance, the system aggregates over the topic assignments for each word in the context of w, and selects the topic with the highest aggregated probability, i.e., $\text{argmax}_z P(t = z|d)$, where d is a document and t is a topic.

Recently, Lau et al. (2013a,b) found this method to give the overall best performance on two WSI shared tasks (Jurgens and Klapaftis, 2013; Navigli and Vannella, 2013), demonstrating that the method is competitive with the state-of-the-art in WSI, and appropriate as the basis for a method for identifying novel word-senses.

[6] http://www.macmillandictionary.com/

[7] http://www.csse.unimelb.edu.au/~tim/etc/novel-sense-dataset.tgz

4.1 Novel Sense Detection

Following Lau et al. (2012), to detect novel senses of a target word using this WSI method, we *jointly* topic model two corpora: a reference corpus — taken to represent standard usage — and a focus corpus of newer texts potentially containing novel senses. In other words, we extract usages of a target word w from both corpora, and then topic model the pooled instances of w. Under this approach, the discovered topics are applicable to both corpora, so there is no need to reconcile two different sets of topics. For the experiments in this paper, we extract three sentences of context for each usage, one sentence to either side of the usage of the target word.

As each usage is given a sense assignment, we can identify novel senses — senses present in the focus corpus, but unattested in the reference corpus — based on differences in the sense distribution for a given word between the two corpora. Lau et al. present a Novelty score which is proportional to the following:

$$\text{Novelty}_{\text{Ratio}}(s) = \frac{p_f(s)}{p_r(s)} \tag{1}$$

where $p_f(s)$ and $p_r(s)$ are the proportion of usages of a given word corresponding to sense s in the focus corpus and reference corpus, respectively, calculated using smoothed maximum likelihood estimates. The score for a given lemma is the maximum score for any of its induced senses. We refer to the *novel sense* for a lemma as the induced sense corresponding to this maximum.

4.2 Alternative Formulations of Novelty

The WSI system underlying the approach of Lau et al. labels each usage of a target lemma with an induced sense. Therefore, any approach to identifying keywords — words that are substantially more frequent in one corpus than another — can potentially be applied to identify novel senses, by viewing "words" as (word,sense) tuples. We consider a version of Novelty based on the difference in relative frequency of an induced sense in the focus and reference corpora, as below:

$$\text{Novelty}_{\text{Diff}}(s) = p_f(s) - p_r(s) \tag{2}$$

We consider a further new variant of Novelty based on the log-likelihood ratio of an induced sense in the two corpora, referred to as $\text{Novelty}_{\text{LLR}}$.

4.3 Incorporating knowledge of expected topics of novel senses

Cook et al. (2013) extended Lau et al.'s method by incorporating the observation that many neologisms are related to topics that are culturally salient (e.g., Ayto, 2006); nowadays we see many neologisms related to computing and the Internet. Indeed this observation was used to construct the gold-standard dataset for this study. Cook et al. identified a set of words, W, related to computing and the Internet, based on manual analysis of keywords for the corpora they considered. They then formulated the Relevance of an induced sense s for a given word as follows:

$$\text{Relevance}_{\text{Manual}}(s) = \sum_{w \in W} p(w|s) \tag{3}$$

For a given lemma, $\text{Relevance}_{\text{Manual}}$ is the maximum of this score for any of its induced senses, similar to Novelty.

Following Cook et al., we calculate Relevance and Novelty for each induced sense of each lemma, and then rank all the induced senses by these measures independently. We then compute the rank sum of each induced sense of each lemma under these two rankings. The final score for a given lemma is then the rank sum of its highest-ranked sense, and this sense is taken as that lemma's novel sense. We refer to this new method as "Rank Sum". Cook et al. only considered Novelty and Rank Sum; here we additionally consider Relevance on its own.

For the keywords, we manually construct a set of words related to computing and the Internet, the topics for which we expect to observe many novel senses in both of our datasets, in a similar way to Cook et al. In order to minimize annotation effort, we concentrate on words that are more-frequent in the

focus corpus than the reference corpus. For a given corpus pair, we begin by computing the keywords for those corpora using Kilgarriff's (2009) method.[8] Two annotators — both computational linguists and not authors of this paper — independently scanned the top-1000 keywords for the focus corpus, and selected those that were, based on their intuition, related to computing and the Internet. We then took the topically-relevant words for a given corpus pair to be those in the intersection of the sets of words selected by the two annotators. For BNC–ukWaC and SiBol/Port this gives 102 and 30 topically-relevant words, respectively. This annotation required, on average, 23 minutes per annotator per corpus pair to complete. Examples of the keywords selected for SiBol/Port include *broadband*, *click*, *device*, *online*, and *tweet*.

4.4 Automatically-extracting keywords

We propose a new fully-automated method for identifying a set of topically-relevant keywords. Because of the differences in corpus composition, the BNC–ukWaC keywords are often related to computing and the Internet. To automatically obtain topically-relevant words, we take the top-1000 keywords for the ukWaC relative to the BNC (i.e., the same keywords annotated for the BNC–ukWaC in Section 4.3). The keywords for SiBol/Port are less-clearly related to the topics of interest, so we therefore use the topically-relevant keywords from BNC–ukWaC for both datasets.

5 Results

In the following subsections we consider results at the type and then token level.

5.1 Type-level results

In these experiments we rank all items — lemmas with a novel sense, and distractors — by the various Novelty, Relevance and Rank Sum methods for the BNC–ukWaC and SiBol/Port datasets. When a lemma takes on a new sense, it might also increase in frequency. We therefore also consider a baseline in which we rank the lemmas by the ratio of their frequency in the focus corpus and the reference corpus. This baseline has not been previously considered by Lau et al. (2012) or Cook et al. (2013).

To compare approaches, we examine precision–recall curves in Figures 1 and 2. In an applied setting, we envision these ranked lists being manually examined; we are therefore primarily interested in the highly-ranked items, i.e., the left portion of the precision–recall curves.

For BNC–ukWaC (Figure 1), $\text{Novelty}_{\text{Diff}}$ and $\text{Novelty}_{\text{Ratio}}$ perform much better than $\text{Novelty}_{\text{LLR}}$, but not better than the frequency ratio baseline, at least for the left-most portion of the precision–recall curve. Surprisingly, for Relevance, $\text{Relevance}_{\text{Auto}}$ outperforms $\text{Relevance}_{\text{Manual}}$. This could be because the focus corpus exhibits a clear topical bias towards computing and the Internet (the expected domain of many neologisms in the focus corpus), and therefore a larger set of potentially noisy keywords is more informative than a smaller, hand-selected set. All of the measures including the baseline, except for $\text{Novelty}_{\text{LLR}}$, assign higher scores to lemmas with a gold-standard novel sense than the distractors, according to a one-sided Wilcoxon rank sum test ($p < 0.05$ in each case).

Turning to SiBol/Port in Figure 2, the frequency ratio baseline is much less effective here; the frequency of the gold-standard novel senses is much lower overall than for BNC–ukWaC. All of the Novelty and Relevance methods outperform the baseline, and — with the exception of $\text{Novelty}_{\text{Ratio}}$ — rank the lemmas with a gold-standard novel sense higher than the distractors (again using a one-sided Wilcoxon rank sum test and $p < 0.05$). Furthermore, in this case, $\text{Relevance}_{\text{Manual}}$ outperforms $\text{Relevance}_{\text{Auto}}$, as expected.

In terms of the three Novelty measures, only $\text{Novelty}_{\text{Diff}}$ ranked items with a novel sense higher than the distractors for both datasets. We therefore also show results for the Rank Sum approach combining $\text{Novelty}_{\text{Diff}}$ and each of $\text{Relevance}_{\text{Manual}}$ and $\text{Relevance}_{\text{Auto}}$, denoted $\text{Rank Sum}_{\text{Diff,manual}}$ and $\text{Rank Sum}_{\text{Diff,auto}}$, respectively, in Figures 1 and 2. For both BNC–ukWaC and SiBol/Port, $\text{Rank Sum}_{\text{Diff,manual}}$

[8] Using this method, the keywordness score for a given word is simply the ratio of its frequency per million words, plus a constant, in two corpora; we set the constant to 100, the value recommended by Kilgarriff.

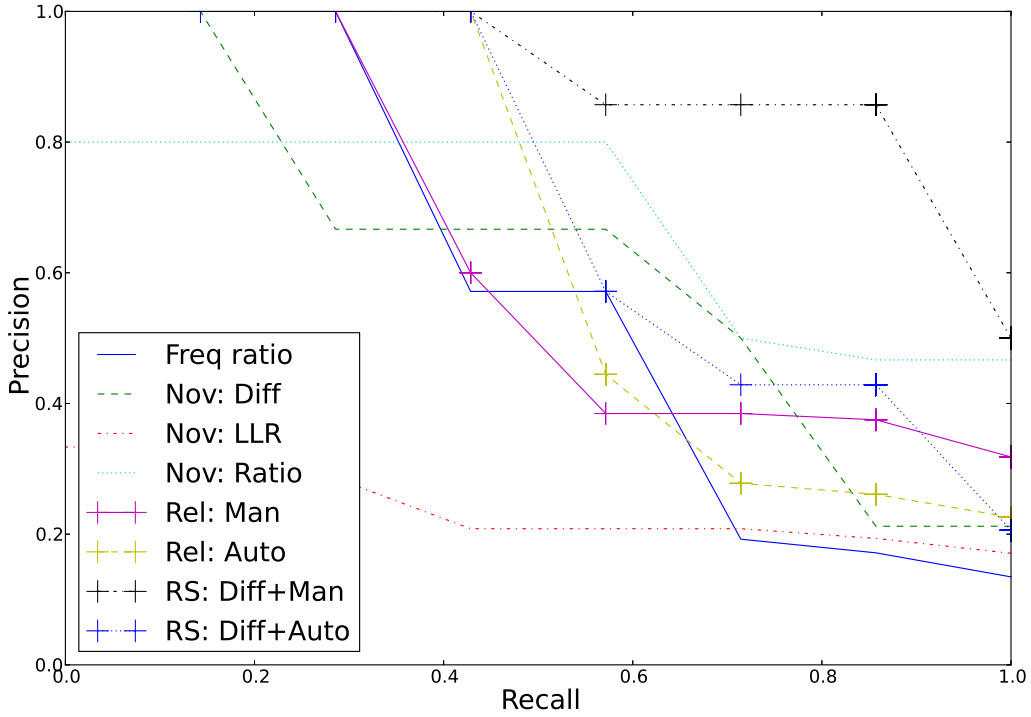


Figure 1: Precision–recall curve for the BNC–ukWaC dataset.

gives the best performance, and is a clear improvement over either of the individual methods. As expected, the performance of Rank $\text{Sum}_{\text{Diff,auto}}$ is not as good, but is nevertheless an improvement over the frequency ratio baseline for both datasets and provides an alternative to manual scrutiny of the keywords.

To further examine the potential of incorporating knowledge of the expected domains of novel senses to improve novel sense identification, we consider the case of *cloud* (n) from the SiBol/Port dataset. The highest-probability words for the topic with highest $\text{Novelty}_{\text{Diff}}$ are the following: *ash*, *volcanic*, *flight*, *@card@*,[9] *travel*, *airline*, *volcano*, *airport*, *air*, *cloud*. This sense appears to be related to the eruption of the Eyjafjallajökull volcano, a major event in 2010 (the year from which the SiBol/Port focus corpus is taken). Such topical differences, which do not correspond to a novel sense, are a problem for any approach to identifying lexical semantic differences between two corpora based on differences in the lexical context of a target word, and indeed observations such as this motivated our use of the methods incorporating Relevance. The highest probability words for the topic with highest $\text{Relevance}_{\text{Auto}}$ are the following: *cloud*, *@card@*, *company*, *service*, *business*, *computing*, *market*, *security*, *datum*, *need*. This topic appears to correspond to the expected novel sense of Internet-based computational resources, demonstrating the potential to improve a system for identifying novel word-senses by incorporating knowledge of the expected domains of neologisms. Moreover, incorporating Relevance is particularly powerful for avoiding false positives. For example, the distractor *clause* (n) is the lemma with the sixth-highest $\text{Novelty}_{\text{Diff}}$ for SiBol/Port. The highest probability words for the corresponding topic are the following: *contract*, *@card@*, *club*, *player*, *million*, *england*, *capello*, *manager*, *sign*, *deal*. This induced sense appears to be related to clauses in Fabio Capello's contract as manager of the England national football team, and is not a novel sense of *clause*. However, none of the induced senses of *clause* have high $\text{Relevance}_{\text{Auto}}$ or $\text{Relevance}_{\text{Manual}}$, and so incorporating information from Relevance can avoid incorrectly identifying this lemma as having a novel sense.

[9] A generic token signifying a cardinal number.

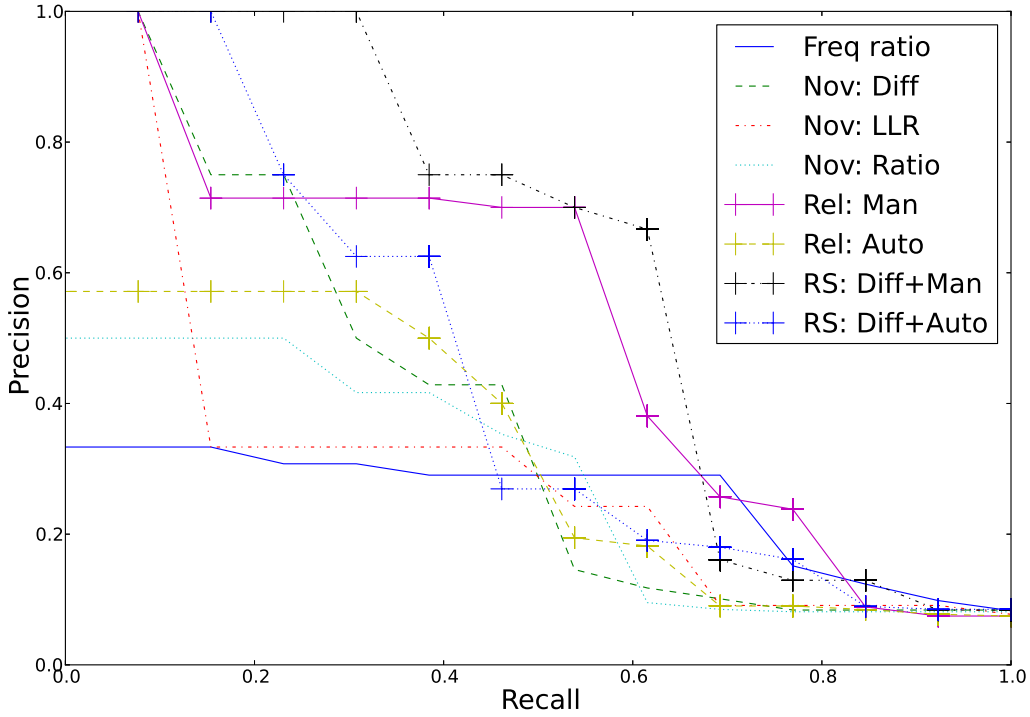


Figure 2: Precision–recall curve for the SiBol/Port dataset.

5.2 Token-level results

In this section, we consider the token-level identification of instances of the gold-standard novel senses. We compare Novelty, Relevance, and Rank Sum to a baseline that assigns all usages of a lemma to a single topic which is selected as the novel sense; in this case recall is 1, and precision is proportional to the frequency of the novel sense. We further consider the theoretical upper-bound of a method which selects a single topic as the novel sense, based on the output of the HDP-based WSI method; this oracle selects the best topic in terms of F-score as the novel sense. Results are presented in Table 2.

Each variant of Novelty and Relevance is an improvement over the baseline, although the Relevance measures don't perform as well as the Novelty ones, despite this dataset only containing novel senses related to computing (despite our efforts to include non-technical novel senses). For consistency with the presentation of the type-level results, we again consider Rank Sum using $\text{Novelty}_{\text{Diff}}$, even though it doesn't perform as well as $\text{Novelty}_{\text{LLR}}$ or $\text{Novelty}_{\text{Ratio}}$ on BNC–ukWaC. Using either automatically- or manually-obtained keywords, the performance of Rank Sum on BNC–ukWaC is remarkably on par with the upper-bound, although for SiBol/Port there is little or no improvement over $\text{Novelty}_{\text{Diff}}$. Nevertheless, these findings are further indication that novel sense identification can be improved by incorporating information about the topics for which we expect to see novel senses. However, this approach is particularly helpful at the type-level, where information about the expected topics of novel senses prevents lemmas not having a novel sense (i.e., the distractors) from being assigned high novelty.

6 Discussion and conclusion

The methods considered in this paper could be applied to any corpus pair, and potentially to identify lexical semantic differences between, for example, domains or language varieties. The focus of this study is English; sufficiently-large comparable corpora of national varieties of English (e.g., British and American English), are not readily-available, but could potentially be inexpensively constructed in the future (Cook and Hirst, 2012). We conducted some preliminary experiments using domain-specific sports

Method	F-score	
	BNC–ukWaC	SiBol/Port
$\text{Novelty}_{\text{Diff}}$	0.57	0.29
$\text{Novelty}_{\text{LLR}}$	0.67	0.28
$\text{Novelty}_{\text{Ratio}}$	0.66	0.28
$\text{Relevance}_{\text{Auto}}$	0.48	0.24
$\text{Relevance}_{\text{Manual}}$	0.45	0.27
$\text{Rank Sum}_{\text{Diff,auto}}$	0.72	0.30
$\text{Rank Sum}_{\text{Diff,manual}}$	0.72	0.29
Upper-bound	0.72	0.42
Baseline	0.36	0.20

Table 2: Token-level F-score for the BNC–ukWaC and SiBol/Port datasets using variants of Novelty, Relevance, and Rank Sum. The F-score of an oracle upper-bound and baseline are also shown.

and finance corpora (Koeling et al., 2005) and the BNC. However, in these experiments we observed very high $\text{Novelty}_{\text{Ratio}}$ for many distractors (selected in a similar way to our other experiments). Unlike the case of time difference, in corpora from different domains, an arbitrarily chosen word will tend to cooccur with very different words in the corpora, and $\text{Novelty}_{\text{Ratio}}$ will consequently be high. To address vocabulary differences between corpora, in their experiments on identifying lexical semantic differences between Dutch dialects, Peirsman et al. (2010) restricted the context words used to represent a target word to those with moderate frequency in each of the two corpora used. We considered a similar restriction in experiments on SiBol/Port, but did not see an overall improvement in performance.

We demonstrated that the performance of a method for identifying novel word-senses can be improved by incorporating information — acquired manually or automatically — about the expected topics of novel senses, which tend to be related to culturally-salient concepts. In future work, we intend to consider improved approaches for automatically identifying topically-relevant words by incorporating information about the top keywords of a corpus harvested from the Web for the domain of interest (e.g., PVS et al., 2012). We also believe that topic models could be useful for identifying emerging or changing domains themselves given the reference and focus corpus, and related work in this area (e.g., Wang and McCallum, 2006; Blei and Lafferty, 2007).

To conclude, we have presented the largest type- and token-level dataset of diachronic sense differences to date, drawing on two pairs of corpora, and have made this dataset available. We applied a recently-proposed WSI-based method to the task of finding sense differences in this data. We demonstrated that, while the method shows promise, on a type-based task it is comparable to a a simple frequency baseline, which had not been previously considered for this task. We carried out the first empirical evaluation of a recently-proposed extension of this method that incorporates manually-acquired knowledge of the expected domains of new senses, and found it to have superior performance at both the type and token level. We further proposed and evaluated an approach that only uses this domain knowledge, and a method for automating its acquisition.

Acknowledgments

We thank Michael Rundell and Macmillan Dictionaries for providing the list of headwords added to MEDAL since its first edition, and Charlotte Taylor for providing us with early access to SiBol/Port. We also thank Richard Fothergill, Karl Grieser, and Andrew Mackinlay for their help in annotation. This research was supported in part by funding from the Australian Research Council.

References

John Ayto. 2006. *Movers and Shakers: A Chronology of Words that Shaped our Age*. Oxford University Press, Oxford.

David Bamman and Gregory Crane. 2011. Measuring historical word sense variation. In *Proceedings of the 2011 Joint International Conference on Digital Libraries (JCDL 2011)*, pages 1–10. Ottawa, Canada.

D. Blei, A. Ng, and M. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

David M. Blei and John D. Lafferty. 2007. Latent dirichlet allocation. *The Annals of Applied Statistics*, 1(1):17–35.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the EACL (EACL 2009)*, pages 103–111. Athens, Greece.

Lou Burnard. 2000. *User Reference Guide for the British National Corpus*. Technical report, Oxford University Computing Services.

Marine Carpuat, Hal Daumé III, Katharine Henry, Ann Irvine, Jagadeesh Jagarlamudi, and Rachel Rudinger. 2013. SenseSpotting: Never let your parallel data tie you to an old domain. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 1435–1445. Sofia, Bulgaria.

Paul Cook and Graeme Hirst. 2011. Automatic identification of words with novel but infrequent senses. In *Proceedings of the 25th Pacific Asia Conference on Language Information and Computation (PACLIC 25)*, pages 265–274. Singapore.

Paul Cook and Graeme Hirst. 2012. Do Web corpora from top-level domains represent national varieties of English? In *Actes des 11es Journées Internationales d'Analyse Statistique des Données Textuelles / Proceedings of the 11th International Conference on Textual Data Statistical Analysis*, pages 281–293. Liège, Belgium.

Paul Cook, Jey Han Lau, Michael Rundell, Diana McCarthy, and Timothy Baldwin. 2013. A lexicographic appraisal of an automatic approach for detecting new word-senses. In *Electronic lexicography in the 21st century: thinking outside the paper. Proceedings of the eLex 2013 conference*, pages 49–65. Tallinn, Estonia.

Paul Cook and Suzanne Stevenson. 2010. Automatically identifying changes in the semantic orientation of words. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*, pages 28–34. Valletta, Malta.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, USA.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop: Can we beat Google*, pages 47–54. Marrakech, Morocco.

Kristina Gulordava and Marco Baroni. 2011. A distributional similarity approach to the detection of semantic change in the Google Books Ngram corpus. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 67–71. Edinburgh, Scotland.

David Jurgens and Ioannis Klapaftis. 2013. Semeval-2013 task 13: Word sense induction for graded and non-graded senses. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 290–299. Atlanta, USA.

Adam Kilgarriff. 2009. Simple maths for keywords. In *Proceedings of the Corpus Linguistics Conference*. Liverpool, UK.

Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell. 2004. The Sketch Engine. In *Proceedings of the Eleventh EURALEX International Congress (EURALEX 2004)*, pages 105–116. Lorient, France.

Rob Koeling, Diana McCarthy, and John Carroll. 2005. Domain-specific sense distributions and predominant sense acquisition. In *Proceedings of Human Language Technology Conference and Conference*

on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005), pages 419–426. Vancouver, Canada.

Jey Han Lau, Paul Cook, and Timothy Baldwin. 2013a. unimelb: Topic modelling-based word sense induction. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 307–311. Atlanta, USA.

Jey Han Lau, Paul Cook, and Timothy Baldwin. 2013b. unimelb: Topic modelling-based word sense induction for web snippet clustering. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 217–221. Atlanta, USA.

Jey Han Lau, Paul Cook, Diana McCarthy, David Newman, and Timothy Baldwin. 2012. Word sense induction for novel sense detection. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 591–601. Avignon, France.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2007. Unsupervised acquisition of predominant word senses. *Computational Linguistics*, 33(4):553–590.

Roberto Navigli and Daniele Vannella. 2013. SemEval-2013 task 11: Word sense induction and disambiguation within an end-user application. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 193–201. Atlanta, USA.

Yves Peirsman, Dirk Geeraerts, and Dirk Speelman. 2010. The automatic identification of lexical variation between language varieties. *Natural Language Engineering*, 16(4):469–491.

Avinesh PVS, Diana McCarthy, Dominic Glennon, and Jan Pomikálek. 2012. Domain specific corpora from the Web. In *Proceedings of the 15th Euralex International Congress*, pages 336–342. Oslo, Norway.

Christian Rohrdantz, Annette Hautli, Thomas Mayer, Miriam Butt, Daniel A. Keim, and Frans Plank. 2011. Towards tracking semantic change by visual analytics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011)*, pages 305–310. Portland, USA.

Michael Rundell and Gwyneth Fox, editors. 2002. *Macmillan English Dictionary for Advanced Learners*. Macmillan Education, Oxford, UK.

Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2009. Semantic density analysis: Comparing word meaning across time and space. In *Proceedings of the EACL 2009 Workshop on GEMS: GEometrical Models of Natural Language Semantics*, pages 104–111. Athens, Greece.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49. Manchester, UK.

Catherine Soanes and Angus Stevenson, editors. 2008. *The Concise Oxford English Dictionary*. Oxford University Press, Oxford, UK, eleventh (revised) edition. Oxford Reference Online.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581.

Della Thompson, editor. 1995. *The Concise Oxford Dictionary of Current English*. Oxford University Press, Oxford, UK, ninth edition.

Xuerei Wang and Andrew McCallum. 2006. Topics over time: A non-Markov continuous-time model of topical trends. In *Proceedings of the Eleventh International Conference on Knowledge Discovery and Data Mining*, pages 424–433. Philadelphia, USA.

Xuchen Yao and Benjamin Van Durme. 2011. Nonparametric Bayesian word sense induction. In *Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing*, pages 10–14. Portland, USA.

Learning to Summarise Related Sentences

Emmanouil Tzouridis[†]
[†] Dep. of Computer Science
TU Darmstadt
Germany
{tzouridis,brefeld}@kma.informatik.tu-darmstadt.de

Jamal Abdul Nasir
Dep. of Computer Science
LUMS Lahore
Pakistan
jamaln@lums.edu.pk

Ulf Brefeld[†‡]
[‡] Inform. Center f. Education
DIPF Frankfurt/Main
Germany
brefeld@dipf.de

Abstract

We cast multi-sentence compression as a structured prediction problem. Related sentences are represented by a word graph so that summaries constitute paths in the graph (Filippova, 2010). We devise a parameterised shortest path algorithm that can be written as a generalised linear model in a joint space of word graphs and compressions. We use a large-margin approach to adapt parameterised edge weights to the data such that the shortest path is identical to the desired summary. Decoding during training is performed in polynomial time using loss augmented inference. Empirically, we compare our approach to the state-of-the-art in graph-based multi-sentence compression and observe significant improvements of about 7% in ROUGE F-measure and 8% in BLEU score, respectively.

1 Introduction

Automatic text summarisation is one of oldest forms of natural language processing (Luhn, 1958; Baxendale, 1958). The goal is to extract the most important part of the content from either a single document or a collection of documents (Mani, 2001; Roussinov and Chen, 2001; McKeown *et al.*, 2005).

Frequently, the information of interest is contained in only a part of a sentence or may be distributed across parts of several sentences. Identifying the content carrying part(s) constitutes an essential technique not only for single- and multi-document extractive summarisation but also text simplification in general. Generating a simplified version of a text traditionally has many applications in question answering (Hermjakob *et al.*, 2002) and speech synthesis (Kaji *et al.*, 2004). Due to limited display sizes of mobile devices, recent applications also deal with summarising/simplifying news articles, social media, emails, or websites (Corston-Oliver, 2001).

Multi-sentence compression (MSC) unifies many of the mentioned characteristics and challenges and can be seen as a key to text summarisation and simplification (Jing and McKeown, 2000). The task in multi-sentence compression is to map a collection of related sentences to a grammatical short sentence that preserves the most important part of the content. Sentence compression methods have been devised using manually crafted rules (Dorr *et al.*, 2003), language models (Hori *et al.*, 2003; Clarke and Lapata, 2008), or syntactical representations (Barzilay and Lee, 2003; Galley and McKeown, 2007; Filippova and Strube, 2008). Filippova (2010) introduces an elegant graph-based approach to multi-sentence compression that simply relies on the words of the sentences and efficient dynamic programming. Her approach implements the observation that the frequency of words influences their appearance in human summaries (Nenkova *et al.*, 2006). Although being an intuitive rule that does work well in practice, frequency-based strategies often remain heuristic.

In this paper we propose a structured learning-based approach to multi-sentence compression. In analogy to Filippova (2010), related sentences are represented by a word graph (the *input*). Words are identified with vertices and directed edges connect adjacent words in at least one sentence, so that the summarising sentence (the *output*) is contained as a path in the graph. Generally, learning mappings between complex structured and interdependent inputs and outputs challenges the standard model of

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1636–1647, Dublin, Ireland, August 23-29 2014.

learning a mapping from independently drawn instances to a small set of labels. To capture the involved dependencies we represent input graphs $\mathcal{G}$ and output paths p jointly by a (possibly rich) feature representation $\Phi(\mathcal{G}, p)$. The goal is to find a linear function $f(\mathcal{G}, p) = \lambda^\top \Phi(\mathcal{G}, p)$ in joint space such that

$$p = \operatorname*{argmin}_{\tilde{p}} f(\mathcal{G}, \tilde{p}) \tag{1}$$

is the desired summary for the collection $\mathcal{G}$. Our approach can therefore be seen as translating the work by Filippova (2010) into the structured prediction framework (Tsochantaridis *et al.*, 2005; Taskar *et al.*, 2004). Instead of applying heuristics, we adapt the decoding machinery to the data by parameterising a shortest path algorithm. The latter admits a representation as a generalised linear model in joint input output space. We devise a structural support vector machine (SVM) (Tsochantaridis *et al.*, 2005) to learn the shortest path in possibly high dimensional joint feature spaces and propose a generalised, loss-augmented decoding algorithm that is solved exactly by an integer linear program in polynomial time. Empirically, we evaluate the structural support vector machine on a real world news headline summarisation task. Our experiments show that a very rudimentary set of five features already suffices to significantly improve the state-of-the-art in graph-based multi-sentence compression. We observe an increase of 7% in in ROUGE F-measure and 8% in BLEU score, respectively.

The remainder of the paper is organised as follows. Section 2 reviews related work and Section 3 introduces word graphs and shortest paths. Our technical contribution is presented in Section 4. We report on empirical results in Section 5 and Section 6 concludes.

2 Related Work

The goal of automatic text summarisation is to produce a summary of a given text (or text collection) that preserves the most important information (Luhn, 1958; Edmundson, 1969). Summarisation systems usually rely on clues or features that help to identify key elements such as the main topic of a document (Salton *et al.*, 1994). Such features may be extracted from sentences (e.g., the length of a sentence, its position in the text), words (e.g., frequency of a word, relative position in sentence) as well as from style and structure elements (Kupiec *et al.*, 1995; Teufel and Moens, 1997; Marcu, 1997).

A special case of text summarisation is sentence compression; given a sentence, the task is to produce a summary of the input that preserves the most important information and is grammatically correct (Jing, 2000). Sentence compression is thus relevant for many NLP tasks including question answering, machine translation, text simplification, speech synthesis applications and multi-sentence compression (e.g., Lin (2003)).

Multi-sentence compression extends sentence compression to collections of related sentences that are to be summarised in a single output sentence. Traditionally, contributions to multi-sentence compression exploit linguistic properties based on lexical information and syntactic dependencies. Dorr *et al.* (2003) for instance propose a headline generation system based on linguistically-motivated, hand-crafted heuristics. Barzilay and Lee (2003) study sentence compression with dependency trees. The aligned trees are represented by a lattice from which a summary is extracted by an entropy-based criterion over all possible traversals of the lattice. Similarly, Barzilay and McKeown (2005) combine syntactic trees of similar sentences by a multi-sequence alignment candidate selection and summary generation. Wan (2007) deploys a language model in combination with maximum spanning trees to rank candidate aggregations satisfying grammatical constraints. Hori *et al.* (2003) propose a statistical model for automatic speech summarisation without using parallel data or syntactic information. Instead they focus on language models to provide a scoring function and use dynamic programming for searching the compression with the highest score. Clarke and Lapata (2008) cast sentence compression as an optimisation problem. They use linguistically motivated constraints and integer linear programming to infer globally optimal compressions.

Recently, graph-based approaches to multi-sentence compression have been proposed. The underlying idea is that syntax may help to find important content. Thus, instead of using hand-crafted rules, parsers, or language models, a simple and robust graph-based method can be used to generate reasonable summaries. Graph-based multi-sentence compression approaches identify the summary with the

shortest path in word graphs (Filippova, 2010). Shortest paths of unweighted word graphs however do not necessarily lead to satisfying summaries. As a remedy, Filippova (2010) introduces heuristic edge weights based on normalised frequencies of the connected words. Boudin and Morin (2013) propose an additional re-ranking scheme to identify summarisations that contain key phrases. The underlying idea is that particular key phrases give rise to certain topics and thus lead to more informative aggregations.

In this paper, we parameterise the graph-based framework by Filippova (2010) such that the shortest path algorithm is adapted to labeled data at hand. Adapting the dynamic programming to the data renders the use of heuristics unnecessary. Instead, word graphs and compressions are embedded in a (possibly high-dimensional) joint feature space where a generalised linear scoring function learns to separate between compressions of different quality. We develop a generalised, loss-augmented shortest path algorithm that is solved exactly by a (relaxed) integer linear program in polynomial time.

3 Preliminaries

3.1 Word Graphs

In a nutshell, word graphs represent collections of sentences efficiently in a graph by mapping identical words to a single vertex while the graph structure preserves the local neighbourhood of words.

From a collection of related sentences a word graph is constructed as follows: Initially, every sentence is augmented by a preceding start token $\langle S \rangle$ and a terminal end symbol $\langle E \rangle$ so that beginning and end of the sentences are preserved in the final graph. Starting with the empty graph, sentences are added one after another. The first word of the first sentence is the auxiliary $\langle S \rangle$ that is converted into the first vertex $v_{\langle S \rangle}$. The second word of the first sentence also becomes a vertex v and the two vertices are connected with a directed edge $v_{\langle S \rangle} \rightarrow v$. The procedure continues with the third word and so on until the end symbol $\langle E \rangle$ is reached. The other sentences are incorporated analogously. A special case arises if the graph already contains a vertex v that is identical to the word that is just to be added. Instead of adding a redundant vertex, the already existing vertex v is used and, if $v \neq v_{\langle S \rangle}$, connected to the respective predecessor as before. In that case, the vertex v has an in-degree of (at least) two and is used as the predecessor for the next word to be added. The procedure continues until all n sentences are incorporated in the graph.

Note that merging nodes to the same vertex requires an appropriate preprocessing of the sentences. Simple lower- or upper-case representations of words often suffice but more complex preprocessing schemas are also possible such as merging vertices carrying synonyms or words possessing small WordNet distances (Miller, 1995; Fellbaum, 1998). As word graphs are a condensed representation of the input sentences, word graphs are also known as compression graphs. The described construction gives us a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of unique words in the sentences and $\mathcal{E}$ the set of neighbouring words. An exemplary word graph is shown in Figure 1.

3.2 Shortest Path Algorithms

Given a directed weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, cost)$ where $\mathcal{V}$ is the set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges. The function $cost : (v, v') \mapsto \Re^+$ assigns positive weights to every edge $(v, v') \in \mathcal{E}$. A path p from a vertex $v_s \in \mathcal{V}$ to a vertex $v_e \in \mathcal{V}$ is a sequence of edges connecting vertices of $\mathcal{G}$. We write $\mathcal{P}(v_s, v_e)$ to denote the set of all possible paths starting in v_s and terminating in v_e. The cost of a path is given by the sum of the weights of the edges on the path.

The *shortest path* from a start vertex $v_s \in \mathcal{V}$ to an end vertex $v_e \in \mathcal{V}$ is defined as the path in G from v_s to v_e with the lowest costs. Introducing auxiliary binary variables $p_{(v,v')}$ indicating whether an edge $(v, v') \in \mathcal{E}$ lies on the path ($p_{v,v'} = 1$) or not ($p_{v,v'} = 0$) the shortest path can be computed by the following optimisation problem

$$p^* = \operatorname*{argmin}_{p} \sum_{(v,v') \in \mathcal{E}} p_{v,v'} \, cost(v, v') \quad \text{s.t.} \quad p \in \mathcal{P}(v_s, v_e). \tag{2}$$

There exist many algorithms for computing shortest paths efficiently (Bellman, 1958; Ford, 1956; Dijkstra, 1959). Usually, these methods are based on dynamic programming or (relaxed) integer program-

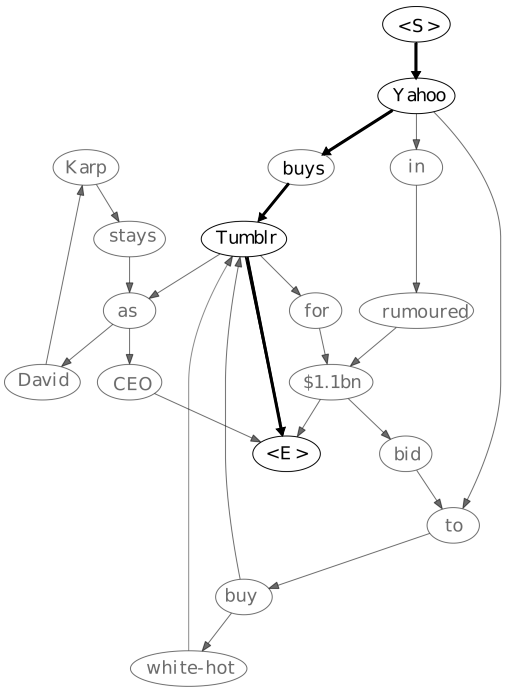


Figure 1: The word graph constructed from the sentences: "Yahoo in rumoured $1.1bn bid to buy white-hot Tumblr", "Yahoo buys Tumblr as David Karp stays as CEO", "Yahoo to buy Tumblr for $1.1bn". The shortest path is highlighted.

ming, where an approximation of the exact quantity is iteratively updated until it converges to the correct solution, which is achieved in polynomial time. A prominent algorithm for computing the k-shortest paths is Yen's algorithm (Yen, 1971). Intuitively, the approach recursively computes the second best solution by considering deviations from the shortest path, the third best solution from the previous two solutions, and so on. Figure 1 visualises the shortest path for the displayed compression graph.

4 Learning to Summarise Related Sentences

4.1 Problem Setting

Given a word graph $\mathcal{G}$, we aim to find a ranking function $f(\mathcal{G}, p)$ that assigns the lowest score to the best summary p^*, that is, $p^* \stackrel{!}{=} \operatorname{argmin}_p f(\mathcal{G}, p)$. Note that f is defined jointly on $\mathcal{G}$ and p to allow for exploiting dependencies between word graph and summary. Our approach can thus be seen as an instance of structured prediction models. The quality of f is measured by the Hamming loss Δ, $\Delta(p^*, \hat{p}) = \frac{1}{2}\sum_{(v_i,v_j)\in\mathcal{V}}[[p^*_{ij} \neq \hat{p}_{ij}]]$, that details differences between the best summary p^* and the prediction $\hat{p}$, where $[[z]]$ is the indicator function returning one if z is true and zero otherwise. The generalisation error is given by

$$R[f] = \int \Delta\left(p, \operatorname*{argmin}_{\tilde{p}} f(\mathcal{G}, \tilde{p})\right) dP(\mathcal{G}, p)$$

and approximated by its empirical counterpart

$$\hat{R}[f] = \sum_{i=1}^{m} \Delta\left(p_i, \operatorname*{argmin}_{\tilde{p}} f(\mathcal{G}_i, \tilde{p})\right) \tag{3}$$

on a finite m-sample of pairs $\{(\mathcal{G}_i, p_i)\}_{i=1}^m$ where $\mathcal{G}_i$ is a word graph and p_i the best summarising sentence. However, minimising the empirical risk directly leads to an ill-posed optimisation problem as

there generally exist many indistinguishable but equally well solutions realising an empirical loss of zero. We thus focus on the minimisation of the regularised empirical risk

$$\hat{R}_{reg}[f] = \Omega(f) + \sum_{i=1}^{m} \Delta\left(p_i, \operatorname*{argmin}_{\tilde{p}} f(\mathcal{G}_i, \tilde{p})\right).$$

The additive regularisation $\Omega(f)$ acts like a prior on f, e.g. to enforce smooth solutions. In the remainder we use $\Omega(f) = \|f\|^2$.

4.2 Representation

The idea of our approach is as follows: We adapt the cost function of the graph to the training sample such that the shortest path of the compression graph is identical to the desired summary. Recall the general form of the cost function of Section 3.2. Instead of a constant or hand-crafted function (Filippova, 2010), we deploy a linear mixture of features ϕ_i, parameterised by λ,

$$cost(v, v') = \sum\nolimits_i \lambda_i \phi_i(v, v') = \lambda^\top \phi(v, v').$$

Features $\phi_i(v, v')$ are drawn from adjacent vertices v, v' in the word graph to capture local dependencies of the connecting edge. Examples for feature functions are frequency-based counts or indicators such as POS-transitions of the form $\phi_{234}(v, v') = [[v \text{ is a noun} \wedge v' \text{ is a verb}]]$. Note that complex features using the context of the edge are straight forward by extending the feature representation to the input graph $\phi(v, v', \mathcal{G})$. The final feature vector is obtained by stacking-up all feature functions, that is, $\phi(v, v') = (\ldots, \phi_i(v, v'), \ldots)^\top$.

Using the parameterised costs in the computation of the shortest path in Equation (2) yields the following objective function (ignoring the constraints for a moment) that can be rewritten as a generalised linear model in joint input output space

$$\sum_{(v_i,v_j)\in\mathcal{V}} p_{ij}\, \lambda^\top \phi(v_i, v_j) = \lambda^\top \underbrace{\left(\sum_{(v_i,v_j)\in\mathcal{V}} p_{ij} \phi(v_i, v_j)\right)}_{\equiv \Phi(\mathcal{G}, p)} = \lambda^\top \Phi(\mathcal{G}, p) = f(\mathcal{G}, p)$$

where the joint feature representation is given by

$$\Phi(\mathcal{G}, p) \equiv \left(\sum_{(v_i,v_j)\in\mathcal{V}} p_{ij} \phi(v_i, v_j)\right).$$

Decoding the shortest path $\hat{p}$ for a fixed parameter vector λ can now be computed by

$$\hat{p} = \operatorname*{argmin}_{p} f(\mathcal{G}, p) \quad s.t.\ p \in \mathcal{P}(\langle S\rangle, \langle E\rangle)$$

using standard shortest path algorithms (Yen, 1971). In addition, reformulating the objective as a generalised linear model allows to adapt the parameters λ to the data to identify shortest paths with summaries.

4.3 Optimisation

Recall that the goal of the optimisation is to find the ranking function $f(\mathcal{G}, p)$ that takes the smallest value for the best summary. That is, for the i-th training instance $(\mathcal{G}_i, p_i)$, we aim at fulfilling the constraints

$$\lambda^\top \Phi(\mathcal{G}_i, p) - \lambda^\top \Phi(\mathcal{G}_i, p_i) > 0 \tag{4}$$

for all $p \in \mathcal{P}(\langle S\rangle, \langle E\rangle) \backslash p_i$. We extend the constraints in Equation (4) by a term that induces a margin between the best path p_i and all alternative paths. A common technique is called margin-rescaling and implies to scale the margin with the actual loss that is induced by decoding $\tilde{p}$ instead of p_i. Thus,

rescaling the margin by the loss implements the intuition that the confidence of rejecting a mistaken output is proportional to its error. In the context of learning shortest paths, margin-rescaling gives us the following constraints

$$\lambda^\top \Phi(\mathcal{G}_i, \tilde{p}) - \lambda^\top \Phi(\mathcal{G}_i, p_i) > \Delta(p_i, \tilde{p}) - \xi_i$$

for all $p \in \mathcal{P}(\langle S \rangle, \langle E \rangle) \backslash p_i$. The non-negative $\xi_i \geq 0$ is a slack-variable that allows point-wise relaxations of the margin. Solving the equation for ξ_i shows that margin rescaling also effects the hinge loss that now augments the structural loss Δ,

$$\ell_\Delta(\mathcal{G}_i, p_i, f) = \max \left[\min_{\tilde{p}} \left[\Delta(p_i, \tilde{p}) - f(\mathcal{G}_i, \tilde{p}) + f(\mathcal{G}_i, p_i) \right] \right].$$

The effective hinge loss upper bounds the structural loss Δ for every pair $(\mathcal{G}_i, p_i)$ and trivially also

$$\sum_{i=1}^{m} \ell_\Delta(\mathcal{G}_i, p_i, f) \geq \sum_{i=1}^{m} \Delta(p_i, \operatorname*{argmin}_{\tilde{p}} f(\mathcal{G}_i, \tilde{p}))$$

holds. A max-margin approach to learning shortest paths therefore leads to the following optimisation problem that is also known as structural support vector machine (Tsochantaridis *et al.*, 2005)

$$\min_{\lambda, \xi \geq 0} \ \|\lambda\|^2 + C \sum_{i=1}^{m} \xi_i \quad s.t. \quad \forall i \, \forall \tilde{p} \in P \backslash p_i : \ f(\tilde{p}) - f(p_i) > \Delta(p_i, \tilde{p}) - \xi_i. \tag{5}$$

The parameter C trades-off margin maximisation and error minimisation and needs to be adjusted by the user. The above optimisation problem can be solved efficiently by cutting plane methods. The idea behind cutting planes is to instantiate only a minimal subset of the exponentially many constraints. That is, for the i-th training example, we decode the shortest path $\hat{p}$ given our current model and consider two cases: (i) For $\hat{p} \neq p_i$ the prediction is erroneous and $\hat{p}$ is called the most strongly violated constraint as it realises the smallest function value and $f(\mathcal{G}_i, \hat{p}) < f(\mathcal{G}_i, p)$ holds for all $p \neq \hat{p}$. Consequentially, the respective constraint of the above optimisation problem is instantiated and influences the subsequent iterations. (ii) If instead the prediction is correct, that is $\hat{p} = p_i$, we need to verify that the runner-up $\hat{p}^{(2)}$ fulfils the margin constraint. If so, we proceed with the next training example, otherwise we instantiate the corresponding constraint, analogously to case (i). Luckily, we do not need to rely on an expensive two-best shortest path algorithm but can compute the most strongly violated constraint directly via the cost function

$$Q(\tilde{p}) = \Delta(p_i, \tilde{p}) - \lambda^\top \Phi(\mathcal{G}_i, \tilde{p}) + \lambda^\top \Phi(\mathcal{G}_i, p_i) \tag{6}$$

that has to be maximised wrt $\tilde{p}$. The following proposition shows that we can equivalently solve a shortest path problem for finding the maximiser of Q.

Proposition 1. *The argmax $\tilde{p}$ of Q in Equation* (6) *can be computed by minimising a shortest path problem with cost function $cost(v_i, v_j) = p_{ij} + \lambda^\top \phi(v_i, v_j)$.*

Proof. We treat the ground truth paths p as graphs and write $\mathcal{V}(p)$ for the set of nodes on the path and $\mathcal{E}(p)$ to denote the set of edges that lie on the path. If, for instance, an element of the binary adjacency matrix representing path p equals one, e.g., $p_{ij} = 1$, we write $p_i, p_j \in \mathcal{V}(p)$ and $(p_i, p_j) \in \mathcal{E}(p)$. First, note that the Hamming loss can be rewritten as

$$\Delta(p, \tilde{p}) = \sum_{(p_i, p_j) \in \mathcal{E}(p)} (1 - p_{ij} \tilde{p}_{ij}) \, .$$

We have

$$\begin{aligned}
\hat{p} &= \operatorname*{argmax}_{\tilde{p}} \Delta(p,\tilde{p}) + \lambda^\top \Phi(\mathcal{G}_i, p) - \lambda^\top \Phi(\mathcal{G}_i, \tilde{p}) \\
&= \operatorname*{argmax}_{\tilde{p}} \Delta(p,\tilde{p}) - \lambda^\top \Phi(\mathcal{G}_i, \tilde{p}) \\
&= \operatorname*{argmax}_{\tilde{p}} \sum_{(p_i,p_j)\in\mathcal{E}(p)} (1 - p_{ij}\tilde{p}_{ij}) - \lambda^\top \Phi(\mathcal{G}_i, \tilde{p}) \\
&= \operatorname*{argmax}_{\tilde{p}} -\sum_{(p_i,p_j)\in\mathcal{E}(p)} p_{ij}\tilde{p}_{ij} - \lambda^\top \Phi(\mathcal{G}_i, \tilde{p}) \\
&= \operatorname*{argmin}_{\tilde{p}} \sum_{(p_i,p_j)\in\mathcal{E}(p)} p_{ij}\tilde{p}_{ij} + \lambda^\top \Phi(\mathcal{G}_i, \tilde{p}) \\
&= \operatorname*{argmin}_{\tilde{p}} \sum_{(p_i,p_j)\in\mathcal{E}(p)} p_{ij}\tilde{p}_{ij} + \lambda^\top \Big[\sum_{(x_i,x_j)\in E(\mathcal{G})} \tilde{p}_{ij}\phi(v_i,v_j)\Big] \\
&= \operatorname*{argmin}_{\tilde{p}} \sum_{(v_i,v_j)\in\mathcal{E}(\mathcal{G})} p_{ij}\tilde{p}_{ij} + \lambda^\top \Big[\sum_{(x_i,x_j)\in E(\mathcal{G})} \tilde{p}_{ij}\phi(v_i,v_j)\Big] \\
&= \operatorname*{argmin}_{\tilde{p}} \sum_{(v_i,v_j)\in\mathcal{E}(\mathcal{G})} \Big[p_{ij} + \lambda^\top \phi(v_i,v_j)\Big]\tilde{p}_{ij}
\end{aligned}$$

The output $\hat{p}$ is the shortest path with costs given by $p_{ij} + \lambda^\top \phi(v_i, v_j)$. □

Using this result, the following lemma shows that we can compute the most strongly violated constraint directly by a linear program.

Lemma 1. *The maximizer $\tilde{p}$ of function Q in Equation* (6) *and thus the shortest path of Proposition 1 can be computed in polynomial time by the following linear program*

$$\min_{\tilde{p}} \sum_{ij} \Big(p_{ij} + \lambda^\top \phi(v_i,v_j)\Big)\,\tilde{p}_{ij}$$

subject to the constraints

$$\begin{aligned}
\forall k \in \mathcal{V}(\mathcal{G})\backslash\{\langle S\rangle, \langle E\rangle\}: \ \sum_j \tilde{p}_{kj} - \sum_i \tilde{p}_{ik} \le 0 \ \wedge \ -\sum_j \tilde{p}_{kj} + \sum_i \tilde{p}_{ik} \le 0 \\
\sum_j \tilde{p}_{\langle S\rangle,j} - \sum_i \tilde{p}_{i,\langle S\rangle} \le 1 \ \wedge \ -\sum_j \tilde{p}_{\langle S\rangle,j} + \sum_i \tilde{p}_{i,\langle S\rangle} \le -1 \\
\sum_i \tilde{p}_{i,\langle E\rangle} - \sum_j \tilde{p}_{\langle E\rangle,j} \le 1 \ \wedge \ -\sum_i \tilde{p}_{i,\langle E\rangle} + \sum_j \tilde{p}_{\langle E\rangle,j} \le -1 \\
\forall (i,j): \ \tilde{p}_{ij} \le \mathcal{G}_{(i,j)} \ \wedge \ \forall (i,j): \ \tilde{p}_{ij} \in \{0,1\}.
\end{aligned}$$

Proof. For lack of space, we only motivate the constraints. The first line of constraints guarantees that every inner node of the path has exactly as many incoming as outgoing edges, the second line forces the path to start in $v_{\langle S\rangle}$ and, analogously, the third line ensures that it terminates in $v_{\langle E\rangle}$. The last line of constraints requires the edges of the path $\tilde{p}$ to adhere to existing paths of $\mathcal{G}$. □

4.4 Parallelisation

Using the result by Zinkevich *et al.* (2011) the proposed approach can easily be distributed on several machines. Note that cutting planes treat one input $(\mathcal{G}, p)$ at a time. Thus, several models can be trained independently in parallel on disjoint subsets of the data. A subsequent merging process aggregates the models where each models impact is proportional to the amount of data it has been trained on. Note that the described parallelisation can easily be realised by the MapReduce/Hadoop framework. Processing training instances and updating local models is performed by (one or more) mappers while the merge operation is carried out by a reduce task.

Table 1: Left: Collection of related sentences. Right: Candidate compressions and number of votes.

related sentences	summary	#
White House: Hong Kong had 'plenty of time' to stop Snowden live coverage	snowden seeks asylum	5
Edward Snowden leaves reporters chasing shadows around an airport	snowden live coverage	5
US warns Moscow not to let Edward Snowden escape Russia	snowden escape russia	1
WikiLeaks forced to defend Ecuador as Edward Snowden seeks asylum	edward snowden seeks asylum	3
Snowden is 'not on plane' to Cuba	wikileaks forced to cuba	1

5 Empirical Results

5.1 Data Preparation

We crawl RSS feeds of 6 major news sites and harvest news headlines of a predefined set of categories including sports, technology, and business. The headlines are processed automatically by a spectral clustering. The data is thus transformed into a fully connected graph where vertices correspond to headlines and edges are weighted by the number of shared non-stopwords. The clustering is performed for each category on a daily basis. Resulting clusters are headlines that belong (with high probability) to the same event and form our related input sentences. Groups with less than five sentences are discarded.

To identify the best summaries, we conduct a crowd sourcing experiment on Crowdflower[1]. Every annotator is given a group of related sentences together with 10 possible summaries generated by a 10-best Yen's algorithm (Yen, 1971). The task of the annotator is to pick the best summary or mark the collection as inappropriate. Each collection is labeled by at least 10 annotators. The group is discarded if the majority of the annotators mark the group as inappropriate. Otherwise, the three most frequent summaries are extracted, ties are broken by the authors. The most frequent summary is used as the ground-truth annotation in the learning phase, the other two are used additionally in the evaluation. The described process leaves us with 1024 sentences that are divided into 164 annotated groups of related sentences. Table 1 shows an exemplary collection of related sentences (left) and a selection of summaries together with the number of votes from the annotators. The overall distribution of votes is displayed in Figure 2. The figure shows the mean value per rank of all 164 normalised and sorted histograms. The best summary receives on average 8% more votes than the runner-up (not shown).

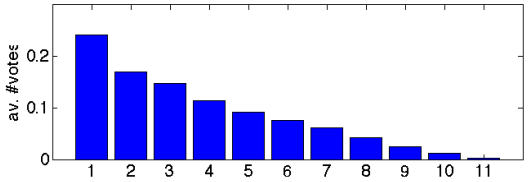


Figure 2: Distribution of annotations.

5.2 Baselines and Features

We compare our learning approach to graph-based sentence compression techniques proposed by Filippova (2010), Boudin and Morin (2013). The two baselines construct word graphs as presented in Section 3.1 and output the weighted shortest path. Filippova (2010) uses a frequency-based heuristic for edges weights and Boudin and Morin incorporate a keyphrase detection framework to re-rank summaries according to the number and importance of keyphrases found. In addition, we also include an unweighted shortest path strategy which is a straight forward application of Yen's algorithm (Yen, 1971) and trivially returns the shortest path in terms of the number of edges. Additional straw men are a random (Random) input sentence and the shortest input sentence (Shortest).

In our learning-based approach, every edge between vertices v and v' is associated with a feature vector. Let $w = \#(v)$ the frequency of word v, w' the analogue for v', $e = \#(v, v')$ the frequency of the edge, and $n = |\mathcal{V}|$ the number of vertices in the graph. The feature vector $\phi(v, v')$ of

[1] http://crowdflower.com

Table 2: ROUGE F-measure scores

		training set size						
		22	35	48	61	74	87	100
R1	Random	46.72	46.82	46.41	46.20	46.39	46.53	46.88
	Shortest	45.93	45.77	46.39	46.56	47.01	47.59	48.04
	Yen	45.14	44.47	45.12	45.13	45.63	46.14	46.39
	Filippova	52.70	52.94	52.16	52.02	52.22	52.45	51.81
	Boudin	52.72	53.12	53.43	53.52	53.10	52.81	52.35
	SVM	48.39	50.30	**55.09**	**54.59**	**57.39**	**54.89**	**57.66**
R2	Random	30.43	30.63	30.56	30.31	30.38	30.64	31.09
	Shortest	27.65	27.43	27.90	27.93	28.64	29.47	30.10
	Yen	31.38	30.82	31.16	31.40	31.90	32.30	32.56
	Filippova	36.12	36.52	35.56	35.49	35.75	35.98	35.64
	Boudin	36.71	37.01	37.79	37.65	36.97	36.75	36.31
	SVM	33.64	35.40	**40.46**	**40.68**	**43.44**	**40.45**	**43.58**
RW1.2	Random	35.91	35.97	35.74	35.58	35.80	35.93	36.07
	Shortest	34.47	34.29	34.77	34.85	35.32	35.9	36.16
	Yen	34.85	34.26	34.74	34.83	35.22	35.62	35.77
	Filippova	40.30	40.53	39.88	39.70	39.94	40.12	39.56
	Boudin	40.79	40.99	41.37	41.31	40.92	40.83	40.36
	SVM	37.94	39.06	**42.61**	**42.33**	**44.63**	**42.90**	**45.00**

the edge $v \rightarrow v'$ consists of the normalised joint frequency $\phi_1(w, w') = \frac{e}{n}$, the maximal word frequency $\phi_2(w, w') = \max\left\{\frac{w}{n}, \frac{w'}{n}\right\}$, the lexical relevance $\phi_3(w, w') = \frac{2}{n}\frac{w \cdot w'}{w + w'}$, the normalised PMI $\phi_4(w, w') = (\log \frac{e}{w \cdot w'})/ - \log \frac{e}{n}$ (Bouma, 2009), and ϕ_5 captures the average location of the phrase in the input sentences (Turney, 2000),

$$\phi_5(w, \tilde{w}) = \begin{cases} 1.0 & : & [0-10]\% \\ 0.4 & : & [10-30]\% \\ 0.8 & : & [30-60]\% \\ 0.6 & : & [60-80]\% \\ 1.0 & : & [80-100]\%. \end{cases}$$

Note that $\phi_i \in [0, 1]$ holds for $1 \leq i \leq 5$. Also note that ϕ denotes a rudimentary set of features. Elaborate representations could for instance also contain POS-tags or named entities.

5.3 Experimental Setup and Results

For the news headline experiment, we draw $m \in \{22, 35, 48, 61, 74, 87, 100\}$ training instances without replacement at random from the collected data. The remaining instances are split randomly into equally sized holdout and test sets. We perform model selection for adjusting the trade-off parameter of the support vector machine on the interval $C \in [2^{-10}, 2^{12}]$. We report average ROUGE F-measures (Lin, 2004) and BLEU scores (Papineni *et al.*, 2012) over 10 repetitions with distinct training, holdout, and test sets. In each repetition, all algorithms are trained and/or evaluated on identical data splits.

ROUGE measures the concordance of system and human generated summaries by determining n-gram, word sequence, and word pair matches. We use unigrams (R1), bigrams (R2), and the weighted longest common subsequence (RW1.2) to evaluate compressions. Note that R1 has been found to correlate well with human evaluations based on various statistical metrics (Lin and Hovy, 2003). Moreover, R1 and R2 emulate human pyramid and responsiveness scores (Owczarzak *et al.*, 2012).

Table 2 shows the resulting ROUGE scores for the news headline experiment. Significant results are marked in bold face according to a paired t-test using a significance level of 5%. For small training sets, the structural support vector machine performs only slightly better than the unweighted application of Yen's algorithm and is clearly outperformed by the unsupervised baselines. However, the SVM improves

Table 3: BLEU scores

		training set size						
		22	35	48	61	74	87	100
B1	Random	38.56	38.40	36.49	36.77	36.00	36.87	37.35
	Shortest	37.45	38.37	38.46	37.25	37.28	37.17	36.64
	Yen	29.46	28.3	29.39	29.98	29.99	31.20	30.64
	Filippova	44.26	43.29	44.66	44.57	45.21	43.10	43.52
	Boudin	44.00	42.54	44.75	44.39	44.80	43.22	43.96
	SVM	39.60	41.96	**48.44**	**47.10**	**50.20**	**46.90**	**50.39**
B2	Random	34.85	34.80	33.12	33.48	32.65	33.77	34.12
	Shortest	33.34	34.27	34.54	33.39	33.39	33.43	33.45
	Yen	28.51	27.27	28.34	28.74	29.06	30.05	29.73
	Filippova	39.92	39.36	40.05	40.27	41.14	39.26	39.60
	Boudin	39.43	38.45	39.99	40.02	40.52	39.20	39.84
	SVM	36.37	38.63	**45.31**	**44.15**	**47.40**	**43.75**	**47.44**
B3	Random	35.91	35.97	35.74	35.58	35.80	35.93	36.07
	Shortest	34.47	34.29	34.77	34.85	35.32	35.90	36.16
	Yen	27.85	26.64	27.61	27.84	28.38	29.34	28.93
	Filippova	36.07	35.88	35.86	36.42	37.37	35.55	35.97
	Boudin	35.05	34.39	35.40	35.76	36.17	34.93	35.85
	SVM	33.26	35.39	**42.31**	**41.05**	**44.54**	**40.52**	**44.51**

continuously with increasing training set sizes and outperforms the baselines significantly for more than 50 training examples. The unsupervised baselines cannot utilise the valuable annotations of the data and remain constant. For 100 training instances, we observe performance improvements of about 5-7% for all three ROUGE F-measures.

The BLEU metric computes scores for individual segments, then averages these scores over the whole corpus for a final score. For our experiments we use BLEU-1, BLEU-2 and BLEU-3 to evaluate compressions. Table 3 shows the corresponding results, significant results are again marked in bold face according to a paired t-test with a significance level of 5%. The table draws a similar picture than the previous one. The SVM continuously improves the performance with increasing training set sizes and beats the baselines again at about 50 training examples significantly. For 100 training instances, all three BLEU scores are improved by about 7-8%, respectively.

5.4 Analysis

The Pearson correlation between BLEU scores per instance and the number of vertices is -0.1886. The negative correlation implies that summarising larger word-graphs is more challenging. A negative correlation of -0.1267 is also observed for the lexical diversity of the collection; diverse groups of sentences are thus more difficult to summarise. A possible remedy could be features that are not frequency-based, such as POS-transitions. By contrast, the density of the graph given by $|\mathcal{E}|/|\mathcal{V}|(|\mathcal{V}|-1)$ shows a positive correlation of 0.1851. The more dense a graph, the more edges interconnect vertices and there exist more paths. These paths however frequently pass through the same vertices and as a consequence the lexical diversity is low. A positive correlation of graph density is therefore closely connected to a negative correlation of lexical diversity.

6 Conclusion

We proposed to learn shortest paths in word graphs for multi-sentence compression. A shortest path algorithm is parameterised and adapted to labeled data at hand using the structured prediction framework. Word graphs and summaries are embedded in a joint feature space where a generalised linear scoring function learns to separate between compressions of different quality. Decoding is performed

by a generalised, loss-augmented shortest path algorithm that can be solved by an integer linear program in polynomial time. Empirically, we observe that a very rudimentary set of five features suffices to significantly improve the state-of-the-art in graph-based multi-sentence compression.

Acknowledgments

Jamal Abdul Nasir is supported by a grant from the Higher Education Commission, H-9 Islamabad, Pakistan.

References

R. Barzilay and L. Lee. 2003. *Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment*, Proceedings of NAACL-HLT.

R. Barzilay and K. R. McKeown. 2005. *Sentence Fusion for Multidocument News Summarization*, Comput. Linguist. 31(3), 297–328.

P. Baxendale 1958. *Machine-made index for technical literature - an experiment*, IBM Journal of Research Development, 2(4):354–361.

R. Bellman 1958. *On a routing problem*, Quarterly of Applied Mathematics 16:87–90.

F. Boudin and E. Morin. 2013 *Keyphrase Extraction for N-best reranking in multi-sentence compression*, Proceedigs of NAACL-HLT

G. Bouma. 2009. *Normalized (pointwise) Mutual information in collocation extraction*, Proceedings of GSCL.

J. Clarke and M. Lapata. 2008. *Global inference for sentence compression: An integer linear programming approach*, Journal of Artificial Intelligence Research, 31:399–429.

S. H. Corston-Oliver 2001. *Text compaction for display on very small screens*, Proceedings of the NAACL Workshop on Automatic Summarization.

E. W. Dijkstra 1959. *A note on two problems in connexion with graphs*, Numerische Mathematik 1:269–271.

B. Dorr, D. Zajic, and R. Schwartz. 2003. *Hedge trimmer: A parse-and-trim approach to headline generation*, Proceedings of the HLT-NAACL Workshop on Text Summarization.

H. P. Edmundson 1969. *New methods in automatic extracting*, Journal of the ACM, 16(2):264–285.

C. Fellbaum (Ed.). 1998. *WordNet: An Electronic Lexical Database*, Cambridge, MA: MIT Press.

K. Filippova. 2010. *Multi-sentence compression: Finding shortest paths in word graphs*, Proceedings of COLING.

K. Filippova and M. Strube. 2008. *Dependency tree based sentence compression*, Proceedings of INLG.

J. Ford, R. Lester 1956. *Network Flow Theory*, Paper P-923, Santa Monica, California: RAND Corporation.

M. Galley and K. R. McKeown. 2007. *Lexicalized Markov grammars for sentence compression*, Proceedings of NAACL-HLT

U. Hermjakob, A. Echihabi, and D. Marcu. 2002. *Natural language based reformulation resource and wide exploitation for question answering*, Proceedings of the Text Retrieval Conference.

C. Hori, S. Furui, R. Malkin, H. Yu, and A. Waibel. 2003. *A statistical approach to automatic speech summarization*, EURASIP Journal on Applied Signal Processing, 2:128–139.

H. Jing 2000. *Sentence reduction for automatic text summarization*, Proc. of ANLP.

H. Jing and K. McKeown. 2000. *Cut and paste based text summarization*, Proc. of NAACL.

N. Kaji, M. Okamoto, and S. Kurohashi,. 2004. *Paraphrasing predicates from written language to spoken language using the web*, Proceedings of HLT-NAACL.

J. Kupiec, J. Pedersen, and F. Chen 1995 *A trainable document summarizer*, Proceedings of SIGIR.

C. Lin. 2003. *Improving summarization performance by sentence compression - a pilot study*, Proceedings of the Int. Workshop on Information Retrieval with Asian Language.

C. Lin. 2004. *Rouge: A package for automatic evaluation of summaries*, Proceedings of the ACL Workshop on Text Summarization Branches Out.

C.-Y. Lin and E. H. Hovy. 2003. *Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics*, Proceedings of HLT-NAACL.

H.P. Luhn. 1958. *The Automatic Creation of Literature Abstracts*, IBM Journal of Research and Development 2(2), 159–165.

I. Mani. 2001. *Automatic Summarization*, Amsterdam, Philadelphia: John Benjamins.

D. Marcu. 1997 *The Rhetorical Parsing of Natural Language Texts*, Proceedings of ACL/EACL.

K. R. McKeown, J. Hirschberg, M. Galley, and S. Maskey. 2005. *From Text to Speech Summarization*, Proceedings of ICASSP.

G. A. Miller. 1995. *WordNet: a lexical database for English*, Communications of the ACM Vol. 38, No. 11: 39-41.

A. Nenkova, L. Vanderwende, and K. McKeown. 2006. *A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization*, Proceedings of SIGIR.

K. Owczarzak, J. M. Conroy, H. T. Dang, and A. Nenkova. 2012. *An assessment of the accuracy of automatic evaluation in summarization*, Proceedings of the Workshop on Evaluation Metrics and System Comparison for Automatic Summarization.

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

D. Roussinov and H. Chen. 2001 *Information Navigation on the Web by Clustering and Summarizing Query Results*, Information Processing and Management, 37 (6), 789–816.

G. Salton, J. Allan, C. Buckley, and A. Singhal, 1994 *Automatic Analysis, Theme Generation, and Summarization of Machine-Readable Texts*, Science 264(5164), 1421–1426.

B. Taskar and D. Klein and M. Collins and D. Koller and C. Manning. 2004. *Max-margin parsing*, Proceedings of EMNLP, 2004.

S. Teufel and M. Moens. 1997 *sentence extraction as a classification task*, Proceedings of the ACL/EACL Workshop on Intelligent and Scalable Text Summarization.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. 2005. *Large margin methods for structured and interdependent output variables*, JMLR, 6 (Sep):1453-1484.

P. D. Turney. 2000. *Learning algorithms for keyphrase extraction*, Information Retrieval 2(4), 303–336.

S. Wan, R. Dale, M. Dras, C. Paris. 2007. *Global revision in summarisation : generating novel sentences with Prim's algorithm*, Proceedings of PACLING.

J. Y. Yen. 1971. *Finding the k shortest loopless paths in a network*, Management Science 17 (11): 712–716

M. Zinkevich, M. Weimer, A. Smola, and L. Li. 2011. *Parallelized stochastic gradient descent*, Proceedings of NIPS.

Learning to Generate Coherent Summary with Discriminative Hidden Semi-Markov Model

Hitoshi Nishikawa[1], **Kazuho Arita**[1], **Katsumi Tanaka**[1],
Tsutomu Hirao[2], **Toshiro Makino**[1] and **Yoshihiro Matsuo**[1]
Nippon Telegraph and Telephone Corporation
[1] 1-1 Hikari-no-oka, Yokosuka-shi, Kanagawa, 239-0847 Japan
[2] 2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0237 Japan
{ nishikawa.hitoshi, arita.kazuho, tanaka.katsumi
hirao.tsutomu, makino.toshiro, matsuo.yoshihiro } @lab.ntt.co.jp

Abstract

In this paper we introduce a novel single-document summarization method based on a hidden semi-Markov model. This model can naturally model single-document summarization as the optimization problem of selecting the best sequence from among the sentences in the input document under the given objective function and knapsack constraint. This advantage makes it possible for sentence selection to take the coherence of the summary into account. In addition our model can also incorporate sentence compression into the summarization process. To demonstrate the effectiveness of our method, we conduct an experimental evaluation with a large-scale corpus consisting of 12,748 pairs of a document and its reference. The results show that our method significantly outperforms the competitive baselines in terms of ROUGE evaluation, and the linguistic quality of summaries is also improved. Our method successfully mimicked the reference summaries, about 20 percent of the summaries generated by our method were completely identical to their references. Moreover, we show that large-scale training samples are quite effective for training a summarizer.

1 Introduction

Single-document summarization is attracting much more attention as a key technology in providing better information access in a commercial context. The Financial Times and CNN have been providing summaries of articles in their websites to attract users, and Summly, which has been acquired by Yahoo!, provided the service of automatically summarizing articles on the Internet. Given the cost of manual summarization, we can greatly improve the information access of Internet users by creating an automatic summarizer that can approach the summarization quality of humans.

To mimic manually-written summaries, one important aspect is coherence (Nenkova and McKeown, 2011). Although coherence has been studied widely in a field of multi-document summarization (Karamanis et al., 2004; Barzilay and Lapata, 2005; Nishikawa et al., 2010; Christensen et al., 2013), it has not been studied enough in the context of single-document summarization. In this paper, we revisit the problem of coherence and employ it to produce both informative and linguistically high-quality summaries.

To obtain such summaries, we introduce a novel summarization method based on a hidden semi-Markov model. The method has the properties of both the popular single-document summarization model, the knapsack problem, which packs the sentences into the given length and the hidden Markov model, which takes summary coherence into account by determining sentence context when selecting sentences. By leveraging this, we can build a summarizer that naturally achieves coherence.

We state the novelty and contributions of this paper as follows:

- We regard single-document summarization as a combinatorial optimization problem modeled by a hidden semi-Markov model and propose an efficient decoding algorithm for the problem.
- We introduce various features related to coherence in a combinatorial formulation. We extend a hidden semi-Markov model to achieve discrimination, so our method can take advantage of many features for predicting coherence.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1648–1659, Dublin, Ireland, August 23-29 2014.

- We show that our large-scale corpus greatly improves the performance of summarization.

This paper is organized as follows. In Section 2, we describe related work. In Section 3, we detail our proposed model. We also explain how the parameters in our model are optimized and how sentences are compressed. In Section 4, we explain how variants of the original sentences are generated. In Section 5, we explain the decoding algorithm for our method. In Section 6, we explain the settings of our experiments, our corpus, and compared methods. In Section 7, we show results of the experiments conducted to evaluate our method. In Section 8, we conclude this paper.

2 Related Work

2.1 Single-Document Summarization

Basically, single-document summarization can be done through sentence selection (Nenkova and McKeown, 2011) . The document to be summarized is decomposed into a set of sentences and then the summarizer selects a subset of the sentences as a summary.

McDonald (2007) pointed out that single-document summarization can be formulated as a well-known combinatorial optimization problem, the knapsack problem. Given a set of sentences together with their lengths and values, the summarizer packs them into a summary so that the total value is as large as possible but the total length is less than or equal to a given maximum summary length. Interestingly, a hidden semi-Markov model (Yu, 2010) can be regarded as a natural extension of the knapsack problem, we take advantage of this property for single-document summarization. We elaborate the relation between the knapsack problem and the hidden semi-Markov model in Section 3.

To generate coherent summaries in single-document summarization, there are two types of approaches[1] : tree-based approaches (Marcu, 1997; Daume and Marcu, 2002; Hirao et al., 2013) and sequence-based approaches (Barzilay and Lee, 2004; Shen et al., 2007). The former rely on the tree representation of a document based on the Rhetorical Structure Theory (RST) (Mann and Thompson, 1988). Basically, the former approaches (Marcu, 1997; Daume and Marcu, 2002; Hirao et al., 2013) trim the tree representation of a document by making use of nucleus-satellite relations among sentences. The advantage of RST-based approaches is that they can take advantage of global information about the documents. However, a drawback is that they depend heavily on the RST parser that is used. Performance is remarkably sensitive to the accuracy of RST parsing, and hence we have to build a good RST parser. Instead of making use of the global structure of the document, the sequence-based methods rely on and take advantage of the local coherence of sentences. As one advantage over the tree-based approaches, the sequence-based approaches do not require tools as RST parsers, and hence they are more robust. For this reason, this paper focuses on sequence-based approaches.

The previous works most closely related to our method are those proposed by Barzilay and Lee (2004) and Shen et al. (2007). Barzilay and Lee built a hidden Markov model to capture the content structure of documents and used it to identify the important sentences. Shen et al. (2007) extended the HMM-based approach to make it discriminative by making use of conditional random fields (Lafferty et al., 2001). Conditional random fields can incorporate various features to identify the importance of a sentence and they showed its effectiveness. A shortcoming of these approaches is that their model only classifies sentences into two classes, it cannot take account of output length directly. This deficiency is problematic because in practical usage the maximum length of a summary is specified by the user; hence, the summarizer should be able to control output length. In contrast to their method, our approach naturally takes the maximum summary length into account when summarizing a document.

2.2 Coherence

In the context of multi-document summarization, coherence has been studied widely. In multi-document summarization, sentences are selected from different documents, and hence some way of ordering the sentences is required. Sentence ordering (Barzilay et al., 2002; Althaus et al., 2004; Karamanis et al.,

[1]As an interesting related work, Clarke and Lapata (2007) compresses documents by making use of Centering Theory (Grosz et al., 1995). However, in their approach, the desired length of an output summary could not be specified and hence they said their method was compression rather than summarization.

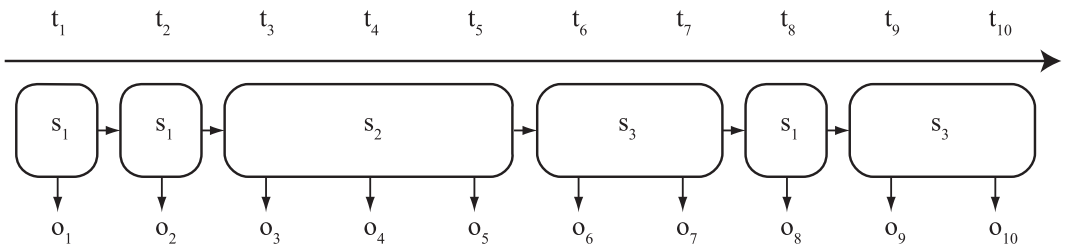


Figure 1: An example of the hidden semi-Markov model. The system observes a sequence consisting of 10 symbols $o_1...o_{10}$ over time $t_1...t_{10}$ and transitions between states $s_1...s_3$. Unlike the basic hidden Markov model, states can persist for a non-unit length. In this figure, state s_2 and state s_3 persist for non-unit lengths. Hence, the system traverses only 6 states despite observing 10 symbols.

2004; Okazaki et al., 2004) is a task to order extracted sentences and is closely related to coherence (Lapata, 2003; Barzilay and Lapata, 2005; Nenkova et al., 2010; Pitler et al., 2010; Louis and Nenkova, 2012). Many effective features have been found out to capture coherence and we utilize these features.

Some work proposed a model that could jointly taking the content of the summary and its coherence into account (Nishikawa et al., 2010; Christensen et al., 2013). Since extracted sentences in multi-document summarization must be ordered, a task that is NP-hard, they relied on integer linear programming (Nishikawa et al., 2010) or a local search strategy (Christensen et al., 2013). The former can locate the optimal solution at a heavy computation cost, while the latter runs quickly but there is no guarantee of locating the optimal solution. In contrast to their trade-off, our proposed algorithm, based on dynamic programming, can locate the optimal solution quickly because the single-document summarization can skip the ordering operation by reproducing the original order of the input sentences.

In this paper, we show that coherence also takes an important role in single-document summarization. We model the coherence between adjacent sentences in the summary by leveraging the hidden semi-Markov model, which can naturally capture the coherence between sentences.

3 Summarization with Hidden Semi-Markov Model

We first introduce the knapsack problem, which can naturally model single-document summarization. Next, we explain the hidden semi-Markov model and show its relationship to the knapsack problem. Then, we elaborate our summarization method.

3.1 Knapsack Problem

The knapsack problem is a type of combinatorial optimization problem (Korte and Vygen, 2008). Given a set of elements, each of which has a score and size, the problem is formulated as the task of finding the best subset in terms of maximizing the sum of their scores under the size limitation. As mentioned above, single-document summarization can be regarded as an instance of the knapsack problem. The best combination of input sentences can be found by calculating the value of each sentence and packing them into a summary through the dynamic programming knapsack algorithm.

3.2 Hidden Semi-Markov Model

The hidden semi-Markov model (HSMM) is an extension of the hidden Markov model (HMM) (Yu, 2010). In the popular hidden Markov model, each state persists for only one unit length. For example, if a system observes 10 discrete symbols, it outputs 10 hidden states. In the HSMM, each state can persist for some unit lengths through the concept of duration. For example, if a system observes 10 discrete symbols and each state persists for two unit lengths, i.e., their duration is 2, the system outputs 5 hidden states. We show an example in Figure 1. The system observes a sequence consisting of 10 symbols $o_1...o_{10}$ over time $t_1...t_{10}$ and transitions between states $s_1...s_3$. Unlike the basic HMM, states can persist for a non-unit length. In this figure, state s_2 and state s_3 persist for a non-unit length. Hence, the system traverses 6 states even though it observes 10 symbols. This property has been utilized for

sequential tagging, such as named entity recognition (Sarawagi and Cohen, 2004), scene text recognition (Weinman et al., 2008) and phonetic recognition (Kim et al., 2011).

The hidden semi-Markov model is closely related to the knapsack problem. The length, K, of the observed symbols can be regarded as a knapsack constraint. We can consider that the system tries to *pack* the states of the model into the observed sequence of symbols by transitioning over the states under the knapsack constraint so as to maximize the likelihood. Therefore, the hidden semi-Markov can naturally be used for single-document summarization. Suppose that the document to be summarized consists of 10 sentences and the length of each of them is measured by the number of words. In this case, the system transitions over 10 states corresponding to the 10 sentences until it cannot select any further sentence due to the given length requirement. Since each state persists for the length of the corresponding sentence, the remaining length decreases every time the system transitions to a new state.

While an HMM is basically a generative model, Collins (2002) extended it to create a discriminative model. An HSMM can also be extended to become discriminative model (Sarawagi and Cohen, 2004). Our discriminative HSMM learns through the application of max-margin training.

3.3 Formulation

We consider there are n input sentences $s_1, s_2, ..., s_n$. These sentences have lengths $\ell_1, \ell_2, ..., \ell_n$ and weights $w_1, w_2, ..., w_n$. We assume that a sentence that has a high weight should be present in the output summary. We also consider each sentence, s_i, has m_i variants $s_{i,1}, s_{i,2}, ..., s_{i,m}$, each produced by some sort of sentence compression or paraphrase module. These variants also have lengths $\ell_{i,1}, \ell_{i,2}, ..., \ell_{i,m_i}$ and weights $w_{i,1}, w_{i,2}, ..., w_{i,m_i}$. For simplicity, we hereinafter note the original sentences $s_1, s_2, ..., s_n$ as $s_{1,0}, s_{2,0}, ..., s_{n,0}$. Hence we have original sentence $s_{i,0}$ and variants $s_{i,1}, s_{i,2}, ..., s_{i,m}$. Let $s_{0,0}$ and $s_{n+1,0}$ be special symbols indicating the beginning of a document and the end of a document, respectively. We define coherence $c_{g,h,i,j}$ as the coherence between sentence $s_{g,h}$ and sentence $s_{i,j}$. An output summary is described as a sequence of input sentences, g. Let G be the entire set of sequences that can be constructed from the input sentences, i.e., $g \in G$. Finally, let K be the maximum length of the summary desired. With these notations, our proposed method can be formulated as the following optimization problem:

$$g^* = \underset{g \in G}{\operatorname{argmax}} \sum_{s_{i,j} \in sent(g)} w_{i,j} + \sum_{(s_{g,h}, s_{i,j}) \in adj(g)} c_{g,h,i,j} \tag{1}$$

$$s.t. \sum_{s_{i,j} \in sent(g)} \ell_{i,j} \leq K, \tag{2}$$

where $sent(g)$ and $adj(g)$ indicate a set of sentences in g and a set of adjacent sentences in g, respectively. That is, our model tries to find the best sequence of sentences under the knapsack constraint so as to maximize the sum of weights and sentence coherence. In contrast to the common knapsack problem which cannot take the variants and sentence coherence into account, our method, based on the hidden semi-Markov model, does so naturally.

3.4 Parameter Optimization

Here we elaborate how parameters in the model are optimized to achieve the desired summaries. The goal is to determine the value of $w_{i,j}$ for all i, j and $c_{g,h,i,j}$ for all g, h, i, j. We define $w_{i,j}$ and $c_{g,h,i,j}$ as follows:

$$w_{i,j} = \mathbf{w}_w \cdot \mathbf{f}_w(s_{i,j}) \tag{3}$$

$$c_{g,h,i,j} = \mathbf{w}_c \cdot \mathbf{f}_c(s_{g,h}, s_{i,j}), \tag{4}$$

where $\mathbf{f}_w$ and $\mathbf{f}_c$ are d_w-dimensional and d_c-dimensional feature vectors for sentences and sentence pairs, respectively, and $\mathbf{w}_w$ and $\mathbf{w}_c$ are d_w-dimensional and d_c-dimensional parameter vectors for sentences and sentence pairs, respectively. The goal of optimization is to determine the values of both vector $\mathbf{w}_w$ and

$\mathbf{w}_c$, given feature function $\mathbf{f}_w$ and $\mathbf{f}_c$. For simplicity, let $\mathbf{s}$ be a summary, let $\mathbf{f} = \langle \mathbf{f}_w, \mathbf{f}_c \rangle$ be a $(d_w + d_c)$-dimensional feature function for the whole summary and let $\mathbf{w} = \langle \mathbf{w}_w, \mathbf{w}_c \rangle$ be a $(d_w + d_c)$-dimensional weight vector. The value that the objective function outputs for summary $\mathbf{s}$ is $\mathbf{w} \cdot \mathbf{f}(\mathbf{s})$.

To optimize the parameter, we employ the Passive-Aggressive algorithm (Crammer, 2006), a widely-used structured learning method. Since the algorithm offers online learning, it can learn the parameter quickly and is easy to implement. To learn the parameter so that the output summary is optimized to the evaluation criteria popular in document summarization research, ROUGE (Lin, 2004), we introduce ROUGE as the loss function. The parameter is estimated by solving the following formula iteratively[2]:

$$\mathbf{w}^{new} = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2} ||\mathbf{w} - \mathbf{w}^{old}||^2 \tag{5}$$

$$s.t.\ \ \mathbf{w} \cdot \mathbf{f}(\mathbf{r}) - \mathbf{w} \cdot \mathbf{f}(\mathbf{s}) \geq \mathrm{loss}(\mathbf{s}; \mathbf{r}),$$

where $\mathbf{w}^{new}$ is the parameter vector after update, $\mathbf{w}^{old}$ is the parameter vector before update, $\mathbf{r}$ is a reference summary, and loss is the loss function. We define loss as $1 - \mathrm{ROUGE}(\mathbf{s}; \mathbf{r})$. Among the variants of ROUGE, we used ROUGE-1 for the loss function.

3.4.1 Sentence Feature

The features introduced in this section are used to calculate the weights of sentences, $w_{i,j}$.

Term Frequency: Term frequency is a classic feature in document summarization (Luhn, 1958). We calculate the total number of times each content word occurs in the document and then, for each sentence, sum the totals of the content words that appear in the sentence as the value of this feature.
Word: We also use the words and parts-of-speech as features.
Named Entity: Named entities such as a name of person or organization are important. We use named entities and classes as features.
Length: The length of a sentence may indicate the information value of its content. We use the length of a sentence, measured by character number, as a feature.
Position: The position of a sentence is a classically important feature. We use the position of a sentence, the relative position of a sentence, whether the sentence is the first in the document and whether the sentence is the first in a paragraph, the position of the paragraph in which the sentence is, as features.

3.4.2 Coherence Feature

The features introduced in this section are used to calculate sentence coherence, $c_{g,h,i,h}$.

Lexical Transition: Lapata (2003) showed that the structure of the document can be captured by word-pairs consisting of words of two adjacent sentences. We use this feature for capturing the links between two sentences[3]. We build a set of word pairs where one occurs in a precedent sentence and the other occurs in a succeeding one, and use the elements of the set as a feature.
Lexical Cohesion: Pitler et al. (2010) showed that the similarity of two sentences is one of the strongest features for predicting coherence. We reproduce this feature for generating coherent summaries. We calculate cosine similarity between two sentences and use its value as a feature.
Entity Grid: Previous studies showed that Entity Grid (Barzilay and Lapata, 2005) is a strong feature for predicting coherence (Pitler et al., 2010). We also employ this feature for summarization. While the entity vector made from the entity grid was originally defined for whole documents, we build the entity vector for each pair of two sentences because our model is based on the Markovian assumption, and hence the coherence score is defined between two sentences.

[2]As we explain later in Section 5, computation complexity of our algorithm is pseudo-polynomial, and hence the best solution of our model can be located quickly. This is also advantageous in the learning phase because to learn parameters using structured learning, the learner has to generate a summary to calculate the loss. Since our algorithm can quickly find the best solution and generate a summary, it can also contribute to shortening the time required for learning.

[3]It is expected that this feature will also contribute to sentence selection. Barzilay and Elhadad (1997) showed that a closely related word-pair was a good indicator for sentence selection. This feature captures this property by learning.

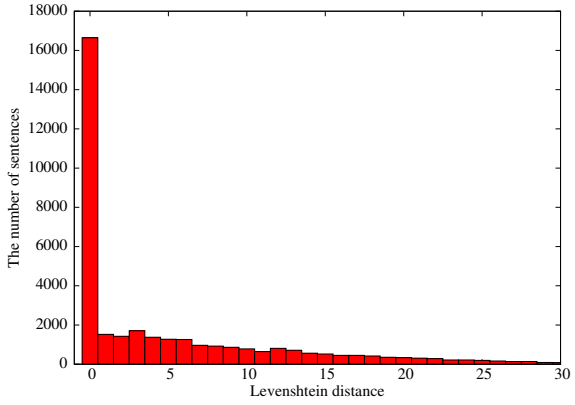


Figure 2: Distribution of Levenshtein distance in the aligned sentences. Among the 36,413 sentences in the references, 16,643 were identical (Levenshtein distance is 0) to the aligned sentences in the input documents.

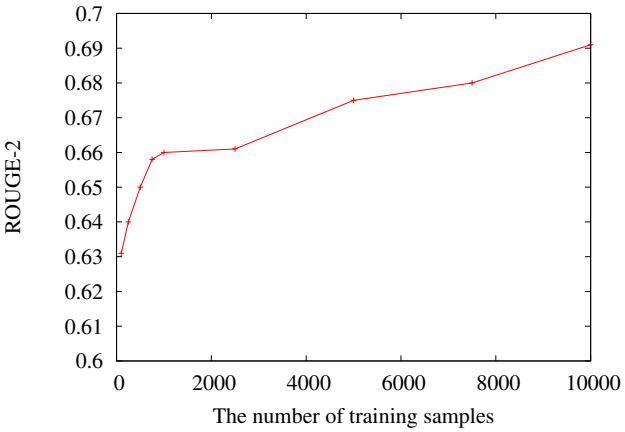


Figure 3: Learning curve of HSMM.

4 Generating Sentence Variants

Since our model can take the variants of an original sentence in the input document as in the multi-candidate reduction framework (Zajic et al., 2007), we incorporate sentence compression.

We generate a few variants of each original sentence by trimming the dependency tree of the sentence; this simple operation is sufficient for reproducing reference summaries. By aligning sentences in a reference summary with those in the corresponding input document[4], we found that human summaries were quite conservative. Among the 36,413 sentences in the references, 16,643 were identical to the aligned sentences in the input documents. Furthermore, most remaining sentences were virtually identical to the original sentences; revisions were minor, and can be reproduced by simple operations. Few sentences exhibited paraphrasing or more sophisticated operations. We plot the distribution of Levenshtein distance in the aligned sentences in Figure 2. According to this observation, we produce the following types of variants by sentence compression:

1. Removing information in parentheses. Some sentences contain parentheses containing additional information for readers. The first type of variant deletes text in parentheses.

2. Shortening sentences by trimming their dependency trees. Basically this method follows the sentence trimmer proposed by Nomoto (2008). While using his method, we keep the predicate and its obligatory arguments in the sentences to keep the sentences grammatical. If a predicate is trimmed, its obligatory arguments are also trimmed and vice versa. Since there are an exponential number of subtrees in one tree, we use only n-best subtrees by ranking them according to n-gram language likelihood and dependency-based language likelihood. We used the dependency parser proposed by Imamura et al (Imamura et al., 2007) to acquire the dependency tree.

5 Decoding with Dynamic Programming

To solve Equation 1 under the constraints of Equation 2, we use dynamic programming. Algorithm 1 shows the pseudo code of the decoding algorithm. Line 1 to Line 7 initializes the variables used in the algorithm. Vector $\mathbf{x} = \langle x_0, ..., x_{n+1} \rangle$ stores which sentence and which variants are included in the output summary. If $x_3 = 2$, $s_{3,2}$ is included in the summary. V, P and S are two-dimensional arrays, each of which is used as a dynamic programming table. They store the process of dynamic programming.

[4]Alignment proceeds in two steps: first, we calculate the Levenshtein distance between sentences in the document and its reference, and then we align sentences so as to minimize the distance between them.

Algorithm 1 Decoding Algorithm: Filling Table

1: $\mathbf{x} = \langle x_0, ..., x_{n+1} \rangle$
2: **for** $i = 0$ to $n + 1$ **do**
3: $x_i = -1$
4: $V[0][i] \leftarrow -1$
5: $P[0][i] \leftarrow -1$
6: $S[0][i] \leftarrow 0$
7: $V[0][0] = 0$
8: **for** $k = 1$ to K **do**
9: **for** $i = 1$ to n **do**
10: $V[k][i] \leftarrow V[k-1][i]$
11: $P[k][i] \leftarrow P[k-1][i]$
12: $S[k][i] \leftarrow S[k-1][i]$
13: **for** $v = 0$ to m_i **do**
14: **if** $\ell_{i,v} \leq k$ **then**
15: **for** $h = 0$ to $i - 1$ **do**
16: $u = V[k - \ell_{i,v}][h]$
17: **if** $u \neq -1 \wedge S[k - \ell_{i,v}][h] + w_{i,v} + c_{h,u,i,v} \geq S[k][i]$ **then**
18: $V[k][i] \leftarrow v$
19: $P[k][i] \leftarrow h$
20: $S[k][i] \leftarrow S[k - \ell_{i,v}][h] + w_{i,v} + c_{h,u,i,v}$
21: $V[K+1][n+1] \leftarrow 0$
22: $P[K+1][n+1] \leftarrow 0$
23: $S[K+1][n+1] \leftarrow 0$
24: **for** $h = 1$ to n **do**
25: $u = V[K][h]$
26: **if** $S[K][h] + c_{h,u,n+1,0} \geq S[K+1][n+1]$ **then**
27: $P[K+1][n+1] \leftarrow h$
28: $S[K+1][n+1] \leftarrow S[K][h] + c_{h,u,n+1,0}$

	Document	Reference
Avg. # of characters	476.2	142.0
Avg. # of words	298.6	88.3
Avg. # of sentences	9.7	2.9

Table 1: The statistics of our corpus.

$V[k][i]$ stores which variants are used at time k, i. If $V[k][i] = 0$, original sentence $s_{i,0}$ is selected at time k, i. If $V[k][i] = -1$, no sentence is selected at time k, i. $P[k][i]$ stores a pointer to the sentence connected to the front of the current sentence. $S[k][i]$ stores the value of the objective function at time k, i. Line 8 to Line 36 locates the best sequence of sentences based on the following recurrence formula:

$$S[k][i] = \begin{cases} \max_{h=0...i-1, v=0...m} S[k - \ell_{i,v}][h] + w_{i,v} + c_{h,V[k-\ell_{i,v}][h],i,v} & \text{(A)} \\ S[k-1][i] & \text{(B)}, \end{cases} \quad (6)$$

where case A is: $\ell_{i,v} \leq k \ \wedge S[k-1][i] \leq S[k - \ell_{i,v}][h] + w_{i,v} + c_{h,V[k-\ell_{i,v}][h],i,v}$ and case B is: $otherwise$. This recurrence formula means that at time k, i the best variant to be selected as can be determined at time $k - \ell_{i,v}, h$. Hence, for all $k \in 1...K$ and $i \in 1...n$, the algorithm finds the best sequence of sentences at time k, i. After Algorithm 1 locates the best sequence of sentences by filling the tables, the best sequence can be restored by backtracing along the pointers stored in P. Finally, the algorithm outputs $\mathbf{x}$, which stores which sentences and variants are used in the best sequence. Since this algorithm is based on a dynamic programming knapsack algorithm (Korte and Vygen, 2008), it runs in pseudo-polynomial time. This is a significant advantage over the methods that rely on integer linear programming solvers due to their substantial computation cost.

6 Experiments

6.1 Data

We prepared 12,748 pairs of Japanese newspaper articles and their manually-written reference summaries. This is one of the largest corpus available for single-document summarization research. The length of all references is within 150 characters. All references in the corpus were written by a specialist staff in a Japanese newspaper company and the company sold these summaries for commercial purposes.

We list the statistics of our corpus in Table 1. As shown, the task is to summarize the document in about a third of its original length in terms of the number of words.

6.2 Evaluation Criteria

ROUGE; ROUGE is an automatic evaluation method for automatic summarization proposed by Lin (2004). We used ROUGE-1 and ROUGE-2 to evaluate the summaries. Since our document-reference pairs are written in Japanese, we segmented the sentences into words using the Japanese morphological analyzer developed by Fuchi and Takagi (1998). When calculating the ROUGE score, we used only content words (i.e. nouns, verbs and adjectives) and so excluded function words as stop words.
Linguistic Quality: To evaluate the linguistic quality of the summaries generated by our method, we performed a manual evaluation according to quality questions proposed by the National Institute of Standards and Technology (NIST) (2007)[5]. We randomly sampled 100 summaries from the outputs of each method described below and asked 7 subjects to evaluate the summaries according to the questions. All subjects were Japanese native and none were among the authors. Since the quality questions by NIST (2007) were designed for multi-document summarization, we used 3 of the 5 NIST questions for single-document summarization: grammaticality, referential clarity, and structure/coherence. We also asked the subjects to evaluate overall summary quality.

6.3 Compared Methods

We compared the following 8 methods.

Random: Random method selects sentences in the input document randomly.
Lead: Lead method is a classic baseline in single-document summarization. It only extracts the words from the beginning of the document until the extracted words reach the given length. We simply extracted 150 characters from the beginning of each document.
Knapsack: The knapsack problem can be used as a single-document summarization model (McDonald, 2007). In this baseline, the weight of each sentence was calculated based on the average probabilities of the words in the sentence (Nenkova and Vanderwende, 2005). Then, a summary was generated by solving the knapsack problem.
Knapsack with Supervision: Instead of the average word probabilities used in the above baseline, we used only sentence features $\mathbf{f}_w$ to weigh a sentence.
Conditional Random Fields: Conditional random fields can be used to weigh sentences (Shen et al., 2007). Since CRFs required binary labels in learning, we aligned sentences in an input document with the sentences in its reference as explained in Section 4. We used the probabilities of sentences from CRFs as the weights of the knapsack problem.
Hidden Semi-Markov Model: This is our proposed method without variants of the original sentences. It selected sentences only from the set of original sentences.
Hidden Semi-Markov Model with Compression: This is our proposed method with variants of the original sentences. It selected from among the variants and the original ones.
Human: In the linguistic quality evaluation, we added references to the summaries generated by the above methods to show the upper bound.

When learning, we did 10-fold cross validation. In the experiments, statistical significance was checked by Wilcoxon signed-rank test (Wilcoxon, 1945). To counteract the problem of multiple comparisons, we used the Holm-Bonferroni method (Holm, 1979) to adjust the significance level, α.

7 Results and Discussion

We show the results of our experiment in Table 2 and Table 3. In this section, first we discuss the results of the ROUGE evaluation, and then we discuss the results of the linguistic quality evaluation.

In the ROUGE evaluation, all the compared methods except for RANDOM showed good performance. This is because, as shown in Section 4, many references consisted of sentences identical to the original

[5]Some recent studies have tried to predict the readability of the text automatically (Pitler et al., 2010).

Method	R-1	R-2	Idt.
RANDOM	0.417	0.291	1.2%
LEAD	$0.779^{\mathrm{C,S,U,R}}$	$0.727^{\mathrm{C,S,U,R}}$	4.4%
KP	0.704^{R}	0.611^{R}	9.3%
KP(S)	$0.729^{\mathrm{U,R}}$	$0.647^{\mathrm{U,R}}$	10.4%
CRFs	$0.741^{\mathrm{U,R}}$	$0.675^{\mathrm{S,U,R}}$	11.3%
HSMM	$0.769^{\mathrm{C,S,U,R}}$	$0.703^{\mathrm{C,S,U,R}}$	15.2%
HSMM(C)	$0.785^{\mathrm{C,S,U,R}}$	$0.722^{\mathrm{C,S,U,R}}$	20.4%

Table 2: Results of the ROUGE evaluation. "R-1" and "R-2" correspond to ROUGE-1 and ROUGE-2, respectively. The values in the column of "Idt." are the percentage of summaries completely-identical to the corresponding references. In the table, $^{\mathrm{C,S,U,L,R}}$ indicate statistical significance against CRFs, KP(S), KP, LEAD, RANDOM, respectively.

Method	Gram.	Ref.	S./C.	Overall
LEAD	1.9	3.9	2.5	2.1
KP	4.1^{L}	3.7	3.4	3.5
KP(S)	4.2^{L}	3.6	3.5	3.6^{L}
CRFs	4.1^{L}	3.9	3.7^{L}	3.6^{L}
HSMM	4.3^{L}	4.0	4.1^{L}	4.0^{L}
HSMM(C)	4.0^{L}	3.9	4.0^{L}	3.9^{L}
HUMAN	4.7^{L}	4.5	4.7^{L}	4.8^{L}

Table 3: Results of the linguistic quality evaluation. The values ranged from 1 (very poor) to 5 (very good) (National Institute of Standards and Technology, 2007). We show statistical significance with the same notations as Table 2.

ones, and hence the references can be reproduced if important sentences are identified. Since the compression rate in our corpus was relatively light, it made important information easy to identify. Among the compared methods, both LEAD and our proposed method, HSMM(C), achieved the best result. There was no significant difference between LEAD and HSMM(C). This surprising performance of LEAD was due to the ROUGE evaluation. The words in the document leads were likely to be important, and LEAD drew on this property. However, as we mentioned later, it sacrificed the linguistic quality to achieve the high ROUGE score. Furthermore, it failed to yield summaries identical to the reference. In contrast to LEAD, almost 20% of the summaries generated by HSMM(C) were identical to the references. This shows that our method successfully mimicked human assessments. HSMM followed the best models. There was a statistically significant difference between HSMM(C) and HSMM. Since some sentences, especially the first sentence in the document, were long and the first sentence was particularly important to summarize the document, sentence compression yielded a significant improvement. As shown in Table 2, employing compression greatly improved the percentage of identical summaries. HSMM significantly outperformed all of the baseline extractive methods except LEAD. While CRFs can take advantage of all features used in HSMM, CRFs cannot take the evaluation measure such as ROUGE and the knapsack constraint into account in learning. HSMM also significantly outperformed KP(S). This difference is particularly important, and shows the usefulness of features related to coherence. While KP(S) used only features about sentences, HSMM successfully mimicked the references as it drew on the features related to coherence.

We show the learning curve of HSMM in Figure 3. We fixed 2,748 pairs for testing, and learned parameters from 100, 250, 500, 1,000, 2,500, 5,000, 7,500 and 10,000 pairs. The curve in the figure clearly shows the effectiveness of our large-scale corpus in learning. It seems that the curve does not saturate and hence HSMM performance can be improved by more training samples. As in the results recently shown by Filippova (2013), this result implies that large-scale data is important in the field of document summarization as in other fields of computational linguistics. Past studies in document summarization relied on relatively small datasets consisting of a few dozen or at most a few hundred pairs of a document and its reference in learning. In contrast to the past studies, there are over 10,000 pairs in our dataset and the results show its effectiveness.

Second, we discuss the result of the linguistic quality evaluation. Unlike the ROUGE evaluation, HSMM achieved the best result. As previous studies have pointed out (Nenkova and McKeown, 2011), sentence compression commonly tends to degrade the linguistic quality of a summary while improving its content. As shown in Table 3, the grammaticality of HSMM(C) is lower than that of HSMM, but the

difference is not significant. Although we could not observe any significant difference between HSMM and other extractive baselines, our proposals, HSMM and HSMM(C), yielded the best result in terms of structure/coherence. By making use of the features related to coherence, we successfully improved summary quality. In contrast to the surprising performance of LEAD in the ROUGE evaluation, in the linguistic quality evaluation, LEAD yielded the worst performance. Since LEAD had to cut the sentences when it reached the given length, it create ungrammatical fragments.

Finally, we touch on the balance between the quality of content and linguistic quality. Comparing Table 2 to 3, we can see the correlation between the quality of content and linguistic quality. This result is reasonable because we can extract much more information from grammatical and well-organized sentences. Although we optimized the parameter to maximize the ROUGE score, it also yielded improvements in linguistic quality. This is because the manually-generated reference summaries are basically grammatical and well-organized and the parameter is learnt to mimic them. However, there is an inherent trade-off between the quality of content and linguistic quality. For example, under stricter length limitations, instead of cohesive devices such as conjunctions, which can improve the coherence of sentences, content words would be preferred for summary inclusion to augment information. Balancing them to maximize reader satisfaction is an interesting problem.

8 Conclusions

In this paper we presented a novel single-document summarization method based on the hidden semi-Markov model, which is a natural extension of the knapsack problem. Our model naturally takes account of sentence context when identifying important sentences. This property is particularly important to ensure the coherence of output summaries and to produce informative and linguistically high-quality summaries. We also proposed an algorithm based on dynamic programming so the best solution can be located quickly. Experiments on a very large-scale single-document summarization corpus showed that our proposed method significantly outperforms competitive baselines.

As future work, we plan to tackle on the summarization task where higher compression is demanded. To generate shorter summaries, we plan to employ more sophisticated approaches, such as paraphrasing.

Acknowledgement

The corpus used in this paper is owned by The Mainichi Newspapers Co., Ltd. and is leased to Nippon Telegraph and Telephone Corporation. We sincerely appreciate their consideration. We also appreciate the insightful comments from reviewers. Their comments greatly improved the quality of this paper.

References

Ernst Althaus, Nikiforos Karamanis, and Alexander Koller. 2004. Computing locally coherent discourses. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL)*, pages 399–406.

Regina Barzilay and Michael Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS)*, pages 10–17.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: an entity-based approach. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL)*, pages 141–148.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL 2004: Main Proceedings*, pages 113–120.

Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173.

James Clarke and Mirella Lapata. 2007. Modelling compression with discourse constraints. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1–11.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–8.

Koby Crammer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(Mar):551–585.

Hal Daume, III and Daniel Marcu. 2002. A noisy-channel model for document compression. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 449–456.

Katja Filippova. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1481–1491.

Takeshi Fuchi and Shinichiro Takagi. 1998. Japanese morphological analyzer using word co-occurrence: Jtag. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (ACL-COLING)*, pages 409–413.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: a framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. 2013. Single-document summarization as a tree knapsack problem. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1515–1520.

Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70.

Kenji Imamura, Genichiro Kikui, and Norihito Yasuda. 2007. Japanese dependency parsing using sequential labeling for semi-spoken language. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 225–228.

Nikiforos Karamanis, Massimo Poesio, Chris Mellish, and Jon Oberlander. 2004. Evaluating centering-based metrics of coherence. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL)*, pages 391–398.

Sungwoong Kim, Sungrack Yun, and Chang D. Yoo. 2011. Large margin discriminative semi-markov model for phonetic recognition. *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, 7(19):1999–2012.

Bernhard Korte and Jens Vygen. 2008. *Combinatorial Optimization*. Springer-Verlag, third edition.

John Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 282–289.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 545–552.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of ACL Workshop Text Summarization Branches Out*, pages 74–81.

Annie Louis and Ani Nenkova. 2012. A coherence model based on syntactic patterns. In *Proceedings of the 2012 Conference on Empirical Methods on Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Hans P. Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 22(2):159–165.

William C. Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 1997. From discourse structure to text summaries. In *Proceedings of ACL/EACL 1997 Summarization Workshop*, pages 82–88.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European Conference on Information Retrieval (ECIR)*, pages 557–564.

National Institute of Standards and Technology. 2007. The linguistic quality questions. http://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt.

Ani Nenkova and Kathleen McKeown. 2011. *Automatic Summarization*. Now Publishers.

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. Technical report, MSR-TR-2005-101.

Ani Nenkova, Jieun Chae, Annie Louis, and Emily Pitler. 2010. Structural features for predicting the linguistic quality of text: Applications to machine translation, automatic summarization and human-authored text. In Emiel Krahmer and Theunem Mariet, editors, *Empirical Methods in Natural Language Generation: Data-oriented Methods and Empirical Evaluation*, pages 222–241. Springer.

Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Genichiro Kikui. 2010. Opinion summarization with integer linear programming formulation for sentence extraction and ordering. In *Coling 2010: Posters*, pages 910–918.

Tadashi Nomoto. 2008. A generic sentence trimmer with crfs. In *Proceedings of the 46th Annual Conference of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 299–307.

Naoaki Okazaki, Yutaka Matsuo, and Mitsuru Ishizuka. 2004. Improving chronological sentence ordering by precedence relation. In *Proceedings of the 20th International Conference on Computational Linguistics (Coling)*, pages 750–756.

Emily Pitler, Annie Louis, and Ani Nenkova. 2010. Automatic evaluation of linguistic quality in multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 544–554.

Sunita Sarawagi and William W. Cohen. 2004. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems 17*, pages 1185–1192.

Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. 2007. Document summarization using conditional random fields. In *Proceedings of the 20th international joint conference on Artifical intelligence (IJCAI)*, pages 2862–2867.

Jerod J. Weinman, Erik Learned-Miller, and Allen Hanson. 2008. A discriminative semi-markov model for robust scene text recognition. In *Proceedings of the 19th International Conference on Pattern Recognition (ICPR)*, pages 1–5.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Shun-Zheng Yu. 2010. Hidden semi-markov models. *Artificial Intelligence*, 174(2):215–243.

David M. Zajic, Bonnie J. Dorr, Jimmy Lin, and Schwartz Richard. 2007. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing and Management*, 43:1549–1570.

Query-Focused Opinion Summarization for User-Generated Content

Lu Wang[1] **Hema Raghavan**[2] **Claire Cardie**[1] **Vittorio Castelli**[3]
[1]Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
{luwang, cardie}@cs.cornell.edu
[2]LinkedIn, CA, USA
hraghavan@linkedin.com
[3]IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA
vittorio@us.ibm.com

Abstract

We present a submodular function-based framework for query-focused opinion summarization. Within our framework, relevance ordering produced by a statistical ranker, and information coverage with respect to topic distribution and diverse viewpoints are both encoded as submodular functions. Dispersion functions are utilized to minimize the redundancy. We are the first to evaluate different metrics of text similarity for submodularity-based summarization methods. By experimenting on community QA and blog summarization, we show that our system outperforms state-of-the-art approaches in both automatic evaluation and human evaluation. A human evaluation task is conducted on Amazon Mechanical Turk with scale, and shows that our systems are able to generate summaries of high overall quality and information diversity.

1 Introduction

Social media forums, such as social networks, blogs, newsgroups, and community question answering (QA), offer avenues for people to express their opinions as well collect other people's thoughts on topics as diverse as health, politics and software (Liu et al., 2008). However, digesting the large amount of information in long threads on newsgroups, or even knowing which threads to pay attention to, can be overwhelming. A text-based summary that highlights the diversity of opinions on a given topic can lighten this information overload. In this work, we design a submodular function-based framework for opinion summarization on community question answering and blog data.

Question: What is the long term effect of piracy on the music and film industry?

Best Answer: Rising costs for movies and music. ... If they sell less, they need to raise the price to make up for what they lost. The other thing will be music and movies with less quality. ...
Other Answers:
Ans1: Its bad... really bad. (Just watch this movie and you will find out ... Piracy causes rappers to appear on your computer).
Ans2: By removing the profitability of music & film companies, piracy takes away their motivation to produce new music & movies. If they can't protect their copyrights, they can't continue to do business. ...
Ans4: *It is forcing them to rework their business model, which is a good thing.* In short, I don't think the music industry in particular will ever enjoy the huge profits of the 90's. ...
Ans6: Please-People in those businesses make millions of dollars as it is!! I don't think piracy hurts them at all!!!

Figure 1: Example discussion on Yahoo! Answers. Besides the best answer, other answers also contain relevant information (in *italics*). For example, the sentence in blue has a contrasting viewpoint compared to the other answers.

Opinion summarization has previously been applied to restricted domains, such as product reviews (Hu and Liu, 2004; Lerman et al., 2009) and news (Stoyanov and Cardie, 2006), where the output summary is either presented in a structured way with respect to each aspect of the product or organized along contrastive viewpoints. Unlike those works, we address user generated online data: community QA and blogs. These forums use a substantially less formal language than news articles, and at the same time address a much broader spectrum of topics than product reviews. As a result, they present new challenges for automatic summarization. For example, Figure 1 illustrates a sample question from Yahoo! Answers[1] along with the answers from different users. The question receives more than one answer, and one of them is selected as the "best answer" by the asker or other participants. In general, answers from other users also provide relevant information. While community QA successfully pools rich knowledge from the wisdom of the crowd, users might need to seine through numerous posts to extract the information

[1]http://answers.yahoo.com/

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1660–1669, Dublin, Ireland, August 23-29 2014.

they need. Hence, it would be beneficial to summarize answers automatically and present the summaries to users who ask similar questions in the future. In this work, we aim to return a summary that encapsulates different perspectives for a given opinion question and a set of relevant answers or documents.

In our work we assume that there is a central topic (or query) on which a user is seeking diverse opinions. We predict query-relevance through automatically learned statistical rankers. Our ranking function not only aims to find sentences that are on the topic of the query but also ones that are "opinionated" through the use of several features that indicate subjectivity and sentiment. The relevance score is encoded in a submodular function. Diversity is accounted for by a dispersion function that maximizes the pairwise distance between the pairs of sentences selected.

Our chief contributions are:
(1) We develop a submodular function-based framework for query-focused opinion summarization. To the best of our knowledge, this is the first time that submodular functions have been used to support opinion summarization. We test our framework on two tasks: summarizing opinionated sentences in community QA (Yahoo! Answers) and blogs (TAC-2008 corpus). Human evaluation using Amazon Mechanical Turk shows that our system generates the best summary 57.1% of the time. On the other hand, the best answer picked by Yahoo! users is chosen only 31.9% of the time. We also obtain significant higher Pyramid F1 score on the blog task as compared to the system of Lin and Bilmes (2011).
(2) Within our summarization framework, the statistically learned sentence relevance is included as part of our objective function, whereas previous work on submodular summarization (Lin and Bilmes, 2011) only uses ngram overlap for query relevance. Additionally, we use Latent Dirichlet Allocation (Blei et al., 2003) to model the topic structure of the sentences, and induce clusterings according to the learned topics. Therefore, our system is capable of generating summaries with broader topic coverage.
(3) Furthermore, we are the first to study how different metrics for computing text similarity or dissimilarity affect the quality of submodularity-based summarization methods. We show empirically that lexical representation-based similarity, such as TFIDF scores, uniformly outperforms semantic similarity computed with WordNet. Moreover, when measuring the summary diversity, topical representation is marginally better than lexical representation, and both of them beats semantic representation.

2 Related Work

Our work falls in the realm of query-focused summarization, where a user asks a question and the system generates a summary of the answers containing pertinent and diverse information. A wide range of methods have been investigated, where relevance is often estimated through TF-IDF similarity (Carbonell and Goldstein, 1998), topic signature words (Lin and Hovy, 2000) or by learning a Bayesian model over queries and documents (Daumé and Marcu, 2006). Most work only implicitly penalizes summary redundancy, e.g. by downweighting the importance of words that are already selected.

Encouraging diversity of a summary has recently been addressed through submodular functions, which have been applied for multi-document summarization in newswire (Lin and Bilmes, 2011; Sipos et al., 2012), and comments summarization (Dasgupta et al., 2013). However, these works either ignore the query information (when available) or else use simple ngram matching between the query and sentences. In contrast, we propose to optimize an objective function that addresses both relevance and diversity.

Previous work on generating opinion summaries mainly considers product reviews (Hu and Liu, 2004; Lerman et al., 2009), and formal texts such as news articles (Stoyanov and Cardie, 2006) or editorials (Paul et al., 2010). Mostly, there is no query information, and summaries are formulated in a structured way based on product features or contrastive standpoints. Our work is more related to opinion summarization on user-generated content, such as community QA. Liu et al. (2008) manually construct taxonomies for questions in community QA. Summaries are generated by clustering sentences according to their polarity based on a small dictionary. Tomasoni and Huang (2010) introduce coverage and quality constraints on the sentences, and utilize an integer linear programming framework to select sentences.

3 Submodular Opinion Summarization

In this section, we describe how query-focused opinion summarization can be addressed by submodular functions combined with dispersion functions. We first define our problem. Then we introduce the

Basic Features	Sentiment Features
- answer position in all answers/sentence position in blog - length of the answer/sentence - length is less than 5 words	- number/portion of sentiment words from a lexicon (Section 3.2) - if contains sentiment words with the same polarity as sentiment words in query
Query-Sentence Overlap Features	**Query-Independent Features**
- unigram/bigram TF/TFIDF similarity with query - number of key phrases in the query that appear in the sentence. A model similar to that described in (Luo et al., 2013) was applied to detect key phrases.	- unigram/bigram TFIDF similarity with cluster centroid - sumBasic score (Nenkova and Vanderwende, 2005) - number of topic signature words (Lin and Hovy, 2000) - JS divergence with cluster

Table 1: Features used for candidate ranking. We use them for ranking answers in both community QA and blogs.

components of our objective function (Sections 3.1–3.3). The full objective function is presented in Section 3.4. Lastly, we describe a greedy algorithm with constant factor approximation to the optimal solution for generating summaries (Section 3.5).

A set of documents or answers to be summarized are first split into a set of individual sentences $V = \{s_1, \cdots, s_n\}$. Our problem is to select a subset $S \subseteq V$ that maximizes a given objective function $f : 2^V \rightarrow \mathbb{R}$ within a length constraint: $S^* = \arg\max_{S \subseteq V} f(S)$, subject to $\mid S \mid \leq c$. $\mid S \mid$ is the length of the summary S, and c is the length limit.

Definition 1 *A function $f : 2^V \rightarrow \mathbb{R}$ is submodular iff for all $s \in V$ and every $S \subseteq S' \subseteq V$, it satisfies $f(S \cup \{s\}) - f(S) \geq f(S' \cup \{s\}) - f(S')$.*

Previous submodularity-based summarization work assumes this diminishing return property makes submodular functions a natural fit for summarization and achieves state-of-the-art results on various datasets. In this paper, we follow the same assumption and work with non-decreasing submodular functions. Nevertheless, they have limitations, one of which is that functions well suited to modeling diversity are not submodular. Recently, Dasgupta et al. (2013) proved that diversity can nonetheless be encoded in well-designed *dispersion functions* which still maintain a constant factor approximation when solved by a greedy algorithm.

Based on these considerations, we propose an objective function $f(S)$ mainly considering three aspects: *relevance* (Section 3.1), *coverage* (Section 3.2), and *non-redundancy* (Section 3.3). Relevance and coverage are encoded in a non-decreasing submodular function, and non-redundancy is enforced by maximizing the dispersion function.

3.1 Relevance Function

We first utilize statistical rankers to produce a preference ordering of the candidate answers or sentences. We choose ListNet (Cao et al., 2007), which has been shown to be effective in many information retrieval tasks, as our ranker. We use the implementation from Ranklib (Dang, 2011).

Features used in the ranking algorithm are summarized in Table 1. All features are normalized by standardization. Due to the length limit, we cannot provide the full results on feature evaluation. Nevertheless, we find that ranking candidates by TFIDF similarity or key phrases overlapping with the query can produce comparable results with using the full feature set (see Section 5).

We take the ranks output by the ranker, and define the relevance of the current summary S as: $r(S) = \sum_i^{|S|} \sqrt{rank_i^{-1}}$, where $rank_i$ is the rank of sentence s_i in V. For QA answer ranking, sentences from the same answer have the same ranking. The function $r(S)$ is our first submodular function.

3.2 Coverage Functions

Topic Coverage. This function is designed to capture the idea that a comprehensive opinion summary should provide thoughts on distinct aspects. Topic models such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and its variants are able to discover hidden topics or aspects of document collections, and thus afford a natural way to cluster texts according to their topics. Recent work (Xie and Xing, 2013) shows the effectiveness of utilizing topic models for newsgroup document clustering. We first learn an LDA model from the data, and treat each topic as a cluster. We estimate a sentence-topic distribution $\vec{\theta}$ for each sentence, and assign the sentence to the cluster k corresponding to the mode of the distribution (i.e., $k = \arg\max_i \theta_i$). This naive approach produces comparable clustering performance to the state-of-the-art according to (Xie and Xing, 2013). $\mathcal{T}$ is defined as the clustering induced by our algorithm on the set V. The topic coverage of the current summary S is defined as $t(S) = \sum_{T \in \mathcal{T}} \sqrt{|S \cap T|}$.

From the concavity of the square root it follows that sets S with uniform coverages of topics are preferred to sets with skewed coverage.

Authorship Coverage. This term encourages the summarization algorithm to select sentences from different authors. Let $\mathcal{A}$ be the clustering induced by the sentence to author relation. In community QA, sentences from the answers given by the same user belong to the same cluster. Similarly, sentences from blogs with the same author are in the same cluster. The authorship score is defined as $a(S) = \sum_{A \in \mathcal{A}} \sqrt{|S \cap A|}$.

Polarity Coverage. The polarity score encourages the selection of summaries that cover both positive and negative opinions. We categorize each sentence simply by counting the number of polarized words given by our lexicon. A sentence belongs to a positive cluster if it has more positive words than negative ones, and vice versa. If any negator co-occurs with a sentiment word (e.g. within a window of size 5), the sentiment is reversed.[2] The polarity clustering $\mathcal{P}$ thus have two clusters corresponding to positive and negative opinions. The score is defined as $p(S) = \sum_{P \in \mathcal{P}} \sqrt{| S \cap P |}$. Our lexicon consists of MPQA lexicon (Wilson et al., 2005), General Inquirer (Stone et al., 1966), and SentiWordNet (Esuli and Sebastiani, 2006). Words with conflicting sentiments from different lexicons are removed.

Content Coverage. Similarly to Lin and Bilmes (2011) and Dasgupta et al. (2013), we use the following function to measure content coverage of the current summary S: $c(S) = \sum_{v \in V} \min(cov(v, S), \theta \cdot cov(v, V))$, where $cov(v, S) = \sum_{u \in S} sim(v, u)$. We experiment with two types of similarity functions. One is a Cosine TFIDF similarity score. The other is a WordNet-based semantic similarity score between pairwise dependency relations from two sentences (Dasgupta et al., 2013). Specifically, $sim_{Sem}(v, u) = \sum_{rel_i \in v, rel_j \in u} WN(a_i, a_j) \times WN(b_i, b_j)$, where $rel_i = (a_i, b_i), rel_j = (a_j, b_j)$, $WN(w_i, w_j)$ is the shortest path length. All scores are scaled onto $[0, 1]$.

3.3 Dispersion Function

Summaries should contain as little redundant information as possible. We achieve this by adding an additional term to the objective function, encoded by a dispersion function. Given a set of sentences S, a complete graph is constructed with each sentence in S as a node. The weight of each edge (u, v) is their dissimilarity $d'(u, v)$. Then the distance between any pair of u and v, $d(u, v)$, is defined as the total weight of the shortest path connecting u and v.[3] We experiment with two forms of dispersion function (Dasgupta et al., 2013): (1) $h_{sum} = \sum_{u,v \in V, u \neq v} d(u, v)$, and (2) $h_{min} = \min_{u,v \in V, u \neq v} d(u, v)$.

Then we need to define the dissimilarity function $d'(\cdot, \cdot)$. There are different ways to measure the dissimilarity between sentences (Mihalcea et al., 2006; Agirre et al., 2012). In this work, we experiment with three types of dissimilarity functions.

Lexical Dissimilarity. This function is based on the well-known Cosine similarity score using TFIDF weights. Let $sim_{tfidf}(u, v)$ be the Cosine similarity between u and v, then we have $d'_{Lex}(u, v) = 1 - sim_{tfidf}(u, v)$.

Semantic Dissimilarity. This function is based on the semantic meaning embedded in the dependency relations. $d'_{Sem}(u, v) = 1 - sim_{Sem}(v, u)$, where $sim_{Sem}(v, u)$ is the semantic similarity used in content coverage measurement in Section 3.2.

Topical Dissimilarity. We propose a novel dissimilarity measure based on topic models. Celikyilmaz et al. (2010) show that estimating the similarity between query and passages by using topic structures can help improve the retrieval performance. As discussed in the topic coverage in Section 3.2, each sentence is represented by its sentence-topic distributions estimated by LDA. For candidate sentence u and v, let their topic distributions be P_u and P_v. Then the dissimilarity between u and v can be defined as: $d'_{Topic}(u, v) = JSD(P_u || P_v) = \frac{1}{2}(\sum_i P_u(i) \log_2 \frac{P_u(i)}{P_a(i)} + \sum_i P_v(i) \log_2 \frac{P_v(i)}{P_a(i)})$ where $P_a(i) = \frac{1}{2}(P_u(i) + P_v(i))$.

3.4 Full Objective Function

The objective function takes the interpolation of the submodular functions and dispersion function:

$$\mathcal{F}(S) = r(S) + \alpha t(S) + \beta a(S) + \gamma p(S) + \eta c(S) + \delta h(S). \quad (1)$$

[2] There exists a large amount of work on determining the polarity of a sentence (Pang and Lee, 2008) which can be employed for polarity clustering in this work. We decide to focus on summarization, and estimate sentence polarity through sentiment word summation (Yu and Hatzivassiloglou, 2003), though we do not distinguish different sentiment words.

[3] This definition of distance is used to produce theoretical guarantees for the greedy algorithm described in Section 3.5.

The coefficients $\alpha, \beta, \gamma, \eta, \delta$ are non-negative real numbers and can be tuned on a development set.[4] Notice that each summand except $h(S)$ is a non-decreasing, non-negative, and submodular function, and summation preserves monotonicity, non-negativity, and submodularity. Dispersion function $h(s)$ is either h_{sum} or h_{min} as introduced previously.

3.5 Summary Generation via Greedy Algorithm

Generating the summary that maximizes our objective function in Equation 1 is NP-hard (Chandra and Halldórsson, 1996). We choose to use a greedy algorithm that guarantees to obtain a constant factor approximation to the optimal solution (Nemhauser et al., 1978; Dasgupta et al., 2013). Concretely, starting with an empty set, for each iteration, we add a new sentence so that the current summary achieves the maximum value of the objective function. In addition to the theoretical guarantee, existing work (McDonald, 2007) has empirically shown that classical greedy algorithms usually works near-optimally.

4 Experimental Setup

4.1 Opinion Question Identification

We first build a classifier to automatically detect opinion oriented questions in Community QA; questions in the blog dataset are all opinionated. Our opinion question classifier is trained on two opinion question datasets: (1) the first, from Li et al. (2008a), contains 646 opinionated and 332 objective questions; (2) the second dataset, from Amiri et al. (2013), consists of 317 implicit opinion questions, such as "*What can you do to help environment?*", and 317 objective questions. We train a RBF kernel based SVM classifier to identify opinion questions, which achieves F1 scores of 0.79 and 0.80 on the two datasets when evaluated using 10-fold cross-validation (the best F1 scores reported are 0.75 and 0.79).

4.2 Datasets

Community QA Summarization: Yahoo! Answers. We use the Yahoo! Answers dataset from Yahoo! *Webscope*^{TM} program,[5] which contains 3,895,407 questions. We first run the opinion question classifier to identify the opinion questions. For summarization purpose, we require each question having at least 5 answers, with the average length of answers larger than 20 words. This results in 130,609 questions.

To make a compelling task, we reserve questions with an average length of answers larger than 50 words as our test set for both ranking and summarization; all the other questions are used for training. As a result, we have 92,109 questions in the training set for learning the statistical ranker, and 38,500 in the test set. The category distribution of training and test questions (Yahoo! Answers organizes the questions into predefined categories) are similar. 10,000 questions from the training set are further reserved as the development set. Each question in the Yahoo! Answers dataset has a user-voted best answer. These best answers are used to train the statistical ranker that predicts relevance. Separate topic models are learned for each category, where the category tag is provided by Yahoo! Answer.

Blog Summarization: TAC 2008. We use the TAC 2008 corpus (Dang, 2008), which consists of 25 topics. 23 of them are provided with human labeled nuggets, which TAC used in human evaluation. TAC also provides snippets (i.e., sentences) that are frequently retrieved by participant systems or identified as relevant by human annotators. We do not assume those snippets are known to any of our systems.

4.3 Comparisons

For both opinion summarization tasks, we compare with (1) the approach by Dasgupta et al. (2013), and (2) the systems from Lin and Bilmes (2011) with and without query information. The sentence clustering process in Lin and Bilmes (2011) is done by using CLUTO (Karypis, 2003). For the implementation of systems in Lin and Bilmes (2011) and Dasgupta et al. (2013), we always use the parameters reported to have the best performance in their work.

For cQA summarization, we use the **best answer** voted by the user as a baseline. Note that this is a strong baseline since all the other systems are unaware of which answer is the best. For blog summarization, we have three additional baselines – the **best systems** in TAC 2008 (Kim et al., 2008; Li et al., 2008b), top sentences returned by our **ranker**, a baseline produced by TFIDF similarity and a lexicon

[4] The values for the coefficients are $5.0, 1.0, 10.0, 5.0, 10.0$ for $\alpha, \beta, \gamma, \eta, \delta$, respectively, as tuned on the development set.

[5] `http://sandbox.yahoo.com/`

(henceforth called **TFIDF+Lexicon**). In TFIDF+Lexicon, sentences are ranked by the TFIDF similarity with the query, and then sentences with sentiment words are selected in sequence. This baseline aims to show the performance when we only have access to lexicons without using a learning algorithm.

5 Results

5.1 Evaluating the Ranker

We evaluate our ranker (described in Section 3.1) on the task of best answer prediction. Table 2 compares the average precision and mean reciprocal rank (MRR) of our method to those of three baselines, (1) where answers are ranked randomly (**Baseline (Random)**), (2) by length (**Baseline (Length)**), and (3) by Jensen Shannon Divergence (**JSD**) with all answers. We expect that the best answer is the one that covers the most information, which is likely to have a smaller JSD. Therefore, we use JSD to rank answers in the ascending order. Table 2 manifests that our ranker outperforms all the other methods.

	Baseline (Random)	**Baseline (Length)**	**JSD**	**Ranker (ListNet)**
Avg Precision	0.1305	0.2834	0.4000	**0.5336**
MRR	0.3403	0.4889	0.5909	**0.6496**

Table 2: Performance for best answer prediction. Our ranker outperforms the three baselines.

5.2 Community QA Summarization

Automatic Evaluation. Since human written abstracts are not available for the Yahoo! Answers dataset, we adopt the Jensen-Shannon divergence (JSD) to measure the summary quality. Intuitively, a smaller JSD implies that the summary covers more of the content in the answer set. Louis and Nenkova (2013) report that JSD has a strong negative correlation (Spearman correlation $= -0.737$) with the overall summary quality for multi-document summarization (MDS) on news articles and blogs. Our task is similar to MDS. Meanwhile, the average JSD of the best answers in our test set is smaller than that of the other answers (0.39 vs. 0.49), with an average length of 103 words compared with 67 words for the other answers. Also, on the blog task (Section 5.3), the top two systems by JSD also have the top two ROUGE scores (a common metric for summarization evaluation when human-constructed summaries are available). Thus, we conjecture that JSD is a good metric for community QA summaries.

Table 3 (left) shows that our system using a content coverage function based on Cosine using TFIDF weights, and a dispersion function (h_{sum}) based on lexicon dissimilarity and 100 topics, outperforms all of the compared approaches (paired-t test, $p < 0.05$). The topic number is tuned on the development set, and we find that varying the number of topics does not impact performance too much. Meanwhile, both our system and Dasgupta et al. (2013) produce better JSD scores than the two variants of the Lin and Bilmes (2011) system, which implies the effectiveness of the dispersion function. We further examine the effectiveness of each component that contributes to the objective function (Section 3.4), and the results are shown in Table 3 (right).

	Length	
	100	200
Best answer	0.3858	-
Lin and Bilmes (2011)	0.3398	0.2008
Lin and Bilmes (2011) + q	0.3379	0.1988
Dasgupta et al. (2013)	0.3316	0.1939
Our system	**0.3017**	**0.1758**

	JSD$_{100}$	**JSD$_{200}$**
Rel(evance)	0.3424	0.2053
Rel + Aut(hor)	0.3375	0.2040
Rel + Aut + TM (Topic Models)	0.3366	0.2033
Rel + Aut + TM + Pol(arity)	0.3309	0.1983
Rel + Aut + TM + Pol + Cont(ent Coverage)	0.3102	0.1851
Rel + Aut + TM + Pol + Cont + Disp(ersion)	**0.3017**	**0.1758**

Table 3: [**Left**] Summaries evaluated by Jensen-Shannon divergence (JSD) on Yahoo Answer for summaries of 100 words and 200 words. The average length of the best answer is 102.70. [**Right**] Value addition of each component in the objective function. The JSD on each line is statistically significantly lower than the JSD on the previous ($\alpha = 0.05$).

Human Evaluation. Human evaluation for Yahoo! Answers is carried out on Amazon Mechanical Turk[6] with carefully designed tasks (or "HITs"). Turkers are presented summaries from different systems in a random order, and asked to provide two rankings, one for overall quality and the other for information diversity. We indicate that informativeness and non-redundancy are desirable for quality; however, Turkers are allowed to consider other desiderata, such as coherence or responsiveness, and write down those when they submit the answers. Here we believe that ranking the summaries is easier than evaluating each summary in isolation (Lerman et al., 2009).

[6] https://www.mturk.com/mturk/

We randomly select 100 questions from our test set, each of which is evaluated by 4 distinct Turkers located in United States. 40 HITs are thus created, each containing 10 different questions. Four system summaries (best answer, Dasgupta et al. (2013), and our system with 100 and 200 words respectively) are displayed along with one noisy summary (i.e. irrelevant to the question) per question in random order.[7] We reject Turkers' HITs if they rank the noisy summary higher than any other. Two duplicate questions are added to test intra-annotator agreement. We reject HITs if Turkers produced inconsistent rankings for both duplicate questions. A total of 137 submissions of which 40 HITs pass the above quality filters.

Turkers of all accepted submissions report themselves as native English speakers. An inter-rater agreement of Fleiss' κ of 0.28 (fair agreement (Landis and Koch, 1977)) is computed for quality ranking and κ is 0.43 (moderate agreement) for diversity ranking. Table 4 shows the percentage of times a particular method is picked as the best summary, and the macro-/micro-average rank of a method, for both overall quality and information diversity. Macro-average is computed by first averaging the ranks per question and then averaging across all questions.

For overall quality, our system with a 200 word limit is selected as the best in 44.6% of the evaluations. It outperforms the best answer (31.9%) significantly, which suggests that our system summary covers relevant information that is not contained in the best answer. Our system with a length constraint of 100 words is chosen as the best for quality 12.5% times while that of Dasgupta et al. (2013) is chosen 11.0% of the time. Our system is also voted as the best summary for diversity in 78.7% of the evaluations. More interestingly, both of our systems, with 100 words and 200 words, outperform the best answer and Dasgupta et al. (2013) for average ranking (both overall quality and information diversity) significantly by using Wilcoxon signed-rank test ($p < 0.05$). When we check the reasons given by Turkers, we found that people usually prefer our summaries due to "helpful suggestions that covered many options" or being "balanced with different opinions". When Turks prefer the best answers, they mostly stress on coherence and responsiveness. Sample summaries from all the systems are displayed in Figure 2.

	Length of Summary	**Overall Quality**			**Information Diversity**		
		% Best	Average Rank Macro	Micro	% Best	Average Rank Macro	Micro
Best answer	102.70	31.9%	2.68	2.69	9.6%	3.27	3.29
Dasgupta et al. (2013)	100	11.0%	2.84	2.83	5.0%	2.95	2.94
Our system		12.5%	**2.50***	**2.50***	6.7%	**2.43***	**2.43***
Our system	200	**44.6%**	**1.98***	**1.98***	**78.7%**	**1.35***	**1.34***

Table 4: Human evaluation on Yahoo! Answer Data. **Boldface** implies statistically significance compared to other results in the same columns using paired-t test. Both of our systems are ranked higher (i.e. numbers in **bold** with *) than the best answers voted by Yahoo! users and system summaries from Dasgupta et al. (2013).

Question: What is the long term effect of piracy on the music and film industry?

Dasgupta et al. (2013) (Qty Rank=2.75 Div. Rank=2.5):
- In short, I don't think the music industry in particular will ever enjoy the huge profits of the 90's.
- Please-People in those businesses make millions of dollars as it is !! I don't think piracy hurts them at all !!!
- The other thing will be music and movies with less quality.
- Its a big gray area, I dont see anything wrong with burning a mix cd or a cd for a friend so long as youre not selling them for profit.
- By removing the profitability of music & film companies, piracy takes away their motivation to produce new music & movies.

Our system (100 words) (Qty Rank=2.25 Div. Rank=2.25):
- Rising costs for movies and music. The other thing will be music and movies with less quality.
- Now, with piracy, there isn't the willingness to take chances.
- But it's also like the person put the effort into it and they aren't getting paid. It's a big gray area, I don't see anything wrong with burning a mix cd or a cd for a friend so long as you're not selling them for profit.
- It is forcing them to rework their business model, which is a good thing.

Our system (200 words) (Qty. Rank=2.25, Div Rank=1.25):
- Rising costs for movies and music. The other thing will be music and movies with less quality.
- Now, with piracy, there isn't the willingness to take chances. American Idol is the result of this. .... The real problem here is that the mainstream music will become even tighter. Record labels will not won't to go far from what is currently like by the majority.
- I hate when people who have billions of dollars whine about not having more money. But it's also like the person put the effort into it and they aren't getting paid ... I don't see anything wrong with burning a mix cd or a cd for a friend ....
- It is forcing them to rework their business model, which is a good thing.
- By removing the profitability of music & film companies, piracy takes away their motivation to produce new music & movies.

Figure 2: Sample summaries from Dasgupta et al. (2013), and our systems (100 words and 200 words). Sentences from separate bullets (•) are partial answers from different users.

[7]Note that we aim to compare results with the gold-standard best answers of about 100 words. The evaluation of the 200-word summaries is provided only as an additional data-point.

5.3 Blog Summarization

Automatic Evaluation. We use the ROUGE (Lin and Hovy, 2003) software with standard options to automatically evaluate summaries with reference to the human labeled nuggets as those are available for this task. ROUGE-2 measures bigram overlap and ROUGE-SU4 measures the overlap of unigram and skip-bigram separated by up to four words. We use the ranker trained on Yahoo! data to produce relevance ordering, and adopt the system parameters from Section 5.2. Table 5 (left) shows that our system outperforms the best system in TAC'08 with highest ROUGE-2 score (Kim et al., 2008), the two baselines (TFIDF+Lexicon, and our ranker), Lin and Bilmes (2011), and Dasgupta et al. (2013).

	ROUGE-2	**ROUGE-SU4**	**JSD**
Best system in TAC'08	0.2923	0.3766	0.3286
TFIDF + Lexicon	0.3069	0.3876	0.2429
Ranker (ListNet)	0.3200	0.3960	0.2293
Lin and Bilmes (2011)	0.2732	0.3582	0.2330
Lin and Bilmes (2011) + q	0.2852	0.3700	0.2349
Dasgupta et al. (2013)	0.2618	0.3500	0.2370
Our system	**0.3234**	**0.3978**	*0.2258*

	Pyramid F-score
Best system in TAC'08	0.2225
Lin and Bilmes (2011)	0.2790
Our system	**0.3620**

Table 5: Results on TAC'08 dataset. [**Left**] Our system has significant better ROUGE scores than all the other systems except our ranker (paired-t test, $p < 0.05$). We also achieve the best JS divergence. [**Right**] Human evaluation with Pyramid F-score. Our system significantly outperforms the others.

Human Evaluation. For human evaluation, we use the standard Pyramid F-score used in the TAC'08 opinion summarization track with $\beta = 3$ (Dang, 2008). In the TAC task, systems are allowed to return up to 7,000 non-white characters for each question. Since the TAC metric favors recall we do not produce summaries shorter than 7,000 characters. We ask two human judges to evaluate our system along with the one that got the highest Pyramid F-score in the TAC'08 and Lin and Bilmes (2011). Cohen's κ for inter-annotator agreement is 0.68 (substantial). While we did not explicitly evaluate non-redundancy, both of our judges report that our system summaries contain less redundant information.

5.4 Further Discussion

Yahoo! Answer				
	DISPERSION$_{sum}$		DISPERSION$_{min}$	
DISSIMI	Cont$_{tfidf}$	Cont$_{sem}$	Cont$_{tfidf}$	Cont$_{sem}$
Semantic	0.3143	0.324 3	0.3129	0.3232
Topical	0.3101	0.3202	0.3106	0.3209
Lexical	**0.3017**	0.3147	0.3071	0.3172

TAC 2008				
	DISPERSION$_{sum}$		DISPERSION$_{min}$	
DISSIMI	Cont$_{tfidf}$	Cont$_{sem}$	Cont$_{tfidf}$	Cont$_{sem}$
Semantic	0.2216	0.2169	0.2772	0.2579
Topical	0.2128	0.2090	**0.3234**	0.3056
Lexical	0.2167	0.2129	0.3117	0.3160

Table 6: Effect of different dispersion functions, content coverage, and dissimilarity metrics on our system. [**Left**] JSD values for different combinations on Yahoo! data, using LDA with 100 topics. All systems are significantly different from each other at significance level $\alpha = 0.05$. Systems using summation of distances for dispersion function (h_{sum}) uniformly outperform the ones using minimum distance (h_{min}). [**Right**] ROUGE scores of different choices for TAC 2008 data. All systems use LDA with 40 topics. The parameters of our systems are adopted from the ones tuned on Yahoo! Answers.

Given that the text similarity metrics and dispersion functions play important roles in the framework, we further study the effectiveness of different content coverage functions (Cosine using TFIDF vs. Semantic), dispersion functions (h_{sum} vs. h_{min}), and dissimilarity metrics used in dispersion functions (Semantic vs. Topical vs. Lexical). Results on Yahoo! Answer (Table 6 (left)) show that systems using summation of distances for dispersion functions (h_{sum}) uniformly outperform the ones using minimum distance (h_{min}). Meanwhile, Cosine using TFIDF is better at measuring content coverage than WordNet-based semantic measurement, and this may due to the limited coverage of WordNet on verbs. This is also true for dissimilarity metrics. Results on blog data (Table 6 (right)), however, show that using minimum distance for dispersion produces better results. This indicates that optimal dispersion function varies by genre. Topical-based dissimilarity also marginally outperforms the other two metrics in blog data.

6 Conclusion

We propose a submodular function-based opinion summarization framework. Tested on community QA and blog summarization, our approach outperforms state-of-the-art methods that are also based on submodularity in both automatic evaluation and human evaluation. Our framework is capable of including statistically learned sentence relevance and encouraging the summary to cover diverse topics. We also study different metrics on text similarity estimation and their effect on summarization.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada, 7-8 June. Association for Computational Linguistics.

Hadi Amiri, Zheng-Jun Zha, and Tat-Seng Chua. 2013. A pattern matching based model for implicit opinion question identification. In *AAAI*. AAAI Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 129–136, New York, NY, USA. ACM.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 335–336, New York, NY, USA. ACM.

Asli Celikyilmaz, Dilek Hakkani-Tur, and Gokhan Tur. 2010. Lda based similarity modeling for question answering. In *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search*, SS '10, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.

Barun Chandra and Magnús M. Halldórsson. 1996. Facility dispersion and remote subgraphs. In *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory*, SWAT '96, pages 53–65, London, UK, UK. Springer-Verlag.

Hoa Tran Dang. 2008. Overview of the tac 2008 opinion question answering and summarization tasks. In *Proc. TAC 2008*.

Van Dang. 2011. RankLib. http://www.cs.umass.edu/˜vdang/ranklib.html.

Anirban Dasgupta, Ravi Kumar, and Sujith Ravi. 2013. Summarization through submodularity and dispersion. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1014–1022, Sofia, Bulgaria, August. Association for Computational Linguistics.

Hal Daumé, III and Daniel Marcu. 2006. Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 305–312, Stroudsburg, PA, USA. Association for Computational Linguistics.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06*, pages 417–422.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

George Karypis. 2003. CLUTO - a clustering toolkit. Technical Report #02-017, November.

Hyun Duk Kim, Dae Hoon Park, V.G.Vinod Vydiswaran, and ChengXiang Zhai. 2008. Opinion summarization using entity features and probabilistic sentence coherence optimization: Uiuc at tac 2008 opinion summarization pilot. In *Proc. TAC 2008*.

J R Landis and G G Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Kevin Lerman, Sasha Blair-Goldensohn, and Ryan McDonald. 2009. Sentiment summarization: Evaluating and learning user preferences. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 514–522, Stroudsburg, PA, USA. Association for Computational Linguistics.

Baoli Li, Yandong Liu, and Eugene Agichtein. 2008a. Cocqa: Co-training over questions and answers with an application to predicting question subjectivity orientation. In *EMNLP*, pages 937–946.

Wenjie Li, You Ouyang, Yi Hu, and Furu Wei. 2008b. Polyu at tac 2008. In *Proc. TAC 2008*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 510–520, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. COLING '00, pages 495–501, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 71–78.

Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 497–504, Stroudsburg, PA, USA. Association for Computational Linguistics.

Annie Louis and Ani Nenkova. 2013. Automatically assessing machine summary content without a gold standard. *Comput. Linguist.*, 39(2):267–300, June.

Xiaoqiang Luo, Hema Raghavan, Vittorio Castelli, Sameer Maskey, and Radu Florian. 2013. Finding what matters in questions. In *HLT-NAACL*, pages 878–887.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. ECIR'07, pages 557–564, Berlin, Heidelberg. Springer-Verlag.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, pages 775–780. AAAI Press.

G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. 1978. An analysis of approximations for maximizing submodular set functionsI. *Mathematical Programming*, 14(1):265–294, December.

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005-101*.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January.

Michael J. Paul, ChengXiang Zhai, and Roxana Girju. 2010. Summarizing contrastive viewpoints in opinionated text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 66–76, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ruben Sipos, Pannaga Shivaswamy, and Thorsten Joachims. 2012. Large-margin learning of submodular summarization models. EACL '12, pages 224–233, Stroudsburg, PA, USA. Association for Computational Linguistics.

Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, Cambridge, MA.

Veselin Stoyanov and Claire Cardie. 2006. Partially supervised coreference resolution for opinion summarization through structured rule learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 336–344, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mattia Tomasoni and Minlie Huang. 2010. Metadata-aware measures for answer summarization in community question answering. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 760–769, Stroudsburg, PA, USA. Association for Computational Linguistics.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pengtao Xie and Eric Xing. 2013. Integrating document clustering and topic modeling. In *Proceedings of the Twenty-Ninth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-13)*, pages 694–703, Corvallis, Oregon. AUAI Press.

Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Generating Supplementary Travel Guides from Social Media

Liu Yang[1,2], **Jing Jiang**[2,*], **Lifu Huang**[1,2], **Minghui Qiu**[2], **Lizi Liao**[2,3]
[1]Peking University / Beijing, China, 100871
[2]Singapore Management University / Singapore, Singapore, 178902
[3]Beijing Institute of Technology / Beijing, China, 100081
yang.liu@pku.edu.cn, jingjiang@smu.edu.sg
{warrior.fu, minghuiqiu, liaolizi.llz}@gmail.com

Abstract

In this paper we study how to summarize travel-related information in forum threads to generate supplementary travel guides. Such summaries presumably can provide additional and more up-to-date information to tourists. Existing multi-document summarization methods have limitations for this task because (1) they do not generate structured summaries but travel guides usually follow a certain template, and (2) they do not put emphasis on named entities but travel guides often recommend points of interest to travelers. To overcome these limitations, we propose to use a latent variable model to align forum threads with the section structure of well-written travel guides. The model also assigns section labels to named entities in forum threads. We then propose to modify an ILP-based summarization method to generate section-specific summaries. Evaluation on threads from Yahoo! Answers shows that our proposed method is able to generate better summaries compared with a number of baselines based on ROUGE scores and coverage of named entities.

1 Introduction

Online forums and community question answering (CQA) sites contain much useful information from ordinary users, such as their personal experience, opinions, suggestions and recommendations. Extracting and summarizing information from these rich information sources has a wide range of applications. In this work, we study how to tap into user-generated content in forums such as Yahoo! Answers to generate supplementary city travel guides. Travel guides published by well-known publishers such as Lonely Planet are written by a small number of authors based on their travel experience. Presumably if we could summarize the large amount of information given by ordinary users about a city, such a summary could supplement the official travel guide and cover more up-to-date information.

However, social media content is diverse and noisy because it is contributed by many different authors. Directly applying existing multi-document summarization methods to forum and CQA threads may not produce good travel guides for the following reasons: (1) Summaries produced by standard summarization methods are not structured, but travel guides usually follow a template structure. (2) Travel guides put much emphasis on points of interest, which are usually location entities, but standard text summarization methods are not entity-oriented.

To illustrate our points, in Table 1 we show (i) the overall structure of a travel guide for Sydney from Lonely Planet, (ii) an excerpt from a summary generated by a state-of-the-art ILP-based summarization method (Gillick and Favre, 2009) from a set of threads related to Sydney, and (iii) excerpts of a structured summary generated by our proposed method. The comparison shows that the summary generated by the standard ILP method mixes information on different topics together and does not mention many

* Corresponding author.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1670–1681, Dublin, Ireland, August 23-29 2014.

Travel Guide from Lonely Planet (http://www.lonelyplanet.com/australia/sydney/)
Restaurants: **Sepia**:There's nothing washed out or brown-tinged about Sepia's food: Martin Benn's picture-perfect creations are presented in ... **Icebergs Dining Room**: Poised above the famous Icebergs swimming pool, Icebergs views sweep across the Bondi Beach arc to ... *Shopping*: **Strand Arcade**: Constructed in 1891, the Strand rivals the QVB in the ornateness stakes. Three floors of designer fashions ... **Westfield Sydney**: The city's newest shopping mall is a bafflingly large complex gobbling up Sydney Tower and a fair chunk of ... *Transport*: **Sydney Airport**: Sydneys Kingsford Smith Airport , 10km south of the city centre, is Australias busiest airport, handling flights ... **Water Taxis Combined**:Fares based on up to four passengers; add $10 per person for additional passengers. Sample fares ...

Yahoo! Answers Summary Generated by Standard ILP Method	Yahoo! Answers Summary Generated by Our Method
It 's not too far from **Sydney** . **Sydney** is the most expensive place in **Australia** . They are a little lame ... Then you can go to **Darling Harbour**, a beautiful habour which is a 10-minute walk from town hall station . Make sure , if you are up to it to do the bridge climb , this is a real treat . There are lots of interesting things to see and do in and around **Sydney** . The suburbs-much cheaper than the **CBD**. It was in the basement of a big shopping mall . The only way to do that is to drive . Got to walk on top of the **Sydney** harbour bridge and go up centre point tower ! Walk around the street and see the beach . I would like to stay at a nice hotel . My friend and I are wanting to take a trip to **Sydney** for the summer . But you 'll need to get there by taxi . **Sydney** is so pretty, so you should be able to find stuff to do . And they have many facilities . Good luck and have fun . Public transport is not very good . Depending on what you 're in **Sydney** to do it 's hard to say ...	*Restaurants*: Go to the two major restaurant areas close to the city **Darlinghurst** , along **Oxford Street** , and **Newtown** , along **King Street** . **Chinatown** which is off **George St.** in the city look up **Dixon st.** is a great place to get a cheap **Chinese** meal ... *Shopping*: **Queen Victoria Building** and **Pitt St Mall** , **World Square** and the **Strand** are good ideas to check out . Hair driers you can get in many places , but the main places would be the department stores such as **Target** , **Big W** , **K-Mart** , **Myer**, **David Jones** ... *Transport*: The **CBD** is about 15 minutes by train from the airport and there is a station at **Circular Quay** , right on the **Harbour** with access to the bridge and the **Opera House** . You can catch an intercity train with **Cityrail** from just about anywhere in **Sydney** ...

Table 1: Comparison of different travel guides about Sydney. Top: excerpts from Lonely Planet. Bottom left: excerpt from a summary generated by standard ILP. Bottom right: excerpts from summary generated by our method. Named entities are highlighted in bold font.

interesting places to visit. The summary by our proposed method, in contrast, organizes the information into sections and has a high coverage of places a tourist can visit.

To generate the kind of summaries as shown in the bottom right of Table 1, we propose to first leverage the section structure of well-written travel guides and use a latent variable model to align forum threads with the different sections from these travel guides. Moreover, observing that points of interest are organized by sections in these travel guides, we also identify location names from user-generated content and try to uncover their underlying section labels. We then treat the remaining problem as a multi-document summarization task. We modify an Integer Linear Programming (ILP)-based extractive summarization framework (Gillick and Favre, 2009) to select sentences from forum threads to generate section-specific summaries, where we specifically emphasize the inclusion of potential points of interest for each section. Experiments using threads from Yahoo! Answers show that our proposed method generates better summaries than a number of baselines in terms of ROUGE scores and coverage of named entities.

Our work makes the following contributions. First, we study a new problem of summarizing multiple forum threads to generate city travel guides based on known template structure from well-written travel guides. Second, we propose a principled approach based on latent variable models and Integer Linear Programming. Third, we evaluate our method using real forum threads and human generated model summaries, and the results are positive.

2 Overview of Our Method

Our task is to summarize travel-related information from forum threads for potential tourists. In order to inject some structure into the generated summaries, we assume that we have a set of I well-written travel guides that correspond to I different cities and have the same structure. We refer to these travel guides as official travel guides. Each official travel guide consists of a fixed set of S sections such as *restaurants* and *shopping*, and this section structure will be used to organize our generated summaries. We further assume that each section of an official travel guide consists of a list of points of interest, each with a name and a short description, as illustrated in Figure 1. We believe that this is a fairly common structure followed by many if not all travel guides.

Given a target city, we assume that we can collect a set of threads about this city from travel-related forums. In this paper we use threads from Yahoo! Answers, but our solution does not use any CQA properties of the threads, so threads from other general forums can also be used. Our goal is to generate a text summary with S sections from these threads, where each section has a length limit.

As we have mentioned, we treat the problem as a multi-document summarization task. However, different from standard text summarization, our generated summaries should contain S sections. To achieve this goal, we first select a set of relevant threads for each section and then perform section-specific summarization from the selected threads.

Thread selection: To select relevant threads given a section, a naive solution is to rank the threads based on their relevance to the section, where relevance can be measured by, for example, cosine similarity between a thread and all the text in the given travel guides belonging to the section. But we observe that the language used in forum threads could be very different from that in the official travel guides, making it hard to measure relevance purely based on lexical overlap. For example, in the *entertainment* section, forum threads may contain words such as "djs," "Xmas," "b'day" and "anni.", but these words do not occur in the official travel guides. To overcome this difficulty, we propose to use a latent variable model that jointly models official travel guides and forum threads. We treat the S sections as S latent factors that govern the generation of the forum threads. With the latent factors observed in the official travel guides, we receive some supervision; and yet by jointly modeling both the official travel guides and the forum threads, we allow the latent factors to adapt to the lexical variations in user-generated content. In the end, the learned latent factors can help us align forum threads with the sections and subsequently select the most relevant ones for each section.

Section-specific summarization: Given the selected relevant threads for a section, we adopt an ILP-based extractive summarization framework that has been shown to be effective (Gillick and Favre, 2009). We modify the objective function in this framework to consider two factors: (1) Since not every sentence in the selected threads is highly relevant to the section, we want to give preference to those more relevant sentences in the objective function, where relevance can be measured using word distributions learned by the latent variable model. (2) Since travel guides are expected to recommend points of interest to readers, we try to maximize the coverage of section-specific location entities in the objective function.

3 Joint City Section Model

3.1 Model

In this section we present our Joint City Section Model (JCSM), which links official travel guides and forum threads. The model is a typical extension of LDA, where a number of latent topics (i.e. latent factors) are assumed to have generated the observed text. First of all, for each pre-defined section there is a latent topic. These explain words such as "food" and "menu" for *restaurants* and "store" and "mall" for *shopping*. In addition, in both travel guides and forum threads, some words are more related to the city being discussed than any specific section. For example, when New York City is being discussed, words such as "NYC" and "Manhattan" may frequently show up in any section. We therefore further assume that for each city there is a city-specific topic. A switch variable is used to determine whether a word comes from a city-specific or section-specific topic.

A special design of our model that differs from many existing LDA extensions is the treatment of named entities. We first use a named entity recognizer to identify potential names of locations from forum threads. We assume that each of these entities belongs to a section, which is indicated by a latent variable. We then assume that the section labels of the non-entity words in forum threads are dependent on the section labels of these entities. By doing so, we emphasize the importance of associating potential points of interest with sections, which will be useful when we generate summaries.

We now formally present JCSM. To simplify the model description, we assume that we work with I cities, each of which has a given, well-written travel guide and a set of forum threads. Note that in practice this model can be easily extended such that a target city with forum threads does not need to have a given travel guide to begin with. Let ϕ_i denote the word distribution for the city-specific latent topic associated with city i. Let ψ_s denote the word distribution for the section-specific latent topic for section s. Let $d_{i,s,n}$ denote the n-th word in the s-th section of the i-th city's travel guide. Here $1 \leq d_{i,s,n} \leq V$ is an index into the vocabulary with size V. Let $x_{i,s,n}$ be a switch variable associated with $d_{i,s,n}$ to indicate whether this word is city-specific or section-specific. For the j-th forum thread related to the i-th city, we assume there is a distribution over sections, denoted as $\theta_{i,j}$. For the l-th location entity in the k-th post

of this thread, we assume a latent variable $c_{i,j,k,l}$ ($1 \leq c_{i,j,k,l} \leq S$) that indicates the section label of this entity. Then for the m-th word in this post, we first use a switch variable $y_{i,j,k,m}$ to determine whether the word is city-specific or section-specific. If it is section-specific, we then choose one of the entities in the same post, denoted as $z_{i,j,k,m}$, and its corresponding section label as the section for this word.

All the binary switch variables follow a global Bernoulli distribution parameterized by π. There are hyperparameters α, β, β' and γ that define the prior distributions. The complete model is depicted in Figure 1. The generative process of JCSM is also described as follows.

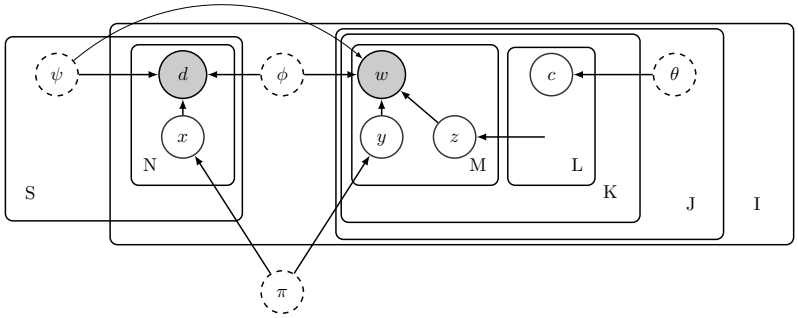


Figure 1: The plate notation of the Joint City Section Model (JCSM). Dashed variables will be integrated out in Gibbs sampling. For clarity, the Dirichlet and Beta priors are omitted. The arrow pointing to z indicates that z is drawn from a uniform distribution over the integers from 1 to L.

- For each city i, ($i = 1, 2, \cdots, I$), draw a city-specific word distribution $\phi_i \sim \text{Dir}(\beta')$
- For each section s, ($s = 1, 2, \cdots, S$), draw a section-specific word distribution $\psi_s \sim \text{Dir}(\beta)$
- Draw a switch distribution $\pi \sim \text{Beta}(\gamma)$
- For each city i ($i = 1, 2, \cdots, I$)
  - For each section s ($s = 1, 2, \cdots, S$)
    - For the n-th word in the given travel guide
      - Draw $x_{i,s,n} \sim \text{Bernoulli}(\pi)$
      - If $x_{i,s,n} = 1$, draw $d_{i,s,n} \sim \text{Multi}(\psi_s)$; otherwise, draw $d_{i,s,n} \sim \text{Multi}(\phi_i)$.
  - For the j-th thread
    - Draw a thread specific section distribution $\theta_j \sim \text{Dir}(\alpha)$
    - For the k-th post
      - For the l-th entity, draw $c_{i,j,k,l} \sim \text{Multi}(\theta_j)$
      - For the m-th word, draw $y_{i,j,k,m} \sim \text{Bernoulli}(\pi)$. If $y_{i,j,k,m} = 1$, draw $z_{i,j,k,m} \sim \text{Uniform}(1, \cdots, L_{i,j,k})$ and then draw $w_{i,j,k,m} \sim \text{Multi}(\psi_{c_{i,j,k,z_{i,j,k,m}}})$; otherwise, draw $w_{i,j,k,m} \sim \text{Multi}(\phi_i)$.

3.2 Inference

We use collapsed Gibbs sampling to estimate the parameters in the model. The problem is to compute the Gibbs update rules for sampling $x_{i,s,n}$, $c_{i,j,k,l}$, $z_{i,j,k,m}$, $y_{i,j,k,m}$.

Sample entity topic $c_{i,j,k,l}$

Let b denote $\{i, j, k, l\}$ and u denote $\{i, j, k\}$. We can derive the Gibbs update rule for sampling entity topic $c_{i,j,k,l}$ as follows:

$$p(c_b = s|\mathbf{C}_{\neg_b}, \mathbf{W}, \mathbf{D}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}) = \frac{n^s_{i,j,\neg_b} + \alpha}{\sum_{s'=1}^{S} n^{s'}_{i,j,\neg_b} + S\alpha} \cdot \frac{\prod_{w=1}^{V} \prod_{i'=1}^{n^w_{u,y=1,z=l}} (n^w_{y=1,z=l,\neg_u} + \beta + i' - 1)}{\prod_{j'=1}^{n^{\mathbf{w}}_{y=1,z=l,u}} (\sum_{w=1}^{V} n^w_{y=1,z=l,\neg_u} + V\beta + j' - 1)},$$

where $n^s_{i,j,\neg_b}$ denotes the number of entities whose topic assignments are s in thread $\{i, j\}$ without consideration of entity $\{i, j, k, l\}$. $n^w_{u,y=1,z=l}$ denotes the number of times term w occurs in the post $\{i, j, k\}$ with the constraint that $y = 1$ and $z = l$. $n^w_{y=1,z=l,\neg_u}$ is the number of times term w occurs in all posts except the post $\{i, j, k\}$ with the constraint that $y = 1$ and $z = l$.

Sample switch label $x_{i,s,n}$

We can derive the Gibbs update rule for sampling $x_{i,s,n}$ in a similar way. Note that the sampling of $x_{i,s,n}$ is in travel guide word level. Let g denote$\{i, s, n\}$, the Gibbs update rule for sampling $x_{i,s,n}$ is as follows:

$$p(x_g = 0|\mathbf{C}, \mathbf{W}, \mathbf{D}_{\neg_g}, \mathbf{X}_{\neg_g}, \mathbf{Y}, \mathbf{Z}) = \frac{n^{x=0}_{\neg_g} + \gamma}{\sum_{x=0}^{1} n^{x}_{\neg_g} + 2\gamma} \cdot \frac{n^{w_g}_{x=0,i,\neg_g} + \beta'}{\sum_{w=1}^{V} n^{w}_{x=0,i,\neg_g} + V\beta'}$$

$$p(x_g = 1|\mathbf{C}, \mathbf{W}, \mathbf{D}_{\neg_g}, \mathbf{X}_{\neg_g}, \mathbf{Y}, \mathbf{Z}) = \frac{n^{x=1}_{\neg_g} + \gamma}{\sum_{x=0}^{1} n^{x}_{\neg_g} + 2\gamma} \cdot \frac{n^{w_g}_{x=1,s,\neg_g} + \beta}{\sum_{w=1}^{V} n^{w}_{x=1,s,\neg_g} + V\beta}$$

Sample post word topic $z_{i,j,k,m}$ and switch label $y_{i,j,k,m}$

For words in the thread posts, We can derive the Gibbs update rule for sampling post word topic $z_{i,j,k,m}$ and switch label $y_{i,j,k,m}$. Note that the sampling of $z_{i,j,k,m}$ and $y_{i,j,k,m}$ is in post word level. Let f denote$\{i, j, k, m\}$. The Gibbs update rule for sampling $z_{i,j,k,m}$ and $y_{i,j,k,m}$ is as follows:

$$p(z_f = s|\mathbf{C}, \mathbf{W}_{\neg_f}, \mathbf{D}, \mathbf{X}, \mathbf{Y}_{\neg_f}, \mathbf{Z}_{\neg_f}) = \frac{n^{w_f}_{y=1,s',\neg_f} + \beta}{\sum_{w=1}^{V} n^{w}_{y=1,s',\neg_f} + V\beta} \cdot \frac{1}{L_{i,j,k}}$$

$$p(y_f = 0|\mathbf{C}, \mathbf{W}_{\neg_f}, \mathbf{D}, \mathbf{X}, \mathbf{Y}_{\neg_f}, \mathbf{Z}_{\neg_f}) = \frac{n^{y=0}_{\neg_f} + \gamma}{\sum_{y=0}^{1} n^{y}_{\neg_f} + 2\gamma} \cdot \frac{n^{w_f}_{y=0,i,\neg_f} + \beta'}{\sum_{w=1}^{V} n^{w}_{y=0,i,\neg_f} + V\beta'}$$

$$p(y_f = 1|\mathbf{C}, \mathbf{W}_{\neg_f}, \mathbf{D}, \mathbf{X}, \mathbf{Y}_{\neg_f}, \mathbf{Z}_{\neg_f}) = \frac{n^{y=1}_{\neg_f} + \gamma}{\sum_{y=0}^{1} n^{y}_{\neg_f} + 2\gamma} \cdot \frac{n^{w_f}_{y=1,s',\neg_f} + \beta}{\sum_{w=1}^{V} n^{w}_{y=1,s',\neg_f} + V\beta}$$

where $s' = c_{i,j,k,l}$ which is the topic index of the associated entity of this word.

Parameter estimation

After Gibbs Sampling, we can make the following parameter estimation:

$$\theta_{i,j,s} = \frac{n^{s}_{i,j} + \alpha}{\sum_{s'=1}^{S} n^{s'}_{i,j} + S\alpha}. \quad \text{thread-section distribution.}$$

$$\psi_{s,w} = \frac{n^{w}_{s,y=1} + \beta}{\sum_{w'=1}^{V} n^{w'}_{s,y=1} + V\beta}. \quad \text{section-word distribution.}$$

$$\phi_{i,w} = \frac{n^{w}_{i,y=0} + \beta'}{\sum_{w'=1}^{V} n^{w'}_{i,y=0} + V\beta'}. \quad \text{city-word distribution.}$$

$$\pi_{y} = \frac{n^{y}_{(.)} + \gamma}{\sum_{y'=0}^{1} n^{y'}_{(.)} + 2\gamma}. \quad \text{switch distribution.}$$

4 Generating Section-specific Summaries

With the JCSM model presented in the last section, we can learn a word distribution for each section, which can help us find more relevant content for the section. For each section, we rank the forum threads by how likely the words inside a thread is generated from the corresponding section-specific word distribution. We select the top-K threads for each section to perform section-specific summarization.

Extractive summarization has been well studied and many algorithms have been proposed. We choose to build our solution on top of an ILP-based framework proposed by Gillick and Favre (2009), partly because our experiments comparing this ILP framework and other existing methods show its advantage on our data sets (see Section 5). Below we first briefly review this ILP-based summarization framework and then present our proposed improvements.

The idea behind the ILP framework by Gillick and Favre (2009) is to maximize the coverage of so-called "concepts" from the original corpus in the generated summary. In practice, bigrams are used as concepts. Specifically, let us use i to index all the concepts from the original corpus. Let w_i denote the weight of the i-th concept computed based on its frequency and $b_i \in \{0, 1\}$ denote the absence or

presence of the concept. The framework aims to maximize $\sum_i w_i b_i$, i.e. the total weighted coverage of the concepts, subject to the following constraints:

$$\begin{array}{rcl} & \sum_j l_j s_j \leq L, & (l_j \text{ is the length of the } j\text{-th sentence in terms of words, and } L \text{ is the length limit of the summary.}) \\ \forall i,j: & s_j o_{i,j} \leq b_i, & (s_j \in \{0,1\} \text{ denotes the absence or presence of the } j\text{-th sentence.}) \\ \forall i: & \sum_j s_j o_{i,j} \geq b_i. & (o_{i,j} \in \{0,1\} \text{ denotes whether concept } i \text{ occurs in sentence } j.) \end{array}$$

Although this framework works well for standard summarization, our task is different. We propose the following changes to this framework:

Favoring relevant sentences: Recall that although we select presumably the most relevant threads for each section, we cannot guarantee that each sentence in these threads is related to the section. For example, we observe that the *things-to-do* section is often mixed with content from *restaurants*, *sights*, *transport* and *entertainment* sections. Also, some sentences are less relevant to the target city than others. In order to select the more relevant sentences in the summary, we propose to add the second term in Eqn. 1 below. Here j is used to index all the candidate sentences and u_j is a weight for sentence j based on its relevance.

We measure relevance with respect to both the city and the section. Let $\text{LL}(j, \psi)$ denote the log likelihood of generating sentence j from the section-specific topic ψ and $\text{LL}(j, \phi)$ denote the log likelihood of generating sentence j from the city-specific topic ϕ. We define u_j as follows:

$$u_j \quad \propto \quad \exp\left(\rho \text{LL}(j,\psi) + (1-\rho)\text{LL}(j,\phi)\right).$$

u_j are then normalized to be between 0 and 1. Note that here ρ is a manually defined parameter used to control the tradeoff between city-specific relevance and section-specific relevance. As we will show in Section 5, both relevance factors turn out to be useful.

Covering section-specific points of interest: We hypothesize that a good summary travel guide should mention potential points of interest to the reader. To this end, the last term in Eqn. 1 is added. Specifically, k is an index for unique location names we find that have been labeled as belonging to section s according to the JCSM model. $e_k \in \{0,1\}$ denotes whether the k-th entity is present in the selected sentences, and v_k denotes the weight for this entity based on its frequency.

Eventually, the summarization task is formulated as the following optimization problem:

$$\begin{array}{ll} \text{Maximize:} & \lambda_1 \sum_i w_i b_i + \lambda_2 \sum_j u_j s_j + (1-\lambda_1-\lambda_2) \sum_k v_k e_k \\ \text{Subject to:} & \sum_j l_j s_j \leq L, \\ & \forall i: \sum_j s_j o_{i,j} \geq b_i, \qquad \forall i,j: s_j o_{i,j} \leq b_i, \\ & \forall j: \sum_k s_j p_{j,k} \geq e_k, \qquad \forall j,k: s_j p_{j,k} \leq e_k. \end{array} \tag{1}$$

Here $o_{i,j}$ denotes whether concept i occurs in sentence j, and $p_{j,k}$ denotes whether entity k occurs in sentence j. For the weights w_i and v_k, we normalize them using the total occurrences of bigrams/entities to ensure their values are between 0 and 1. We solve the above optimization problem using the IBM ILOG CPLEX Optimizer[1].

5 Experiments

5.1 Data and Experimental Setup

We use real data from Yahoo! Answers and Lonely Planet for evaluation. We first crawl the travel guides for 10 cities from Lonely Planet, where each travel guide has 8 sections. We then crawl the top 60000 Q&A threads ranked by number of posts related to these 10 cities (6000 for each city) from Yahoo! Answers under the "travel" category where all questions have been grouped by cities. We filter out trivial factoid questions using features used by Tomasoni and Huang (2010). We then use the Stanford

[1] http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

Method	*Singapore*			*Sydney*			*New York City*			*Los Angeles*			Overall Average		
	R-1	R-2	RSU4	R-1	R-2	RSU4	R-1	R-2	RSU4	R-1	R-2	RSU4	R-1	R-2	RSU4
Random	0.4091	0.1046	0.1576	0.4496	0.1100	0.1925	0.4442	0.1192	0.1858	0.4154	0.1130	0.1693	0.4309	0.1115	0.1771
Centroid	0.4029	0.0993	0.1484	0.4228	0.1100	0.1764	0.4235	0.1192	0.1722	0.3763	0.0787	0.1386	0.4133	0.1077	0.1640
LexRank	0.4396	0.1451	0.1891	0.4406	0.1296	0.1955	0.4304	0.1397	0.1859	0.4032	0.0992	0.1661	0.4350	0.1331	0.1894
DivRank	0.4534	0.1504	0.1888	0.4473	0.1161	0.1925	0.4391	0.1167	0.1804	0.4275	0.1180	0.1733	0.4487	0.1317	0.1888
GMDS	0.3918	0.0890	0.1415	0.4339	0.1066	0.1784	0.4064	0.0845	0.1576	0.3846	0.0809	0.1413	0.4045	0.0916	0.1553
ILP-BL	0.4635	0.1650	0.2000	0.4948	0.1731	0.2333	0.4691	0.1613	0.2073	0.4545	0.1445	**0.1981**	0.4755	0.1654	0.2136
Our Method	**0.4723**	**0.1655**	**0.2035**	**0.5078**	**0.1787**	**0.2397**	**0.4716**	**0.1713**	**0.2086**	**0.4543**	**0.1565**	0.1945	**0.4804**‡	**0.1715**‡	**0.2144**‡

Table 2: Comparison of the summarization results. ‡ means the result is better than others except ILP-BL in the same column at 5% significance level measured by Wilcoxon signed rank test. Note that only the average scores are tested for statistical significance based on the 32 summarization tasks in total.

NER tool to recognize named entities in these threads. Since we notice that sometimes entities tagged as PER are also possible points of interest, we include all entities of LOC, ORG and PER types. In order to use higher quality threads for evaluation, for each city we pick the top 600 threads that have the most overlapping points of interest with the Lonely Planet travel guides. On average, each thread contains 5.0 posts and 618.1 words. These 600×10 threads are used to train the JCSM model.

We need human generated model summaries for evaluation. Since it is too time consuming to ask human annotators to look through 600 threads and generate structured summaries, we instead opt to first retrieve the top 30 relevant threads per section per city based on the JCSM results and then ask human annotators to summarize these 30 threads to generate a section-specific summary. Our summarization method as well as the baselines are also applied to these 30 threads per section per city for fair comparison. We randomly select 4 cities for human annotation, giving us $8 \times 4 = 32$ section-specific summarization tasks. For each task, we ask four annotators to read all 30 threads and write a summary as model summaries in our experiments[2].

We use the following baseline algorithms for comparison: (1) **Random**, which randomly picks summary sentences. (2) **Centroid** (Radev et al., 2004), which selects sentences according to several features like tfidf, cluster centroid and position. (3) **LexRank** (Erkan and Radev, 2004b)., which applies a graph-based algorithm . (4) **DivRank** (Mei et al., 2010), which employs a time-variant random walk to enhance diversity. (5) **GMDS** (Wan, 2008), which incorporates the document-level information and the sentence-to-document relationship into the ranking process. (6) **ILP-BL**, which is the method proposed by Gillick and Favre (2009).

We empirically set Dirichlet hyperparameters $\alpha = 0.5, \beta = 0.01, \gamma = 0.01, \beta' = 0.1$. We run JCSM with 400 iterations of Gibbs sampling. For the weight parameters in the ILP model, we empirically set $\lambda_1 = 0.7, \lambda_2 = 0.1, \rho = 0.7$ after we conduct multiple experiments to determine the best values of them from 0.1 to 0.9.

5.2 Summarization Results

To compare the summaries generated by our method with those generated by the baselines, we first compute their ROUGE scores against the human generated model summaries. ROUGE scores have been widely used for evaluation of summarization systems (Lin and Hovy, 2003). We use the ROUGE toolkit[3], which provides multiple kinds of ROUGE metrics including ROUGE-N, ROUGE-L, ROUGE-W and ROUGE-SU4. In the experiment results we report three ROUGE F-measure scores, namely, ROUGE-1, ROUGE-2 and ROUGE-SU4. The higher the ROUGE scores, the better a summary is.

In Table 2 we show the summarization results of our method (with the optimal parameter setting) and the baseline methods. For each city, the scores we show are averaged over the 8 sections. The overall average scores on the right hand side are averaged over the 4 cities. We have the following findings from the table: (1) Compared with the other baselines, the ILP-based baseline clearly shows its advantage, justifying our our design choice of adopting an ILP-based framework as the basis of our method. (2) Our method performs slightly better than ILP-BL based on the overall scores, but the difference is not statistically significant.

[2]The summary dataset can be found at `https://sites.google.com/site/liuyang198908/code-data`.

[3]`http://www.isi.edu/licensed-sw/see/rouge/`

section	*Singapore*		*Sydney*		*New York City*		*Los Angeles*	
	ILP-BL	Our Method	ILP-BL	Our Method	ILP-BL	Our Method	ILP-BL	Our Method
restaurants	0.3750	**0.5417**	0.5714	**0.7143**	0.2500	**0.3750**	0.1053	**0.2105**
hotels	0.4091	0.4091	0.0000	**0.5000**	0.3636	**0.5000**	0.4500	**0.5500**
shopping	0.1429	**0.5357**	0.3750	0.3750	0.1905	0.1905	0.0455	**0.1818**
sights	0.5000	**0.5789**	0.3846	**0.4615**	0.3636	**0.6364**	0.1143	**0.2571**
entertainment	0.1304	**0.2174**	0.2500	**0.7500**	0.0909	**0.2273**	0.2500	**0.4167**
activities	0.4167	**0.5833**	0.2500	0.2500	0.1250	**0.5000**	0.2069	**0.2759**
transport	0.3889	**0.5556**	0.7500	0.7500	0.6000	**0.8000**	0.3158	**0.7368**
things-to-do	0.2105	**0.2632**	0.2500	**0.5500**	0.4583	**0.5833**	0.0000	**0.2000**
average	0.3217	**0.4606**	0.3539	**0.5439**	0.3052	**0.4766**	0.1860	**0.3536**

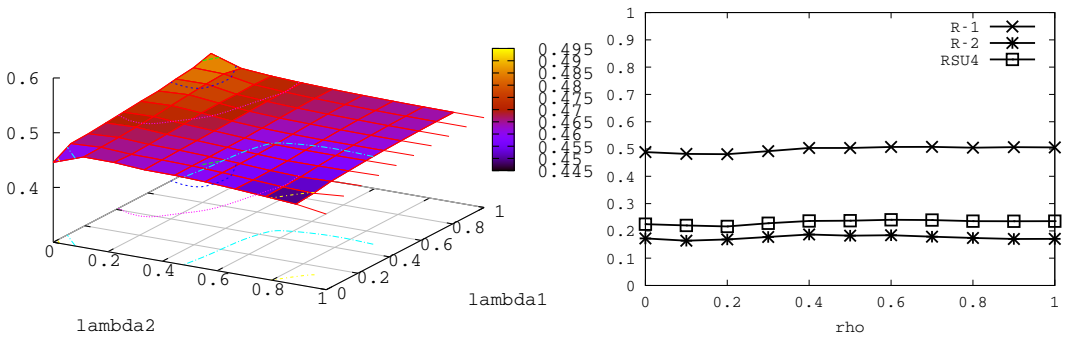
Table 3: Comparison of the recall of named entities of ILP-BL and our method.

Method	Our Complete Model	−EC	−SR	−SecRel	−CityRel
R-1	**0.4804**	0.4520	0.4657	0.4672	0.4796
R-2	**0.1715**	0.1430	0.1669	0.1652	0.1685
RSU4	**0.2144**	0.1987	0.2028	0.2039	0.2120

Table 4: Summarization results of the degenerate versions of our method. "−" means removing this component from our complete method. The table shows the average results over data sets of all cites.

Considering that an importance difference between our method and ILP-BL is our focus on points of interest, we further compared ILP-BL and our method using a different metric. The objective is to test the coverage of points of interest in our generated summaries versus the summaries generated by ILP-BL. To this end, we first identify all the named entities in the model summaries using the Stanford NER tool. We then check the percentage of these named entities covered in the generated summaries and report these recall scores in Table 3. We can see that for majority of the 32 section-specific summaries, our method clearly has a higher recall score than ILP-BL, showing that our method generates summaries with more potential points of interest.

To further understand whether all the components of our improved ILP method have contributed to the performance improvement, we compare our overall method with a few degenerate versions of our method. In each degenerate version, we remove a single component of the objective function. The results are shown in Table 4, where −EC removes the consideration of entity coverage (i.e. setting $\lambda_1 + \lambda_2 = 1$), −SR removes the consideration of sentence relevance (i.e. setting $\lambda_2 = 0$), −SecRel removes only the section-specific relevance of the sentences (i.e. setting $\rho = 0$), and −CityRel removes only the city-specific relevance of the sentences (i.e. setting $\rho = 1$). We can see that each degenerate version of our method performs worse than the complete method, which shows that all components of the objective function are useful. In particular, entity coverage and section-specific relevance seem to be the more important components.



(a) R-1(λ_1, λ_2) (b) ρ

Figure 2: Summarization performance of our method by varying the value of the parameters λ_1, λ_2 and ρ.

5.3 Analysis of Topic Words

We show some further analysis of our results. To begin with, we analyze the learning results of JCSM. The top words in city-specific word distributions and section-specific word distributions learnt by JCSM are presented in Table 5 and Table 6. Generally we observe clean top words for each city and each section. For each city, city-specific words are those associated with the corresponding city. For example, for Singapore, we see words such as "s$" (Singapore dollars), "sentosa" (an island resort in Singapore), "orchard" (a boulevard that is the retail and entertainment hub of Singapore) and "bugis" (a popular shopping place). For New York City, we see "square", "times" and "manhattan". For each section, section-specific words are those words which frequently appear when people discuss about this section, such as "menu", "dishes" and " seafood" for the restaurant section and "train", "bus" and "station" for the transport section.

Topic 1 *Singapore*	Topic 2 *SFO*	Topic 3 *Chicago*	Topic 4 *Boston*	Topic 5 *LA*	Topic 6 *NYC*	Topic 7 *Seattle*	Topic 8 *Pairs*	Topic 9 *London*	Topic 10 *Sydney*
singapore	sf	chicago	boston	beach	york	downtown	paris	london	sydney
s$	san	downtown	end	hollywood	nyc	seattle	de	tube	harbour
centre	francisco	park	north	los	park	needle	metro	underground	beach
food	gate	city	downtown	angeles	central	space	eiffel	central	manly
shopping	golden	neighborhood	fenway	la	square	market	french	centre	beaches
sentosa	bay	north	bay	downtown	times	rain	la	british	house
road	bart	lake	harvard	drive	manhattan	place	du	palace	opera
orchard	union	mile	place	california	broadway	pike	tower	thames	quay
chinese	wharf	loop	city	miles	city	center	des	end	australian
mrt	muni	ave	college	hills	street	waterfront	rue	kensington	rocks
bugis	square	field	subway	long	east	area	le	station	bridge

Table 5: Top city specific words discovered by JCSM.

Topic 1 *restaurants*	Topic 2 *hotels*	Topic 3 *shopping*	Topic 4 *sights*	Topic 5 *entertainment*	Topic 6 *activities*	Topic 7 *transport*	Topic 8 *thingsToDo*
food	hotel	shop	museum	bar	visit	train	bar
restaurant	rooms	store	city	music	park	bus	place
menu	free	stores	park	club	tour	station	tour
dishes	wi-fi	shopping	art	place	fun	airport	city
place	walk	shops	building	night	city	time	food
bar	located	find	built	dance	walk	line	art
chicken	offers	clothes	world	beer	day	car	day
fish	station	wear	place	clubs	time	walk	including
fresh	features	mall	house	crowd	shopping	minutes	music
seafood	tv	place	area	bars	museum	hours	restaurant

Table 6: Top section specific words discovered by JCSM.

5.4 Parameter Sensitivity Analysis

We further give parameter sensitivity analysis for our proposed method. We show how sensitive our results are with respect to the parameters λ_1, λ_2 and ρ. We choose the Sydney data set to perform parameter sensitivity analysis. In Figure 2(a), we show how ROUGE-1 varies with respect to λ_1 and λ_2. We can see that the performance fluctuates within a limited range as we vary λ_1 and λ_2. We find the trend for ROUGE-2 and ROUGE-SU4 is similar so we leave out the figures for them. In Figure 2(b) we see that the performance is pretty stable as we vary ρ.

5.5 Sample Output and Case Study

Finally, we show a sample travel guide our method generates for Sydney in Table 7. We can see that first of all the sentences selected by our method have high relevance to the corresponding sections. Second, through observation we find that humans tend to select sentences containing more points of interest as summary. Our summary sentences contain many points of interest as highlighted, showing the advantage of our method.

Sample Summary Sentences Generated from Yahoo! Answers by Our Method for Sydney
Hotel
Sorry can not recommend you a hotels as I have no idea of pricing , but if you want a nice area , check hotels in **Bondi** and **Manly Beaches** . As for the **Acer Arena** , that is in the **Homebush Olympic Park** and you can choose to live in either **Parramatta** or the city . You need to live in one of the surrounding residential suburbs , close to a train line . Try **Alexandria** , **Newtown** , **Surry hills** for inner suburbs . . . .
Sights
You can walk around the harbor area to the **Opera House** and you can see the beautiful **Harbor Bridge** . All this is apart from the **Opera House** and the **Botanical Gardens** . Visit the **Custom House Circular Quay** and see a model of **Sydney**. You must also do a day trip to the **Blue Mountains** . **Harbour Wedding** is one of the major attraction in **Sydney** . . . .
Entertainment
George Street has a number of bars . All the bars around the harbour are really good day and night . If you want to stay in a hotel where there is entertainment at night , you could look at **Woolloomooloo** , **Darlinghurst** , **Surry Hills** , **Kings Cross** or **Potts Point** . **Newtown** is good for bars . Get them to see a theatre show or something at the **Opera House** . . . .
Things-to-do
If you are going out for the day , starting with a walk to the city will be most enjoyable . Take a public ferry from **Circular Quay** to **Darling Harbour** , about 15 minutes across the harbour and under the bridge , when you get to **Darling Harbour** go and see the **Chinese Gardens** . There are lots of interesting things to see and do in and around **Sydney** . . . .
Activities
They have good markets at the weekend and great views of the **Opera House** . The **Opera House** is free to have a look at , if you like art then walk through the **Botanical Gardens** and go and see the art gallery . If you 're feeling brave , you can do a **Harbour Bridge** walk , though I think it may be a little pricey . . . .

Table 7: Excerpts from the summary generated from Yahoo! Answers by our method for Sydney. We show summaries for the 5 sections other than the 3 sections shown in Table1. Named entities are highlighted in bold font.

6 Related Work

Multi-document summarization is a process to generate a text summary by reducing documents in size while retaining the main points of the original documents. It has been extensively studied in the NLP community, with most efforts on extractive summarization. Our work is also based on extractive summarization. Extractive summarization essentially selects a set of sentences from the original documents to form a summary.

To select sentences, different features and ranking strategies have been studied. Early work focuses on finding good features to select summary sentences. Radev et al. (2004) proposed a centroid-based summarizer which combines several pre-defined features like tfidf, cluster centroid and position to score sentences. Lin and Hovy (2002) built the NeATS multi-document summarization system using term frequency, sentence position, stigma words and simplified Maximal Marginal Relecvance (MMR). Nenkova et al. (2006) proved that high-frequency words were significant in reflecting the focus of documents. Ouyang et al. (2010) studied the influence of different word positions in summarization. Later, graph-based ranking algorithms have been successfully applied to summarization. LexPageRank (Erkan and Radev, 2004a) is a representative one based on the PageRank algorithm (Page et al., 1999). Later extensions include ToPageRank (Pei et al., 2012), which incorporates topic information into the propagation mechanism, the manifold-ranking based method for topic-focused summarization (Wan et al., 2007) and DivRank (Mei et al., 2010), which introduces a time-variant matrix into a reinforced random walk to balance prestige and diversity.

More recently, Integer Linear Programming (ILP) based framework was introduced as a global inference algorithm for multi-document summarization by McDonald (2007), which considers information and redundancy at the sentence level. Gillick and Favre (2009) studied information and redundancy at a sub-sentence, "concept" level, modeling the value of a summary as a function of the concepts it covers. In our work we also model concept level coverage of the summaries. Li et al. (2013) proposed a regression model to estimate the frequency of bigrams in the reference summary and analyzed the impact of bigram selection, weight estimation and ILP setup. Haghighi and Vanderwende (2009) constructed a sequence of generative probabilistic models for multi-document summarization, exhibiting ROUGE gains along the way. Sauper and Barzilay (2009) investigated an approach for creating a comprehensive textual overview of subject composed of information drawn from the Internet and applied ILP to optimize both local fit of information into each topic and global coherence across the entire overview. Li et al. (2011) developed an entity-aspect LDA model to cluster sentences into aspects and then extend LexRank algorithm to rank sentences. Hu and Wan (2013) proposed to use SVR model and ILP method to generate presentation slides for academic papers.

Our work is different from standard ILP-based multi-document summarization. We designed a latent variable model to first separate the threads to be summarized into sections based on model gravel guides.

We also emphasized the inclusion of potential points of interest in formulating the ILP optimization problem.

Our work is also closely related to previous work on answer summarization in community-based QA sites. Previous work on summarizing answers is mainly based on query focused multi-document summarization techniques to summarize multiple answer documents given a single question. Liu et al. (2008) proposed a CQA question taxonomy to classify questions in CQA and question-type oriented answer summarization for better reuse of answers. Tomasoni and Huang (2010) proposed two concept-scoring functions to combine quality, coverage, relevance and novelty measures for answer summary in response to a question and showed that their summarized answers constitute a solid complement to best answers voted by CQA users. Chan et al. (2012) presented an answer summarization method for complex multi-sentence questions. For our work, we study a new problem of summarizing multiple threads to automatically generate city travel guides based on known template structure from well-written travel guides, which is different from the setting of single Q&A thread summarization in the previous related studies.

7 Conclusion and Future Work

In this paper we proposed a summarization framework to generate well structured supplementary travel guides from social media based on a latent variable model and integer linear programming. The latent variable model could align forum threads with the section structure of well-written travel guides. Compared to standard concept based ILP methods, our method additionally tries to cover more named entities as points of interest and maximizes sentence relevance scores measured by section-specific and city-specific word distributions learnt by the latent variable model. Extensive experiments with real data from Yahoo! Answers show that our proposed method is able to generate better summaries compared with a number of multi-document summarization baselines measured by ROUGE scores.

Currently our generated summaries may have overlap with the well-written model travel guides. In the future, we plan to improve our method to emphasize the selection of *additional* information from social media compared with the model travel guides. We will also look into the problem of how to summarize information that does not fit into the template structure derived from model travel guides.

Acknowledgments

This work was done during Liu Yang's visit to Singapore Management University. The authors would like to thank the reviewers for their valuable comments on this work.

References

Wen Chan, Xiangdong Zhou, Wei Wang, and Tat-Seng Chua. 2012. Community answer summarization for multi-sentence question with group l1 regularization. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 582–591, Stroudsburg, PA, USA. Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004a. Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 4 of *EMNLP '04*.

Günes Erkan and Dragomir R. Radev. 2004b. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479, December.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, ILP '09, pages 10–18, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 362–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yue Hu and Xiaojun Wan. 2013. Ppsgen: Learning to generate presentation slides for academic papers. In *Proceedings of the 21st international jont conference on Artifical intelligence*, IJCAI '13, pages 2099–2105.

Peng Li, Yinglin Wang, Wei Gao, and Jing Jiang. 2011. Generating aspect-oriented multi-document summarization with event-aspect model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1137–1146, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chen Li, Xian Qian, and Yang Liu. 2013. Using supervised bigram-based ilp for extractive summarization. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*, ACL '13, pages 1004–1013, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2002. From single to multi-document summarization: A prototype system and its evaluation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 457–464, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 71–78, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 497–504, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European Conference on IR Research*, ECIR'07, pages 557–564, Berlin, Heidelberg. Springer-Verlag.

Qiaozhu Mei, Jian Guo, and Dragomir Radev. 2010. Divrank: The interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 1009–1018, New York, NY, USA. ACM.

Ani Nenkova, Lucy Vanderwende, and Kathleen McKeown. 2006. A compositional context sensitive multi-document summarizer: Exploring the factors that influence summarization. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 573–580, New York, NY, USA. ACM.

You Ouyang, Wenjie Li, Qin Lu, and Renxian Zhang. 2010. A study on position information in document summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 919–927, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November. Previous number = SIDL-WP-1999-0120.

Yulong Pei, Wenpeng Yin, and Lian'en Huang. 2012. Generic multi-document summarization using topic-oriented information. In *Proceedings of the 12th Pacific Rim International Conference on Trends in Artificial Intelligence*, PRICAI'12, pages 435–446, Berlin, Heidelberg. Springer-Verlag.

Dragomir R. Radev, Hongyan Jing, Malgorzata Styś, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Inf. Process. Manage.*, 40(6):919–938, November.

Christina Sauper and Regina Barzilay. 2009. Automatically generating wikipedia articles: A structure-aware approach. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 208–216, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mattia Tomasoni and Minlie Huang. 2010. Metadata-aware measures for answer summarization in community question answering. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 760–769, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. 2007. Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2903–2908, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Xiaojun Wan. 2008. An exploration of document impact on graph-based multi-document summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 755–762, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ensemble-Based Medical Relation Classification

Jennifer D'Souza and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{jld082000,vince}@hlt.utdallas.edu

Abstract

Despite the successes of distant supervision approaches to relation extraction in the news domain, the lack of a comprehensive ontology of medical relations makes it difficult to apply such approaches to relation classification in the medical domain. In light of this difficulty, we propose an ensemble approach to this task where we exploit human-supplied knowledge to guide the design of members of the ensemble. Results on the 2010 i2b2/VA Challenge corpus show that our ensemble approach yields a 19.8% relative error reduction over a state-of-the-art baseline.

1 Introduction

Medical relation (MR) classification, an information extraction task in the clinical domain that was recently defined in the 2010 i2b2/VA Challenge (Uzuner et al., 2011), involves determining the relation between a pair of medical concepts (problems, treatments, or tests). The ability to classify MRs is indispensable to sound automatic analysis of patient health records.

While MR classification is a relatively new task, there has been a lot of work on extracting semantic relations from news articles. *Supervised approaches* train classifiers on data annotated with the target relation types, typically using a rich feature set (Zhou et al., 2005; Surdeanu and Ciaramita, 2007; Zhou et al., 2007). Since obtaining annotated data is a time-consuming and labor-intensive process, researchers have considered *unsupervised approaches* (Shinyama and Sekine, 2006; Banko et al., 2007). While unsupervised approaches can use a large amount of unannotated data and extract a large number of relations, it may not be easy to map the resulting relations to those needed for a given knowledge base. One way to mitigate this problem is *semi-supervised learning*: starting from a given set of seed instances, a bootstrapping algorithm is used to iteratively learn extraction patterns and extract instances (Brin, 1999; Riloff and Jones, 1999; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002; Etzioni et al., 2005; Pantel and Pennacchiotti, 2006; Bunescu and Mooney, 2007; Rozenfeld and Feldman, 2008). However, the resulting patterns often suffer from semantic drift and low precision. Recent years have seen a surge of interest in *distant supervision* for relation extraction (Mintz et al., 2009; Nguyen and Moschitti, 2011; Krause et al., 2012; Min et al., 2013). The idea is to automatically create annotated relation instances by extracting their labels from relation instances in a knowledge base such as Freebase (Bollacker et al., 2008) and YAGO (Suchanek et al., 2007).

Our goal in this paper is to advance the state of the art in MR classification. One of the major challenges in MR classification is the scarcity of labeled data. At first glance, we can mitigate this problem using distant supervision approaches. However, there is difficulty in applying these approaches to MR classification: only one of the relation types defined in the 2010 i2b2 Challenge is represented in the Unified Medical Language System[1], the most comprehensive medical ontology available to date.

In light of this difficulty, we propose an *ensemble* approach to MR classification, where we exploit *human-supplied knowledge* to guide the design of different members of the ensemble. Unlike state-of-the-art supervised approaches to this task, which represent contextual information largely as *flat* (i.e.,

[1] www.nlm.nih.gov/research/umls/

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1682–1693, Dublin, Ireland, August 23-29 2014.

discrete- or real-valued) features (de Bruijn et al., 2011; Rink et al., 2011) or *structured tree* features (Zhu et al., 2013), we represent contexts as *sequences*, specifically *word* sequences and *dependency* sequences, and use them to derive lexical and dependency patterns. Our ensemble approach exploits human-supplied knowledge in three ways. First, while existing approaches employ similarity functions already defined in off-the-shelf learning algorithms (e.g., linear kernel (Rink et al., 2011), tree kernel (Zhu et al., 2013)) to compute the similarity between two relation instances, we define functions to compare the similarity between two patterns. Second, to complement the automatically induced patterns, we hand-craft patterns based on manual observations made on the training set, specifically by having a human identify the contexts of two concepts that are strongly indicative of a medical relation class. Finally, we employ human knowledge to identify the *constraints* on the classification of different relation instances, and enforce the resulting constraints in an integer linear programming (ILP) framework. Evaluation results on the 2010 i2b2/VA Challenge corpus (henceforth the *i2b2 corpus*) show that our ensemble approach yields a 19.8% relative error reduction over a state-of-the-art system.

The rest of the paper is organized as follows. Section 2 provides an overview of the i2b2 corpus. Section 3 describes the baseline systems. Sections 4 and 5 describe our new components and our ensemble approach. Section 6 discusses our constraints for enforcing global consistency. We present evaluation results in Section 7, conduct an error analysis in Section 8, and conclude in Section 9.

2 Corpus

For evaluation, we use the i2b2 corpus, which comprises 426 de-identified discharge summaries. We adopt the i2b2 organizers' partition of the 426 summaries into a training set (170 summaries) and a test set (256 summaries). As many of the algorithms in our approach require parameter tuning, we reserve 30 of the 170 summaries in the training set for development purposes.

In each discharge summary, the *concepts* and the *medical relation* between each pair of concepts are marked up. Each concept is annotated with a type attribute that indicates whether it is a TEST, a PROBLEM, or a TREATMENT. In addition, a PROBLEM concept has an *assertion* attribute, which specifies whether the problem was present, absent, or possible in the patient, conditionally present in the patient under certain circumstances, hypothetically present in the patient at some future point, or mentioned in the patient report but associated with someone other than the patient.

Eleven types of intra-sentential pairwise relations are annotated. A brief description of these relation types and the relevant statistics are provided in Table 1. As we can see from the table, each medical relation has a *type* and is defined only on *intra-sentence* TREATMENT-PROBLEM, TEST-PROBLEM, and PROBLEM-PROBLEM concept pairs. Also, while there are 11 relation types, three of them, namely Relations 6, 9, and 11, denote the *absence* of a medical relation between the corresponding concepts. The purpose of having "no relation" classes is to ensure that every pair of TEST/PROBLEM/TREATMENT concepts is annotated, whether or not a medical relation exists between them.

3 Baseline MR Classification Systems

We employ two supervised MR classification systems as baselines. The first baseline is a state-of-the-art system that achieved the best performance in the official 2010 i2b2 evaluation. The second baseline is a tree kernel-based system, motivated by the fact that tree kernels are frequently used in relation extraction (e.g., Zhou et al. (2007), Zhu et al. (2013)).

3.1 SVM with Flat Features

Our first baseline, Rink et al.'s (2011) system, employs an SVM classifier trained on a set of *flat* features (i.e., features that are discrete- or real-valued).

Following Rink et al., we create training instances as follows. First, we form training instances between every pair of (PROBLEM, TEST, and TREATMENT) concepts in the training documents, labeling an instance with its relation type. Since the instances belonging to the three "no relation" classes significantly outnumber those belonging to the remaining eight classes, we reduce data skew by downsampling

Id	Relation	Example	Total	(%)
1	**TrIP**: Treatment improves medical problem	*Her pain* resolved after *surgery*	203	(0.6)
2	**TrWP**: Treatment worsens medical problem	treated with *Zofran* with *no relief*	133	(0.4)
3	**TrCP**: Treatment causes medical problem	*Transdermal nitroglycerin* caused *headache*	526	(1.8)
4	**TrAP**: Treatment is administered for medical problem	start on *Decadron* 4 mg q6 to prevent *swelling*	2613	(8.9)
5	**TrNAP**: Treatment is not administered because of medical problem	*His Avandia* was discontinued secondary to *the side effect profile*	174	(0.6)
6	**NTrP**: No relation between treatment and problem	with *sutures* intact and no *erythema* or purulence noted .	4462	(15.2)
7	**TeRP**: Test reveals medical problem	*A postoperative MRI* revealed no *remarkable findings*	3051	(10.4)
8	**TeCP**: Test conducted to investigate medical problem	*An ultrasound* was done to rule out *cholestasis*	504	(1.7)
9	**NTeP**: No relation between test and problem	Throughout the stay *his labs* remained normal and *his pain* controlled .	2964	(10.1)
10	**PIP**: Medical problem indicates medical problem	with *a moderate-sized , dense , fixed inferior defect* indicative of *scar*	2202	(7.5)
11	**NPP**: No relation between paired medical problems	He is somewhat *cantankerous* and *demanding* of the nurses .	12503	(42.6)

Table 1: The 11 relation types for medical relation classification. Each relation type is defined on an ordered pair where concepts in the pair are as specified by the relation. The "Total" and "%" columns show the number and percentage of instances annotated with the corresponding relation type over all 426 discharge summaries, respectively.

instances belonging to the three "no relation" classes.[2] Specifically, we downsample the instances belonging to the three "no relation" classes (i.e., **NTrP**, **NTeP**, and **NPP**) by ensuring that (1) the ratio of the number of **NTrP** instances to the number of TREATMENT-PROBLEM instances is 0.06; (2) the ratio of the number of **NTeP** instances to the number of TEST-PROBLEM instances is 0.03; and (3) the ratio of the number of **NPP** instances to the number of PROBLEM-PROBLEM instances is 0.5. These ratios are selected using our 30-summary development set, as described in Section 2. As mentioned above, each instance corresponds to a pair of concepts, c_1 and c_2, and is represented using 37 groups of features that can be divided into five categories:[3]

Context (13 groups). The words, the POS tags, the bigrams, the string of words, the sequence of phrase chunk types, and the concept types used between c_1 and c_2; the word preceding c_1/c_2; any of the three words succeeding c_1/c_2; the predicates associated with c_1/c_2; the predicates associated with both concepts; and a feature that indicates whether a conjunction regular expression matched the string of words between c_1 and c_2.

Similarity (5 groups). We find the concept pairs in the training set that are most similar to the (c_1,c_2) pair (i.e., its nearest neighbors), and create features that encode the statistics collected from these nearest neighbors. To find the nearest neighbors, we (1) represent each pair in the training set as a sequence; (2) define the number of nearest neighbors to use; and (3) define a similarity metric to compute the similarity of two sequences.

Following Rink et al. (2011), we employ five methods to represent a pair. The five methods are: (1) as a sequence of POS tags for the entire sentence containing the pair; (2) as a phrase chunk sequence between the two concepts; (3) as a word lemma sequence beginning the two words before the first concept, up to and including the second word following the second concept in the pair; (4) as a concept type sequence for all the concepts found in the sentence containing the pair; and (5) as a shortest dependency path sequence connecting the two concepts. Table 2 shows an example of these five methods of generating sequences from the TEST concept *her exam* and the PROBLEM concept *her hyperreflexia* in the sentence

[2]Other methods for addressing class imbalance, such as over-sampling (Chawla et al., 2002) and cost-sensitive learning (Turney, 1995), can also be employed.

[3]To compute the features, we use (1) the Stanford CoreNLP tool (Manning et al., 2014) to obtain POS tags, word lemmas, and dependency structures; (2) GENIA (`http://www.nactem.ac.uk/tsujii/GENIA/tagger`) to obtain phrase chunks; and (3) SENNA (Collobert et al., 2011) to obtain predicate-argument structures.

Generation Method	Sequence
(1)	RB VB , test$_{c1}$ RB VBD RB IN problem$_{c2}$.
(2)	ADVP VP ADVP PP
(3)	postop , test$_{c1}$ only improve slightly in problem$_{c2}$.
(4)	test$_{c1}$ problem$_{c2}$
(5)	test$_{c1}$–nsubj–> prep <–pobj–problem$_{c2}$

Table 2: Examples of the five methods of sequence generation.

Postop, her exam only improved slightly in her hyperreflexia . Note that for better generalization, the two concepts are replaced with their concept type (i.e., *her exam* and *her hyperreflexia* are replaced with test$_{c1}$ and problem$_{c2}$ respectively) before sequence generation. Like Rink et al., we seek different numbers of nearest neighbors for the five methods of generating sequences. For the first method, we use 100 nearest neighbors; for the second method, 15 neighbors; for the third method, 20 neighbors; for the fourth method, 100 neighbors; and for the fifth method, 20 neighbors. We use the Levenshtein distance (Levenshtein, 1966) as the similarity metric.

After finding the nearest neighbors for each of the five methods of sequence representation, we create features as follows. For each method, we compute the percentage of nearest neighbors belonging to each of the 11 relation types, and then create 11 features whose values are these 11 numbers.

Single concept (11 groups). Any word lemma from c_1/c_2; any word used to describe c_1/c_2; the concept type for c_1/c_2; the string of words in c_1/c_2; the concatenation of assertion types for both concepts; and the sentiment category (i.e., positive or negative) of c_1/c_2 obtained from the General Inquirer lexicon (Stone et al., 1968).

Wikipedia (6 groups). Six features are computed based on the Wikipedia articles, their categories, and the links between them. The first feature encodes whether neither c_1 nor c_2 contains any substring that may be matched against the title of an article. The second feature encodes whether the links between the articles retrieved based on the two concepts are absent. The next two features encode whether a link exists from the article pertaining to c_1 (c_2) to the article pertaining to c_2 (c_1). The fifth feature encodes whether there are links between the articles pertaining to both concepts. The last feature encodes whether both concepts have the same concept type according to their Wikipedia categories.

Vicinity (2 groups). The concatenation of the type of c_1 and the type of the closest concept preceding c_1; and the concatenation of the type of c_2 and the type of the closest concept succeeding c_2.

After creating the training instances, we train a 11-class classifier on them using SVM^{multiclass} (Tsochantaridis et al., 2004). We set C, the regularization parameter, to 10,000, since preliminary experiments indicate that preferring generalization to overfitting (by setting C to a small value) tends to yield poorer classification performance. The remaining learning parameters are set to their default values. After training, we use the resulting classifier to make predictions on the test instances, which are generated in the same way as the training instances.

3.2 SVM with Structured Feature

In this framework, each instance is represented using a single structured feature computed from the parse tree of the sentence containing the concept pair. Since publicly available SVM learners capable of handling structured features can only make binary predictions, we train 11 SVM classifiers, one for representing each medical relation. In each classifier's training data, a positive instance is one whose class value matches the medical relation class value of the classifier, and a negative instance is one with other class values applicable to the given concept pair. Since the negative instances significantly outnumber the positive instances in each of these binary classifiers, we reduce data skew by downsampling the negative instances. Following the order of the 11 relations listed in Table 1, the optimal ratios of negative-to-positive instances according to our 30-summary development set are 0.2, 0.2, 0.06, 0.2, 0.5, 1, 1, 0.3, 0.06, 0.06, and 0.09, respectively. We set C to 100 based on the development data.

While we want to use a parse tree directly as a feature for representing an instance, we do *not* want to use the *entire* parse tree as a feature. Specifically, while using the entire parse tree enables a richer

representation of the syntactic context of the two concepts than using a *partial* parse tree, the increased complexity of the tree also makes it more difficult for the SVM learner to make generalizations.

To strike a better balance between having a rich representation of the context and improving the learner's ability to generalize, we extract a subtree from a parse tree and use it as the value of the structured feature of an instance. Specifically, given two concepts in an instance and the associated syntactic parse tree T, we retain as our subtree the portion of T that covers (1) all the nodes lying on the shortest path between the two entities, and (2) all the immediate children of these nodes that are not the leaves of T. This subtree is known as a simple expansion tree.

After training the 11 tree kernel-based relation classifiers, we can apply them to classify a test instance. The class value of an instance is determined based on the classifier with the maximum classification confidence, where the confidence value of an instance is its signed distance from the SVM hyperplane.

4 Exploiting Sequences for MR Classification

Unlike the two baselines, which exploit flat features and parse-based structured features for MR classification, in this section we describe three MR classification systems that exploit sequences.

4.1 Dependency-Based Sequences

The first system is based on sequences of *dependency* relations. To see why dependency relations could be useful for MR classification, consider the sentences in Table 3:

(1)	nsubjpass$_{e1}$ prep$_{e2}$ pobj$_{e2}$ *His pain* was well controlled with *oral medication* at this time .
(2)	nsubjpass$_{e1}$ prep$_{e2}$ pobj$_{e2}$ conj$_{e2}$ *The patient's pain* from his catheters was controlled with Oxycodone and *Motrin* .

Table 3: Example dependency paths.

In sentences (1) and (2), the PROBLEM concepts *His pain* and *The patient's pain* occur as the subject of the verb controlled and the TREATMENT concepts *oral medication* and *Motrin* occur as objects of the prepositional with modifier of the same verb controlled. In other words, intuitively, the verb controlled cues that the PROBLEM concept is being controlled, and together with the preposition with it cues that the TREATMENT concept is doing the controlling. Note that in each case the relation between the PROBLEM and TREATMENT is **TrIP**, which can now be easily inferred given the dependency relations of the concept pairs with the verb controlled. These examples suggest that the verb closest to each of the two concepts is an important word as it cues the relation.

Given the usefulness of dependency structures and the verb closest to each concept for MR classification, we represent each training/test instance as a paired dependency sequence with separate dependency paths traced from each concept in the pair to its closest verb. To reduce data sparsity, for the argument words found in a dependency path, we replace them with their POS tags. For example, given sentence (1), the path extracted from *His pain* is "nsubjpass (controlled NN)" and from *oral medication* is "prep (controlled with) pobj (with NN)".[4]

Next, we describe how to classify a test instance *inst*. First, we identify the set of training instances T that satisfy two conditions: (a) the ancestor verb pair in the training instance is the same as that in *inst*, and (b) each of the two dependency sequences in the training instance either is the same as, or contains, or is contained in the corresponding dependency sequences in *inst*. Second, we find for *inst* its nearest neighbor in T by employing the following similarity function:

$$Similarity(train, test) = cosine(path_{train_{c_1}}, path_{test_{c_1}}) \cdot cosine(path_{train_{c_2}}, path_{test_{c_2}}) \quad (1)$$

[4]Note that sometimes a dependency path cannot be traced (e.g., a verb does not exist, which is not uncommon in a discharge report) for a given concept pair. If this happens, no instance will be generated from the concept pair.

where $cosine(x, y)$ is a function that computes the cosine similarity of x and y.[5] Finally, if the similarity between *inst* and its nearest neighbor in T is greater than a threshold, we classify *inst* using the class value of its nearest neighbor.[6] Otherwise, this system will leave *inst* unclassified. In other words, this system is precision-oriented, classifying only those instances it can classify confidently.

4.2 Lexical Patterns

In our second system, we represent each concept pair as a lexical pattern. Specifically, we employ Generation Method 3 as described in the Similarity features in Section 3.1 to generate a lexical pattern from a concept pair. To classify a test instance *inst*, we employ the one-nearest-neighbor algorithm. To identify the nearest neighbor of *inst*, we employ the Levenshtein distance as the similarity metric.

Two questions naturally arise. First, since these lexical patterns have already been used to generate features in the flat-feature baseline, why do we still employ them in a separate system? To answer this question, note that although these lexical patterns were used to generate features for training the flat-feature baseline classifier, we have no control over whether these features are deemed useful by the learning algorithm and are subsequently used by the resulting classifier. Having a separate system that employs these patterns ensures that they will be used when making the final classification decision.

Second, given that we described five methods to generate sequences in Section 3.1, why do we employ Generation Method 3 but not the remaining methods? In principle, we can employ the remaining four generation methods for generating lexical patterns as well: all we need to do is to create four additional systems, each of which makes use of the patterns created by exactly one of the four methods. In practice, however, not all generation methods are equally good: if a method does not generate patterns that adequately capture context, then employing the resulting patterns may yield poor-performing systems. Consequently, we employ only the system corresponding to the generation method that yields the best performance on the development data, which turns out to be the system corresponding to Method 3.

4.3 Rules

In the previous subsection, we employ automatically induced patterns. In contrast, our third system employs patterns that are *hand-crafted* based on manual observations made on the *training* set. Specifically, we ask a human to identify the contexts of two concepts that are strongly indicative of a relation class. Like the automatically induced patterns, each hand-crafted pattern is composed of the types of the two concepts involved and the context in which they occur. For example, in the pattern *due to* PROBLEM *by* TREATMENT, the TREATMENT is likely to cause the PROBLEM and therefore it will be labeled as **TrCP**. As another example, in the pattern *attributed to* PROBLEM *as a result of* PROBLEM, the two PROBLEMs are likely to have an indicative relation and therefore it will be labeled as **PIP**. At the end of this process, we end up with 136 manually labeled patterns, which we will subsequently refer to as a *ruleset*.

Next, we order these rules in decreasing order of accuracy, where the accuracy of a rule is defined as the number of times it yields the correct MR type divided by the number of times it is applied, as measured on the training set.

Given this ruleset, we can classify a test instance using the first applicable rule in it. If no rules are applicable, the test instance will remain unclassified.[7]

5 The Ensemble

In the previous section, we described three systems for MR classification. Together with the two baseline systems, we have five systems for MR classification. A natural way to make use of all of them for MR classification is to include them in an ensemble. The question, then, is: how do we classify a test instance using this ensemble? The simplest approach is perhaps majority voting, but that presumes that each

[5] To apply cosine similarity, we represent each path as a frequency-count vector, where each dimension in the vector corresponds to a dependency type or an argument word appearing in the path.

[6] Based on development set experiments, the similarity threshold values for each concept pair type are: $T_{Treatment-Problem}$ = 0.85; $T_{Test-Problem}$ = 0.75; and $T_{Problem-Problem}$ = 0.75.

[7] Space limitations preclude a complete listing of these rules. See our website at `https://www.hlt.utdallas.edu/~jld082000/medical-relations/` for the complete list of rules.

member of the ensemble is equally important. In practice, however, some members are more important than the others, so the votes cast by these members should have higher weights.

To model this observation, we combine the (probabilistic) votes of the members in a weighted fashion using the following formula:

$$P_{combined}(c) = w_1 \cdot P_{tree}(c) + w_2 \cdot P_{flat}(c) + w_3 \cdot P_{dependency}(c) + w_4 \cdot P_{word}(c) + w_5 \cdot P_{rules}(c) \quad (2)$$

where w_i ($i = 1, \ldots, 5$) is a combination weight, and $P_x(c)$ is the probability that the test instance belongs to class c according to system x.

Two questions naturally arise. First, how can the combination weights be determined? We perform an exhaustive search on held-out development data to find the combination of weights that jointly maximizes overall accuracy on the development set. We allow each weight to vary between 0 and 1 in steps of 0.1, subject to the constraint that the five weights sum to 1.

Second, how can $P_x(c)$ be computed? In other words, how can each system compute the probability that a given test instance belongs to a certain class? To answer this question, we have to convert the output of each system for each test instance into a 11-element probability vector, which is used to encode the probability that the given test instance belongs to each of the 11 relation types.

We perform the conversion as follows. For the two baseline systems, the SVM outputs a confidence value for each class. Hence, to obtain the probability vector, we first normalize the confidence value associated with each class so that it falls within the [0,1] range, and then normalize the resulting values so that they sum to 1. For the systems employing lexical patterns and dependency-based sequences, the class chosen by each system receives a probability of 0.6, and each of the other classes applicable to the test instance under consideration receives an equal share of the remaining probability mass. For the rule-based system, we take the rule that is used to classify the test instance and apply this rule to each instance in the training set to estimate the probability that the rule is correct with respect to each of the 11 classes. We can then use the resulting 11 probabilities to create the 11-element probability vector.

Finally, recall that some of these systems are not applicable to all of the test instances. If this happens, the corresponding system(s) will return a vector in which all of its elements are set to 0.

6 Enforcing Global Consistency

So far we have had an ensemble that, given a test instance, returns the probability that it belongs to each of the 11 classes. Since the test instances are classified independently of each other, there is no guarantee that the resulting classifications are globally *consistent*. To enforce global consistency, we employ global constraints implemented in the Integer Linear Programming (ILP) framework (Roth and Yih, 2004).

Since our constraints are intra-sentential, we formulate one ILP program for each sentence s in each training summary. Each ILP program contains $11 \times N_s$ variables, where N_s is the number of test instances formed from the concept pairs in s. In other words, there is one binary indicator variable $x_{i,j,r}$ for each relation class r of each test instance *inst* formed from concept i and concept j, which will be set to 1 by the ILP solver if and only if it thinks *inst* should belong to class r.

Our objective is to maximize the linear combination of these variables and their corresponding probabilities given by the ensemble (see (3) below) subject to two types of constraints, the *integrity* constraints and the *consistency* constraints. The integrity constraints ensure that each concept pair is assigned exactly one relation type (see the equality constraint in (4)). The consistency constraints ensure consistency between the predictions made for different instances in the same sentence.

Maximize:

$$\sum_{(i,j) \in R} \sum_{r \in L} p_{i,j,r} x_{i,j,r} \quad (3)$$

subject to:

$$\sum_{r \in L} x_{i,j,r} = 1 \;\; \forall \, (i,j) \in R \quad (4)$$

Relation	Relations in Conflict
TrIP(tr$_i$,p$_j$)	**TrWP(tr$_i$,p$_k$),TrCP(tr$_i$,p$_m$),TrNAP(tr$_i$,p$_n$)**
TrWP(tr$_i$,p$_j$)	**TrIP(tr$_i$,p$_k$),TrCP(tr$_i$,p$_m$),TrNAP(tr$_i$,p$_n$)**
TrCP(tr$_i$,p$_j$)	**TrIP(tr$_i$,p$_k$),TrWP(tr$_i$,p$_m$),TrNAP(tr$_i$,p$_n$)**
TrAP(tr$_i$,p$_j$)	**TrNAP(tr$_i$,p$_k$)**
TrNAP(tr$_i$,p$_j$)	**TrAP(tr$_i$,p$_k$),TrIP(tr$_i$,p$_m$),TrWP(tr$_i$,p$_n$),TrCP(tr$_i$,p$_o$)**
TeRP(te$_i$,p$_j$)	**TeCP(te$_i$,p$_k$)**
TeCP(te$_i$,p$_j$)	**TeRP(te$_i$,p$_k$)**

Table 4: Constraints on relation types.

and consistency constraints.

Note that (1) $p_{i,j,r}$ is the probability that the instance formed from concept i and concept j belongs to relation type r according to the ensemble; (2) L denotes the set of unique relation types; and (3) R is the set of instances in the sentence under consideration.

The consistency constraints are listed in Table 4. Each row of the table represents a constraint and can be interpreted as follows. If the relation in the first column holds, then none of the relations in the second column can hold. Consider, for instance, the constraint in the first row of the table, which says that if TREATMENT tr_i improves PROBLEM p_j, then tr_i cannot worsen, cause, or be administered for any other PROBLEM. At first glance, it may not seem intuitive that a treatment that improves one problem cannot also worsen or cause other problems. This can be attributed to the way a patient discharge summary is written: while the constraint can be violated for concept pairs in *different* sentences, there is no case in which the constraint is violated for concept pairs in the *same* sentence in the training set. These constraints can be implemented as linear constraints in ILP. For example, the constraint "if TREATMENT tr_i improves PROBLEM p_j, then tr_i cannot worsen PROBLEM p_k" can be implemented as follows.

$$x_{i,j,\text{TrIP}} \leq 1 - x_{i,k,\text{TrWP}} \tag{5}$$

7 Evaluation

7.1 Experimental Setup

Following the 2010 i2b2/VA evaluation scheme, we assume that (1) gold concepts and their types are given, and (2) a medical relation classification system is evaluated on all but the "no relation" types. In other words, a system will not be *directly* rewarded if it correctly identifies a "no relation" instance, but will be penalized if it misclassifies a "no relation" instance as one of the eight relation types.

As mentioned before, we use 170 training summaries from the 2010 i2b2/VA corpus for classifier training and reserve 256 test summaries for evaluating system performance. Thirty training summaries are used for development purposes in all experiments that require parameter tuning.

7.2 Results and Discussion

Table 5 shows the 8-class classification results for our MR classification task, where results are expressed in terms of recall (R), precision (P), and micro F-score (F).

Row 1 and row 2 show the results of the flat-feature baseline and the structured-feature baseline, respectively. As we can see, the flat-feature baseline performs significantly better than the structured-feature baseline.[8] It is worth mentioning that since the dataset available to the research community which we are using contains a subset of the summaries from the dataset that was available to the shared task participants, we were unable to directly compare our system's performance with theirs. Nevertheless, we believe that the results of our reimplementation of Rink et al.'s (2011) system in row 1 can be taken to be roughly the state of the art results on this dataset.

Rows 3–5 show the results of the three systems we introduced. As we can see from row 3, by using simple lexical patterns in combination with the Levenshtein similarity metric, we achieve an F-score that is significantly better than that of the structured-feature baseline but significantly worse (at $p < 0.01$) than

[8] All statistical significance tests are paired t-tests with $p < 0.05$ unless otherwise stated.

	Individual System	R	P	F
1	Flat	66.7	58.1	62.1
2	Tree	64.3	55.6	59.6
3	Lexical Patterns	63.9	59.2	61.4
4	Dependencies	4.3	82.9	8.2
5	Rules	11.9	84.4	9.1

	Ensemble System	R	P	F
6	Ensemble$_{(1+2)}$	69.2	61.3	65.0
7	Ensemble$_{(1+2+3)}$	70.4	63.1	66.6
8	Ensemble$_{(1+2+3+4)}$	70.0	64.7	67.2
9	Ensemble$_{(1+2+3+4+5)}$	71.1	64.8	67.8
10	Ensemble$_{(1+2+3+4+5)}$ + ILP	**72.9**	**66.7**	**69.6**

	Single Classifier	R	P	F
11	Single$_{(1+2)}$	53.0	73.6	61.7
12	Single$_{(1+2+3)}$	54.4	74.7	63.0
13	Single$_{(1+2+3+4)}$	56.4	73.7	63.9
14	Single$_{(1+2+3+4+5)}$	56.3	74.5	64.1
15	Single$_{(1+2+3+4+5)}$ + ILP	58.9	75.0	66.0

	Bagged System	R	P	F
16	Bagging$_{(1+2)}$	54.6	73.6	62.7
17	Bagging$_{(1+2+3)}$	54.5	73.8	62.7
18	Bagging$_{(1+2+3+4)}$	56.9	73.2	64.0
19	Bagging$_{(1+2+3+4+5)}$	56.7	73.9	64.2
20	Bagging$_{(1+2+3+4+5)}$ + ILP	59.2	75.5	66.4

Table 5: Medical relation classification results.

that of the flat-feature baseline. On the other hand, the remaining two systems are precision-oriented: they classify an instance only if they can do so confidently, thus resulting in poor recall.

Rows 6–10 show the results of our ensemble approach when the individual MR classification systems are added *incrementally* to the flat-feature baseline. Except for the addition of the dependency-based system and the hand-crafted rules, which yielded insignificant improvements in F-score, the addition of all other components yielded significant improvements. In fact, every significant improvement in F-score is accompanied by a simultaneous rise in recall and precision. The best-performing system is the one that comprises all of our components, achieving an F-score of 69.6. This translates to a relative error reduction of 19.8% and a highly significant improvement ($p < 0.001$) over our reimplementation of Rink et al.'s (2011) state-of-the-art baseline. The weights learned for the members of the ensemble are indeed different: both baselines have a weight of 0.3, the rule-based system and the lexical patterns have a weight of 0.1, and the remaining weight goes to the dependency-based component.

7.3 Additional Comparisons

Given the above results, a natural question is: is an ensemble approach ever needed to combine the knowledge sources exploited by different systems in order to obtain these improvements? In other words, can we achieve similar performance by training a *single* classifier using a feature set containing all the features currently exploited by different members of the ensemble?

To answer this question, we repeat the experiments in rows 6–10 of Table 5, except that in each experiment we train a single classifier on a feature set formed from the union of those features employed by all the members of the corresponding ensemble. Results are shown in rows 11–15 of Table 5. In each of these five experiments the F-score obtained by our ensemble approach is significantly better than that achieved by the corresponding single-classifier approach. In addition, although we see improvements in F-score as we add the individual extensions (including ILP) incrementally to the flat-feature baseline, none of these improvements is statistically significant. Nevertheless, when applied in combination, these extensions yield a system that is significantly better than the flat-feature baseline. Overall, these results provide suggestive evidence that to achieve the same level of performance we cannot replace our ensemble approach with a simpler setup that relies on a single classifier.

Given that our ensemble approach performs better than a single-classifier approach, a relevant question is: do we have to use *our* ensemble approach, or can we still achieve similar performance by replacing it with a generic ensemble learning method such as *bagging* (Breiman, 1996)?

To answer this question, we repeat the experiments in rows 6-10 of Table 5, except that we train a committee of classifiers using bagging. Recall that in bagging each classifier in the committee is trained on a *bootstrap sample* created by randomly sampling instances with replacement from the training data until the size of the bootstrap sample is equal to that of the training data. In our implementation, we train 20 multi-class SVM classifiers using SVM^{multiclass}. Given a test instance, each member of the committee will independently cast a probabilistic vote, and the class that receives the largest number of probabilistic votes from the committee members will be assigned to the test instance. Results are shown in rows 16–20 of Table 5. In each of these five experiments, the F-score obtained by bagging

is significantly worse that that achieved by our ensemble approach. In fact, comparing bagging and the single-classifier approach, their results are statistically indistinguishable in all but one case (row 11 vs. row 16), where bagging achieves significantly better performance. Like in the single-classifier experiments, in the bagging experiments we see improvements in F-score as we add the individual extensions (including ILP) incrementally to the flat-feature baseline, although the improvements are significant only with the addition of ILP and the dependency-based system. Nevertheless, when applied in combination, these extensions yield a system that is significantly better than the flat-feature baseline. Overall, these results provide suggestive evidence that to achieve the same level of performance we cannot replace our ensemble approach with bagging.

8 Error Analysis

To gain additional insights into our ensemble approach and to provide directions for future work, we conduct an error analysis of our best-performing system.

NTeP confused as TeRP. This is a frequent type of confusion where 34% of the TEST–PROBLEM pairs that do not have a relation are misclassified as having a "Test Reveals Problem" relation. Below are two subcategories of errors commonly made by the system in this confusion category.

- TEST **with numeric results followed by** PROBLEM **concepts in written text**

The following example illustrates this confusion:
. . . [$_{test}$ *mean gradient*] 33 mm , [$_{problem}$ *decreased disc motion*] , [$_{problem}$ *mobile mass in LVOT*] , [$_{problem}$ *mild AI*] , [$_{problem}$ *mild to moderate MR*] . . .

In sentences like the one above where a TEST concept has a numeric result (result of TEST *mean gradient* is *33 mm*), since the TEST concept is already associated with its result, it has no relation with any other concepts in the sentence. While in some cases the system is able to correctly classify the relation between the TEST concept and the first following PROBLEM concept, in almost all cases, it fails to propagate this no relation class down through the other PROBLEMs listed in a series following the TEST concept. For the sentence above, it incorrectly classifies the relation between TEST concept *mean gradient* and each of the PROBLEM concepts *mobile mass in LVOT*, *mild AI*, and *mild to moderate MR* as **TeRP** instead of **NTeP**.

- TEST **reveals** PROBLEM **that is consistent with other** PROBLEM

This is a common error where a TEST concept is classified as revealing two consistent PROBLEM concepts when in actuality it only reveals one of the PROBLEMs. Consider the following sentence:
[$_{test}$*Radiograph*] revealed [$_{problem}$ *bilateral diffuse granular pattern*] consistent with [$_{problem}$ *surfactant deficiency*] .

In this sentence, PROBLEM concept *bilateral diffuse granular pattern* is described as being consistent with another PROBLEM concept *surfactant deficiency*. While the system correctly classifies the pair (*Radiograph*, *bilateral diffuse granular pattern*) as **TeRP**, it misclassifies the pair (*Radiograph*, *surfactant deficiency*) as **TeRP**. In case of the second pair, the TEST concept has no relation with the PROBLEM concept. From this common error type, an insight one can derive is that the system is currently missing knowledge of the association of the two PROBLEMs w.r.t. each other, and thus in turn cannot make an informed decision of which of the two PROBLEMs the TEST concept actually reveals.

PIP confused as NPP. The second major confusion in the system's output concerns misclassifying PROBLEM concept pairs that are indicative of each other as having "no relation". We observe that 39.5% of the **PIP** instances get classified as **NPP**.

- PROBLEM **without another** PROBLEM

In a sentence, if a PROBLEM concept is actually said to be without another PROBLEM, then such a pair is commonly misclassified by the classifier into the no-relation class **NPP** instead of the has-relation class **PIP**. An example of this can be found in the sentence "[$_{problem}$ *Angio site*] was clean , dry , and intact without [$_{problem}$ *bleeding*] or [$_{problem}$ *drainage*] .", where PROBLEM *Angio site* is classified as **NPP** with

both concepts *bleeding* and *drainage*, respectively. Such cases call for domain-specific knowledge that can aid in identifying attributes of PROBLEMs, like that the PROBLEM concepts *bleeding* and *drainage* are commonly associated attributes of the PROBLEM concept *Angio site*. With this information the system is better equipped to recognize that PROBLEM *Angio site* is related to its attributes.

9 Conclusion

We investigated a new approach to the medical relation classification task, where we employed human-supplied knowledge to assist the construction of relation classification systems based on sequences, combined them via an ensemble, and then enforced global consistency using constraints in an ILP framework. Experimental results on the i2b2 corpus show a significant relative error reduction of 19.8% over a state-of-the-art baseline.

Acknowledgments

We thank the three anonymous reviewers for their detailed and insightful comments on an earlier draft of this paper. This work was supported in part by NSF Grants IIS-1147644 and IIS-1219142.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, pages 85–94.

Michele Banko, Michael Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction for the Web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2670–2676.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250.

Leo Breiman. 1996. Bagging predictors. *Machine Learning*, 24:123–140.

Sergey Brin. 1999. Extracting patterns and relations from the World Wide Web. In *The World Wide Web and Databases*, pages 172–183. Springer.

Razvan Bunescu and Raymond Mooney. 2007. Learning to extract relations from the Web using minimal supervision. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 576–583.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Berry de Bruijn, Colin Cherry, Svetlana Kiritchenko, Joel Martin, and Xiaodan Zhu. 2011. Machine-learned solutions for three stages of clinical information extraction: The state of the art at i2b2 2010. *Journal of the American Medical Informatics Association*, 18(5):557–562.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the Web: An experimental study. *Artificial Intelligence*, 165(1):91–134.

Sebastian Krause, Hong Li, Hans Uszkoreit, and Feiyu Xu. 2012. Large-scale learning of relation-extraction rules with distant supervision from the Web. In *Proceedings of the International Semantic Web Conference*, pages 263–278.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 777–782.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011.

Truc Vien T. Nguyen and Alessandro Moschitti. 2011. End-to-end relation extraction using distant supervision from external semantic repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 277–282.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 113–120.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 41–47.

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

Bryan Rink, Sanda Harabagiu, and Kirk Roberts. 2011. Automatic extraction of relations between medical concepts in clinical texts. *Journal of the American Medical Informatics Association*, 18(5):594–600.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning*, pages 1–8.

Benjamin Rozenfeld and Ronen Feldman. 2008. Self-supervised relation extraction from the Web. *Knowledge and Information Systems*, 17(1):17–33.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 304–311.

Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, Cambridge, Massachusetts.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: A core of semantic knowledge. In *Proceedings of the 16th International World Wide Web Conference*, pages 697–706.

Mihai Surdeanu and Massimiliano Ciaramita. 2007. Robust information extraction with perceptrons. In *Proceedings of the NIST 2007 Automatic Content Extraction Workshop*.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st International Conference on Machine Learning*, pages 104–112.

Peter Turney. 1995. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409.

Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2011. 2010 i2b2/VA Challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 427–434.

GuoDong Zhou, Min Zhang, DongHong Ji, and QiaoMing Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 728–736.

Xiaodan Zhu, Colin Cherry, Svetlana Kiritchenko, Joel Martin, and Berry de Bruijn. 2013. Detecting concept relations in clinical text: Insights from a state-of-the-art model. *Journal of Biomedical Informatics*, 46(2):275–285.

Combining Natural and Artificial Examples to Improve Implicit Discourse Relation Identification

Chloé Braud
ALPAGE, Univ Paris Diderot
& INRIA Paris-Rocquencourt
75013 Paris - France
chloe.braud@inria.fr

Pascal Denis
MAGNET, INRIA Lille Nord-Europe
59650 Villeneuve d'Ascq - France
pascal.denis@inria.fr

Abstract

This paper presents the first experiments on identifying implicit discourse relations (i.e., relations lacking an overt discourse connective) in French. Given the little amount of annotated data for this task, our system resorts to additional data automatically labeled using unambiguous connectives, a method introduced by (Marcu and Echihabi, 2002). We first show that a system trained solely on these artificial data does not generalize well to natural implicit examples, thus echoing the conclusion made by (Sporleder and Lascarides, 2008) for English. We then explain these initial results by analyzing the different types of distribution difference between natural and artificial implicit data. This finally leads us to propose a number of very simple methods, all inspired from work on domain adaptation, for combining the two types of data. Through various experiments on the French ANNODIS corpus, we show that our best system achieves an accuracy of 41.7%, corresponding to a 4.4% significant gain over a system solely trained on manually labeled data.

1 Introduction

An important bottleneck for automatic discourse understanding is the proper identification of implicit relations between discourse units. What makes these relations difficult is that they lack strong surface cues like a discourse marker. This point is illustrated in the French examples (1) and (2).[1] In (1), the connective *mais* (*but*) triggers a relation of *contrast*, whereas in (2), there is no explicit connective to signal the *explanation* relation, and the relation has to be inferred through other ways (in this case, a causal relation between having injured players and loosing).

(1) La hulotte est un rapace nocturne, **mais** elle peut vivre le jour.
*The tawny owl is a nocturnal bird of prey, **but** it can live in the daytime.*

(2) L'équipe a perdu lamentablement hier. Elle avait trop de blessés.
The team lost miserably yesterday. It had too many injured players.

Implicit relations are very widespread in naturally-occurring data. Thus, they make up between 39.5% and 54% of the annotated examples in the Penn Discourse TreeBank (PDTB) (Prasad et al., 2008), depending on the relation types used.[2] A quick look at other discourse corpora suggests that the problem is as pervasive (if not more) in other languages. The French ANNODIS corpus does not annotate the distinction between explicit and implicit relations, but a projection of a French connective lexicon on the data gives a proportion of 47.4 to 71% of implicit relations, depending on the set of relations.[3] For the German discourse corpus of (Gastel et al., 2011), (Versley, 2013) report 65% of implicit relations.

In this paper, we tackle the problem of automatically identifying implicit discourse relations in French. To date, the large majority of studies on this task have focused on English, and to a lesser extent on German. Performance remain relatively low compared to explicit relations, due to the lack of strong

[1] All our examples are taken from the ANNODIS corpus: http://redac.univ-tlse2.fr/corpus/annodis/.

[2] The former count does not include *AltLex*, *EntRel* and *NoRel* as implicit examples, whereas the latter does.

[3] The first count does not include *attribution*, *e-elaboration* and *frame* examples.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1694–1705, Dublin, Ireland, August 23-29 2014.

predictors. Because it relies on more complex, interacting factors, the identification of implicit relations requires a lot of data. But the available annotated for French is scarce: while the PDTB contains about $40,000$ examples, the French ANNODIS only has about $3,000$ examples. An additional challenge for building such a system for French compared to English is the lack of external lexical ressources (e.g., semantic verb classification, polarity database).

A natural approach to deal with the lack of annotated implicit data is to resort to additional data automatically obtained from explicit examples in which the connective is removed (Marcu and Echihabi, 2002). Provided that one could reliably identify discourse connectives, this approach makes it possible to create large amounts of additional implicit data from raw texts. Unfortunately, (Sporleder and Lascarides, 2008) show that a system trained on this type of artificially generated data does not generalize well, leading to important performance degradation compared to a system solely trained on natural data.

The central question we address in this paper is how to better leverage the large amount of automatically generated data. We first show that the bad generalization performance of the system trained on artificial data lies in important distribution differences between the two datasets. This analysis in turn leads us to investigate various simple schemes for combining natural and artificial data methods inspired from the field of domain adaptation. Our best combined system yields a significant improvement of 4.4% over a system solely trained on the available manually annotated data.

The rest of this paper is organized as follows. Section 2 summarizes previous works on implicit relation identification. In section 3, we describe the problems introduced by the use of artificial data and the methods we develop to deal with them. In section 4, we give a description of the data used, and in section 5, we detail our feature set. Our experiments are then summarized in section 6.

2 Related Work

To date, there have been only a few attempts at building full document-based discourse parsers. On the RST-DT (Carlson et al., 2001), the best performing system is (Joty et al., 2013), who report an F_1 score of 55.71 for labeled stuctures (with 23 relations). On the same corpus, (Sagae, 2009) and (Hernault et al., 2010) report F_1 scores of 44.5 and 47.3, respectively. On the PDTB, the parser of Lin et al. (2010) obtains an F_1 score of 33 (16 explicit relations, 11 implicit relations). On the ANNODIS corpus, Muller et al. (2012) reports F_1 scores of 36.1 (17 relations) and 46.8 (4 relations).

These still modest performance are due to wrong attachment decisions, as well as to errors in relation labeling. Most of these latter errors are mostly imputable to wrong classifications of implicit relations. Thus, the current best accuracy performance on explicit PDTB relations are 94.15% on 4 relations (Pitler and Nenkova, 2009), and 86.77% on 16 relations (Lin et al., 2010). By contrast, the best identification system for implicit PDTB relations obtains an accuracy of 65.4% on 4 relations in (Pitler et al., 2009), and down to 40.2% for 11 of the level 2 relations of PDTB (Lin et al., 2009). For German, Versley (2013)'s study on implicit relations reports 42.5 in F_1 for 5 relations and 18.7 for 21 relations. For French, Muller et al. (2012) report an accuracy score of 63.6% for their relation labeling system (over 17 relations), but they do not provide separate scores for explicit vs. implicit relations.

This performance drop reflects the difficulty of identifying a rhetorical relation in the absence of an explicit discourse marker. As shown by (Park and Cardie, 2012), the identification of implicit relations relies on more diverse and noisy predictors from syntax (in the form of prediction rules) and (lexical) semantics (e.g., polarity, semantic classes and fine-grained semantic tags for verbs). Unfortunately, most of the semantic resources used to derive features for English (polarity database, Inquirer tags) are not available for French. Zhou et al. (2010) try to predict the implicit connectives annotated in the PDTB as a way of predicting the relation, a method only possible with this corpus. They obtain results lower than those reported by (Park and Cardie, 2012). In another context, Sporleder (2008) shows that using WordNet is less effective than lemmatisation for capturing semantic generalization, and (Wang et al., 2010) use tree kernels in order to better capture important syntactic information. In another context, Sporleder (2008) shows that using WordNet is less effective than lemmatisation for capturing semantic generalization, and (Wang et al., 2010) use tree kernels in order to better capture important syntactic information.

Another set of studies we directly build upon explore the idea that many connectives unambiguously trigger a unique relation, thus allowing to construct massive amount of (artificially) labelled implicit examples from raw data. Marcu and Echihabi (2002) were the first to use this method: they were mainly interested in showing that a removed connective could be recovered from its linguistic context. In turn, they only tested their approach on examples that were also generated automatically, and not on manually annotated implicit examples. In this setting, they report an accuracy of 49.7 (6 classes), significantly above luck. Reusing the same approach, Sporleder and Lascarides (2008) then showed that a system trained on a large amount of artificial examples (72000 examples) performs much worse than the same system trained on a much smaller amount of natural examples ($1,051$ examples) implicit examples, with accuracies of 25.8 and 40.3, respectively.

Marcu and Echihabi's (2002) original approach was based on the idea of finding pairs of semantically related words that together trigger a relation (such as "nocturne/jour" ("nocturnal/daytime") in example 1 of *contrast*). Interestingly, Pitler et al. (2009) showed that word pairs extracted from artificial data are not helpful for implicit relation identification and, moreover, that the most informative word pairs are not semantically related. Blair-Goldensohn et al. (2007) showed that, for *cause* and *contrast* at least, results can be enhanced by improving the quality of the artificial data. Finally, Wang et al. (2012) propose a first approach that exploits both natural and artificial data. Specifically, they select the most informative training points among natural and artificial examples, both coming from the PDTB or the RST DT. They define deterministic rules for identifying so-called "typical" examples of a relation, the "seed" sets that are then expanded using a simple clustering algorithm. They report performance results well over those of (Pitler et al., 2009), but using a different evaluation protocole. [4] Also, their method is not easy to repoduce, especially for French, where we can not define the same deterministic rules as some of these depend on polarity information, for which we do not have external resources. Furthermore, their approach only extracts 1 to 5% of the data as seed examples, which would represent too few examples on our corpus. Finally, we are interested in finer-grained relations, thus more difficult to discriminate using these kind of rules.

3 Proposed Approach

Our approach builds upon and extends the method of (Marcu and Echihabi, 2002) and (Sporleder and Lascarides, 2008) by investigating different strategies for combining natural and artificial examples of implicit discourse relations. These different combination schemes are inspired from domain adaptation and are motivated by the fact that artificial and natural examples follow different probability distributions.

3.1 Distribution Differences

Most machine learning algorithms are based on the assumption that data from training and test samples are independently and identically distributed (i.e., the i.i.d. sampling assumption). Yet, it seems that the use of artificial data clearly undermines this assumption. There is indeed no guarantee that our artificial examples should follow a distribution similar to that of the manual examples. This leads to the problem of learning from non-iid data, a problem that has attracted growing attention these last years in machine learning and NLP (Sogaard, 2013), (Hand, 2006).

In this particular context, we have two sets of data with the same output space (i.e., the discourse relations), and the same kind of inputs space (i.e., spans of text). But our data samples can differ in a number of ways. Following the terminology in (Moreno-Torres et al., 2012), we may encounter all the different kinds of *shift* that can appear in a classification problem.

Prior Probability Shift This shift describes changes in the marginal distribution of the *output* (i.e., the relations). The artificial data do not have the same class distribution as the natural ones (see section 4). Neither do they have the same distribution as the natural explicit, because of the automatic extraction. This problem can be easily handled by resampling artificial data (see section 4).

[4] Wang et al. (2012) only use the first annotated relation and ignore the *Entity* relation, whereas Pitler et al. (2009) keep all the annotations and map *Entity* examples to the *Expansion* class.

Covariate Shift This shift describes changes in the marginal distribution of the *input* (i.e., the pairs of spans of text). Artifical examples are originally explicit examples minus their connective, so it is reasonnable to think that these examples will have a different distribution from the natural implicit examples. Moreover, it is possible that, by removing the connective, we have made these examples semantically unfelicitous or even ungrammatical. Segmentation is another issue, since it is automatic and based on heuristics (see section 4). For example, artificial examples can not be multi-sentential whereas it can be the case for natural ones.

Concept Shift This shift describes changes in the *joint* distribution of inputs and outputs. Consider for instance the occurrences of relations within inter- and intra-sentential contexts. The proportion of inter-sentential examples in natural and artificial datasets is the same for *contrast* (57.1%), it is similar for *result* (resp. 45.7% and 39.8%), but very different for *continuation* (resp. 70% and 96.5%) and for *explanation* (resp. 21.4% and 53.0%). Moreover, the extraction method is prone to errors, and it may be the case that we wrongly identify a word form as a discourse connective. Thus, we may produce examples annotated with a wrong relation or that do not involve any discourse relation at all. Finally, deleting a connective can make the discourse ackward or even incoherent (Asher and Lascarides, 2003). We can actually witness this with example (1). As shown by (Sporleder and Lascarides, 2008), deleting the connective can also change the inferred relation. They found examples of *explanation* in which an implicit relation becomes the only one inferable after removing the explicit marker.The deletion can also change the inferred relation (Sporleder and Lascarides, 2008). We found an even worse effect in our French corpus. In example (3), the connective *puisqu(e)* (*because*) triggers an *explanation*, thus the events are ordered following the causal law. The cause, "migrer" ("migrate"), comes before the effect, "deviennent" ("becomes"). But when we delete the connective, the order of the events seems to be reversed. Keeping the first clause as the first argument, we then obtain a *result* relation in this sentence.

(3) Les Amorrites deviennent à la période suivante de sérieux adversaires des souverains d'Ur, **puisqu**'ils commencent alors à migrer en grand nombre vers la Mésopotamie.
In the next period, Amorrites become severe opponents of the sovereigns of Ur, ***because*** *they then begin to migrate in large numbers to Mesopotamia.*

3.2 Methods Inspired by Domain Adaptation

A way to deal with all the distribution differences observed is to reframe our problem within the framework of domain adaptation. Informally, the task of domain adaptation is to port some system from one domain, the *source*, to another, the *target*. Informally, we have a distribution D_s for the source data and a distribution D_t for the target data. The goal of the classifier is to build a good approximation of D_t. If one uses data following the distribution D_s in order to build this approximation, then the performance will depend of the similarity between D_s and D_t. If these distributions are too dissimilar, the approximation will be bad and so will be the performance. It is the case in particular when the domains (e.g., text genres) are different. The goal of domain adaptation is precisely to deal with data from different distributions (Jiang, 2008), (Mansour et al., 2009). We are not exactly in the same setting, but we can regard the artificial data as the *source*, and the natural data, on which we evaluate, as the *target*.

As a first step, we decided to investigate the simplest domain adaptation methods there is, such as those described in (Daumé III, 2007). These methods either combine directly the data or the models built on each set of data. Performance of all these systems will be compared to the base systems trained on only one set of data, in section 6.

Data combination The first possibility is to combine the data. The first model is trained on all natural *and* artificial data together (UNION). This method does not allow us to control the importance of the two sets of data nor to evaluate their influence on the system. We thus refine it in two ways. First, we only add to the manual data randomly selected samples from the artificial data (ARTSUB). Alternatively, we keep all the artificial examples but reweight (or, equivalently, duplicate) the manual examples (NATW). Both these schemes allow us to avoid a massive imbalance between the two kinds of data.

Model combination The second strategy consists in combining the models. A first set of methods involve adding new features. That is, we train a model on the artificial data, then run it on the natural examples. We use these predictions as new attributes for the natural model (ADDPRED). The parameter associated to the attribute therefore measures the importance to be given to the predictions made by the model trained on artificial data. We propose a variation of this method by adding the probabilities of each prediction as supplementary attributes (ADDPROB). The intuition is that even if the classifier is wrong, it could still be consistent in its errors. Yet another model combination consists in using the parameters of the artificial model as initial values for the manual model parameters (ARTINIT). This method allows to give an initial information to the natural model rather than a random intialization. Finally, we also build a model by linearly interpolating the two basic models (LININT).

In addition to these combination schemes, we also add a method to automatically select examples among the artificial set based on the confidence of the artificial model. Its aim is to filter out noisy examples, our hypothesis being that the more confident the model, the less noisy the example.

4 Data

In this work, we choose to focus on 4 relations, *contrast*, *result*, *continuation* and *explanation*, each of which can be either explicit or implicit. These are the same as the relations used in (Sporleder and Lascarides, 2008), allowing for easy comparison across languages, with the exception of the relation *summary* which does not appear in the ANNODIS corpus. Although it is difficult to map these relations onto the relation set of the PDTB, we can say that our relations are closer to level 2 and level 3 (i.e., fine-grained) PDTB relations than level 1 (i.e., coarse-grained) ones.

4.1 Manually Annotated Data: ANNODIS

Our natural implicit examples are taken from the ANNODIS corpus, which is to date the only available French corpus annotated at the discourse level. Its annotations are based on the SDRT framework (Asher and Lascarides, 2003). It consists of 86 newspaper and Wikipedia articles. 3,339 examples have been annotated using 17 relations. In way of comparison, note that the PDTB has roughly 12 times more annotated relations than ANNODIS. Documents are segmented in Elementary Discourse Units (EDUs) which can be clauses, prepositionnal phrases and some adverbials and parentheticals if the span of text describes an event. The relations link EDUs and complex segments, adjacent or not. The connectives are not annotated, which means that the examples of implicit relations had to be extracted automatically.

The corpus has been pre-processed using the MELt tagger (Denis and Sagot, 2009) for POS-tagging, lemmatization and morphological markings. Then, the documents have been parsed using the the MST-Parser (McDonald and Pereira, 2006) trained for French by (Candito et al., 2010). In order to identify implicit examples, we used the French lexicon of connectives (LexConn) developed by Roze et al. (2012). We simply matched all possible connective forms associated with the annotated relations (discarding *à*, which is too ambiguous). We did not add constraints on the connective position, as we wanted to be sure to exclude all explicit examples, this method led us to miss a few implicit examples. Out of 1,108 examples annotated with one of the 4 relations considered, 494 were found to be implicit (see table 2).

4.2 Automatically Annotated Data

The artificial data are automatically extracted from raw data using heuristic rules. We use LexConn to mine explicit instances in the corpus Est Républicain composed of newspaper articles (9M sentences), with the same pre-processings as ANNODIS. LexConn contains 329 connectives, among them, 131 are unambiguous for our 4 relations. We grouped pragmatic relations (i.e., the relation is between speech acts) and non pragmatic relations (i.e., the relation is between facts) relations, assuming they involve the same kind of predictors, and the 3 contrastive relations, as only one type of *contrast* is annotated in ANNODIS. We did not take into account 3 connectives corresponding to unknown part-of-speech. Our first evaluation led us to delete 6 connectives, very ambiguous between discourse and non discourse readings, such as "maintenant" ("now"). We eventually settled on 122 connectives, among which 100 were seen in the corpus in a configuration matching one of our pre-defined patterns. As a comparison, (Sporleder

and Lascarides, 2008) only had 50 such connectives. We finally use 122 connectives, among which 100 were seen in a correct configuration in the corpus. As a comparison, 50 were used in (Sporleder and Lascarides, 2008).

Position	Part-of-speech	Patterns	Examples
Inter-sentential	All POS	A1. **C**(,) A2.	A1. **Malheureusement**(,) A2 A1. **Surtout**, A2.
	Adv.	A1. beg-A2(,) **C**(,) end-A2. A1. A2, **C**.	A1. beg-A2, **de plus**, end-A2. A1. beg-A2(,) **en outre**(,) end-A2. A1. A2, **remarque**.
Intra-sentential	All POS	A1, **C**(,) A2.	A1, **de plus**(,) A2. A1(,) **donc**(,) A2.
	SC and Prep.	**C** A1, A2.	**Preuve que** A1, A2. **Puisque** A1, A2.
	Adv.	A1, beg-A2(,) **C** (,) end-A2. A1, A2, **C**.	A1, beg-A2, **de plus**, end-A2. A1, beg-A2(,) **en outre**(,) A2. A1, A2, **réflexion faite**.

Table 1: Defined patterns with some examples. "A1" stands for the first argument, "A2" for the second and "**C**" stands for the connective ; "beg" and "end" stand resp. for the beginning and the end of an argument ; "(x)" indicates that "x" is not necessary, depending on the connective form. Some patterns are only possible for some sets of connectives based on their part-of-speech (Subordinating Conjunction (SC), Preposition (Prep.), Averbials (Adv.)).

The heuristic used to extract the examples has two main steps. First, we search forms used in discourse readings using patterns (see table 1) that were manually defined for each connective based on its position, its part-of-speechand the punctuation around it. Second, we identify the connectives arguments using the same information. We make the same simplifying assumptions as in the previous studies: an argument covers at most one sentence, and we have at most 2 EDUs within a sentence. As additional constraint, we also require the presence of a verb in each relation argument. When two connectives occur in the same segment, it is possible that one modifies the other. In turn, a naive extraction could produce two examples with different relations but the same arguments. To avoid the creation of spurious examples, we extract two examples in these cases only if one is inter- and the other intra-sentential according to our extraction patterns.

	Natural dataset		Artificial dataset		
Relation	Explicit	Implicit	Available	Training	Test
contrast	100	42	252 793	23 409	2 926
result	52	110	50 297	23 409	2 926
continuation	404	272	29 261	23 409	2 926
explanation	58	70	59 909	23 409	2 926
All	614	494	392 260	93 636	11 704

Table 2: Number of examples in our corpora, for the natural dataset, only the implicit examples are used.

This simple method allows to quickly generate a large amount of data. In total, we extracted $392,260$ examples (see table 2). This initial dataset was rebalanced in a way to keep the maximum number of available examples (thus dealing with the prior probability shift). We used 80% of the data as training set, and 10% the development and test set. Note that there are some important differences in the label distributions between natural and artificial data. For instance, the most represented relation in the natural data (*continuation*) is the least represented in the artificial data. This is because the connectives that trigger this relation are highly ambiguous between discourse and non-discourse readings. Finally, this method generates some noise: out of 250 random examples, we found 37 errors in span boudaries and

18 cases in which the connective form does not have a discourse reading.

5 Features

We adapted various features used in previous studies. The lack of ressources for French prevented us from using them all, especially the semantic ones. These features correspond to surface information and others more linguistic. As a comparison, (Marcu and Echihabi, 2002) only used pairs of words.

Sporleder and Lascarides (2008) used various linguistic features but no syntaxic ones. (Wang et al., 2012) used semantic, syntaxic and lexical information. We used lexico-syntaxic information. Finally, note that our goal is to evaluate the efficiency of data combinations. Thus we did not try to optimize this feature set, as it would have introduced another parameter in our model.

Indication of syntactic complexity: we compute the number of nominal, verbal, prepositional, adjectival and adverbial phrases.

Information concerning the heads of the arguments: we keep the lemma of negative element linked to the head, we also get some temporal/aspectual information (number of auxiliaries dependent of the head, tense, person, number of the auxiliaries), information about the heads dependents (if an object, a by-object or a modifier is present ; if a preposition dependent of the head, subject or object is present ; part-of-speech of the modifiers and prepositional dependents of the head, subject and object) and some morphological information (tense and person of the head if verbal, gender if non verbal, number of the head, precise part-of-speech, "VPP", and simplified,"V"). We also add features pairing the tenses for verbal heads and the heads numbers.

Position: we add a feature indicating if the example is inter or intra-sentential.

Indication of thematic continuity: we compute general lemma overlap and lemma overlap for open class words.

6 Experiments

Our main objective is to assess whether one can use the artificial data to improve the performance of a system solely based on data manually annotated only available in small amount. We therefore test the methods described in section 3.

We experimented with a maximum entropy classifier from the MegaM[5] package, in multiclass classification, with a maximum of 100 iterations. We did not try to optimize the regularization parameter which is then equal to 1.

We rebalance the corpus of manually annotated data to a maximum of 70 examples per relation.[6] We have too few annotated examples to be able to construct a separate test set sufficiently large to make statistical significance test. Thus, we decided to make a stratified nested cross-validation. It has been shown that this method provides an estimate of the error that is very close to that one could obtain on an independent evaluation set ((Varma and Simon, 2006), (Scheffer, 1999)), as it prevents us from optimizing our hyper-parameters and performing evaluation on the same data. Specifically, there are two cross-validation loops: the inner loop is used for tuning the hyper-parameters (as described in section 6.2) and the outer loop estimates the generalization error. The data are first split into N folds. We take the fold k (with $1 \leq k \leq N$) as the current evaluation set. The $N-1$ other folds are used as training data and split into M folds used for model fitting. The best model is then evaluated on the fold k. Finally, we report performance on the N folds. We used two 5-fold cross-validation in order to select and evaluate the best models for the systems described in section 3.2. We have no guarantee to select the best models at each test step, but this procedure allows to evaluate the stability of the system with respect to the hyper-parameters (i.e. the chosen values should not be too scattered), the overfitting (i.e. inner and

[5] http://www.umiacs.umd.edu/~hal/megam/version0_3/

[6] Our focus is on the methodology of data combination, so we left for future work the issue of dealing with the highly imbalanced relation distribution of the natural data. Incidentally, note that this setting prevents us from getting a system solely performing well on highly frequent relations.

outer estimations should be close) and the stability of the models (i.e. variance in the predictive capacity, between the results on the outer folds).

As in the previous studies, we report performance using micro-averaged accuracy and F_1 score per relation. In order to evaluate the statistical significance of our results, we use the Student's t-test (with p-value < 0.05) which has been proved to work with very small sample (see (de Winter, 2013)) if the effect size (computed using the Cohen coefficient) and the correlation between the sample are large enough, while, as noted in (de Winter, 2013), the Wilcoxon signed rank test (that we initially tried) could lead to overestimated p-value with such small sample. The results of the most relevant systems are presented in table 3.

	Without selection				With selection	
	NATONLY	ARTONLY	ADDPRED	ARTINIT	ADDPRED+SELEC	NATW+selec
Accuracy	37.3	23.0	39.3	40.1	**41.7***	41.3
contrast	15.0	23.2	16.0	16.9	20.8	19.2
result	47.6	15.7	50.6	45.9	51.0	48.3
continuation	28.1	32.1	31.9	34.0	31.2	32.4
explanation	47.9	22.4	46.7	52.2	53.9	53.4

Table 3: Most relevant systems, with or without selection of examples, overall accuracy and F_1 score per relation, * corresponds to a significant improvement over NATONLY.

6.1 Basic Models

In the first set of experiments, we trained two classifiers. The first one is trained on the natural implicit data (NATONLY, 252 examples), and the second one on the artificial implicit data (ARTONLY, 93,636 examples). We test both models on natural implicit data.

The overall accuracy of the NATONLY model is 37.3 with F_1 score ranging from 15.0 for *contrast* to 47.9 for *explanation*. The performance on *contrast* is fairly low, probably because this relation is the least frequent in our training set. Note that the overall accuracy obtained is quite close to the 40.3 obtained for English by (Sporleder and Lascarides, 2008).

The overall accuracy of the ARTONLY model is 47.8 when evaluated on the same type of data, that is, artificial ones (11,704 test examples), but only 23.0 when evaluated on natural data. This significant drop in performance has been observed in the previous studies on English. It can be attributed to the distribution differences described in section 3. We can observe that the use of the artificial data lowers the F_1 score for *result* and *explanation* while, for *contrast*, F_1 score is raised by about 10 points.

6.2 Models with Combinations

In this section, we present the results for the systems using both natural and artificial data. We either directly combine the data or use the data to build separate models that are then combined. Some of these models use hyper-parameters. When weighting the natural examples, we test weights $c \in [0.5, 1, 5]$ and $c \in [10; 2000]$ with an increment of 10 until 100, of 50 until 1000 and of 500 until 2000. When adding random subsets of artificial data, we add each time k times the number of natural examples artificial examples with $k \in [0.1; 600]$ with an increment of 0.1 until 1, of 10 until 100 and of 50 until 600. Finally, when taking a linear interpolation of the models, we build a new model by weighting the artificial model by $\alpha \in [0.1; 0.9]$ with increments of 0.1.

In general, we observe that most of the systems lead to similar or higher accuracy than NATONLY, but none of the improvements is statistically significant. The best system is ARTINIT (accuracy 40.1, p-value of 0.18 and a small effect size, 0.39). Two other systems get an accuracy score better than 39, that is ADDPRED (39.3) and LININT (39.3), but not significantly better than NATONLY. The system ADDPROB, similar to ADDPRED, leads to lower accuracy, showing that adding the probabilities decrease the performance. For these systems, the scores on each of the outer folds are close[7], specially for ADDPRED,

[7] ARTINIT : standard deviation (sd) = 0.074, mean = 40.1 ; ADDPRED : sd = 0.037, ADDPROB sd = 0.061, mean $\simeq$ 39

revealing a high model stability. The other systems allow to evaluate the impact of the artificial data on the final results.

The only method leading to lower results is when training on the union of the data sets (UNION), the accuracy (22.6) is similar to ARTONLY. This was expected, as the natural data are about 372 times less numerous than the artificial ones, the new model is thus more influenced by the latter. Note that Wang et al. (2012) also experiment this setting but do not observe such a gap, maybe because their artificial data are based on manually annotated explicit examples, which are likely to be less noisy.

When directly combining the data, either by adding random subsets of the artificial data (ARTSUB, accuracy 34.5) or by weighting the natural examples (NATW, accuracy 38.9), we observe, on the inner folds, an inverse trend. As expected, the accuracy increases as the influence of the artificial data decreases, that is, decreasing the coefficients for ARTSUB and increasing the weights for NATW. Observing the results in the inner folds reveals a same trend about the relative importance of the two kinds of data: natural data have to be around 2.5 times more important than the artificial ones. We also observe this effect with LININT, with the mean of the choosen α values equals to 0.3. We also note that the variance for the values of the hyper-parameter for ARTSUB is pretty high, probably caused by the randomness of the subsamples selection. It is a bit lower for NATW and LININT showing that these methods are more robust. Nevertheless, the strategy does not give an *a priori* good value for the hyper-parameter but restricts the space of values (1020 plus or minus 272 for NATW and 0.3 plus or minus 0.18 for LININT).

6.3 Models with Automatic Selection of Examples

Previous experiments showed that adding artificial data mostly improves the performance but still not significantly. We assume that a lot of the artificial data are noisy, which could hurt the systems. The method of selection of examples thus aims at eliminating potentially noisy examples. The artificial model is used on the training set, and we keep the examples that are predicted with a probability higher than a threshold $s \in [0.3; 0.85]$ with an increment of 0.1 until 0.5 and of 0.05 until 0.85. If the model is confident enough about its prediction, the example might not correspond to noise, that is, a word form that does not have a discourse readings and/or a segmentation error. We also check whether the connective is redundant. For each threshold, we rebalance the data based on the least represented relation (+SELEC systems).

The automatic selection of examples allows to improve previous results. The accuracy of the ARTONLY model moves from 23.0 to 25.0 with selection, and the system UNION move from 22.6 to 40.1 with selection.

The best results are obtained when we use artificial data to create new features but when we add only the relation predicted by the artificial model (ADDPRED+SELEC). With this system, we observe a clear tendency toward significance (accuracy 41.7 with a large effect size, 0.756, and a high correlation, 0.842). The F_1 scores for all classes are improved : 20.8 for *contrast*, 51.0 for *result*, 31.2 for *continuation* and 53.9 for *explanation*. Two other systems get an accuracy over 40: NATW+SELEC (accuracy 41.3, with a trend toward significance[8]) and UNION+SELEC (no significantly higher than NATONLY). We note that ADDPRED corresponds to the best baseline in (Daumé III and Marcu, 2006), which shows the relevance of dealing with the distributions differences in our data through domain adaptation methods.

The automatic selection step allows a more important weight on the informations provided by the artificial data. For LININT+SELEC, the best results are obtained with an almost equal influence of the two models. In the same way, the mean of the choosen values for the coefficient for NATW+SELEC is much lower, and it increases a lot for ARTSUB+SELEC allowing for larger subsamples. Even if the choosen values are widly scattered, these observations tend to prove that the selection improves the quality of our artificial corpus. Regarding the choosen values for the thresholds, the mean over all the systems is 0.7, with a variable standard deviation but always greater than 0.1. This deviation is pretty high, this hyper-parameter probably needs a better optimisation, by repeating the inner loop for example, but these experiments will allow to reduce the search space.

[8] p-value $= 0.077$, large effect size, 0.68 and high correlation, 0.67

The automatic selection of examples leads to one system, namely ADDPRED+SELEC, significantly improving the accuracy of NATONLY. This shows that the artificial data, when rightly integrated, can thus be used to improve a system identifying implicit relations, especially if their influence is low, the model is driven towards the good distribution.

6.4 Effects on the Identification of the Relations

Looking at the F_1 score per relation, we observed that these systems have dissimilar behaviors. A larger influence of the artificial model allows improvements for *contrast*: the best result for this relation is obtained when only the artificial data are used for training (at best, 28.8 F_1 score with ARTONLY+SELEC). The identification of the relation *continuation* seems to be also improved by the influence of the artificial data. We can observe it with the linear interpolation of the models: the mean of the F_1 score increases with the increasing of the α coefficient for these relations. For *continuation*, however, the best mean F_1 is obtained with $\alpha = 0.8$, this relation needs a certain degree of influence from the natural data. Some support for this proposition can be found in the fact that the best result for this relation is obtained with NATW (at best, 44.7 F_1 score). For the other relations, a large weight on the artificial data clearly decreases the F_1 score. However, the identification of *explanation* is improved when we add the predictions of the artificial model (at best, ADDPRED+SELEC, 53.9 F_1 score). Improvement is fairly low for *result* (at best, 51.0 with ADDPRED+SELEC).

The relation *contrast* might take advantage of less noisy artificial data as most of the examples are extracted using the connective *mais* (*but*) always in discourse readings. For *explanation*, predictions of the artificial model could be quiet coherent as most of the artificial examples correspond to the pragmatic relation *explanation**. Moreover, if we look at the feature distribution (850 features overall), we observe a gap of more than 30% for 2 and 5 features for *result* and *explanation* that is not observed for *contrast* and *continuation*, the relations that make the most of the artificial data.

7 Conclusion

We have presented the first system that identifies implicit discourse relations for French. This kind of relation is difficult to identify because of the lack of specific predictors. In the previous studies on English, the performance on this task are fairly low despite the use of complex features, probably because of a lack of manually annotated data. To deal with this issue, even more crucial for French, our system also resorts to additional data, automatically annotated using discourse connectives. These new data, however, do not generalize well to natural implicit data, because of distribution differences. We thus test methods inspired by domain adaptation in order to combine natural and artificial data. We add an automatic selection of examples among the artificial data to deal with noise generated by the method of automatic annotation. We manage to get significant improvement over a system solely trained using available data manually annotated by using automatic selection and the addition of features corresponding to the predictions of the artificial model.

In future work, we will explore more sophisticated methods to deal with data samples that follow different distributions. We will also explore ways to deal with imbalanced data and use our methods on all the relations annotated in our French corpus. Finally, we will test these methods on English corpora, in order to compare their efficiency with previous studies.

References

Stergos Afantenos, Nicholas Asher, Farah Benamara, Myriam Bras, Cécile Fabre, Lydia-Mai Ho-Dac, Anne Le Draoulec, Philippe Muller, Marie-Paule Péry-Woodley, Laurent Prévot, Josette Rebeyrolles, Ludovic Tanguy, Marianne Vergez-Couret, and Laure Vieu. 2012. An empirical resource for discovering cognitive principles of discourse organisation: the annodis corpus (regular paper). In *Proceedings of LREC*.

Nicholas Asher and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge University Press.

Sasha Blair-Goldensohn, Kathleen R. McKeown, and Owen C. Rambow. 2007. Building and refining rhetorical-semantic relation models. In *Proceedings of NAACL HLT*.

Marie Candito, Joakim Nivre, Pascal Denis, and Enrique Henestroza Anguiano. 2010. Benchmarking of statistical dependency parsers for french. In *Proceedings of ICCL (posters)*.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue*.

Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of ACL*.

Joost C.F. de Winter. 2013. Using the student's t-test with extremely small sample sizes. *Practical Assessment, Research & Evaluation*.

Pascal Denis and Benoît Sagot. 2009. Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art POS tagging with less human effort. In *Proceedings of PACLIC*.

Anna Gastel, Sabrina Schulze, Yannick Versley, and Erhard Hinrichs. 2011. Annotation of implicit discourse relations in the tüba-d/z treebank. *GSCL*.

David J. Hand. 2006. Classifier technology and the illusion of progress. *Statistical Science*.

Hugo Hernault, Helmut Prendinger, David A. duVerle, and Mitsuru Ishizuka. 2010. Hilda: A discourse parser using support vector machine classification. *Dialogue and Discourse*.

Jing Jiang. 2008. A literature survey on domain adaptation of statistical classifiers. Available from: http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/da_survey.html.

Shafiq R. Joty, Giuseppe Carenini, Raymond T. Ng, and Yashar Mehdad. 2013. Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In *Proceedings of ACL*.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the penn discourse treebank. In *Proceedings of EMNLP*.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2010. A PDTB-styled end-to-end discourse parser. Technical report, National University of Singapore.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. 2009. Domain adaptation: Learning bounds and algorithms. In *Proceedings of COLT*.

Daniel Marcu and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of ACL*.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*.

Jose G. Moreno-Torres, Troy Raeder, Rocío Alaiz-Rodríguez, Nitesh V. Chawla, and Francisco Herrera. 2012. A unifying view on dataset shift in classification. *Pattern Recognition*.

Joonsuk Park and Claire Cardie. 2012. Improving implicit discourse relation recognition through feature set optimization. In *Proceedings of SIGDIAL*.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP (Short Papers)*.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. In *Proceedings of ACL-IJCNLP*.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of LREC*.

Charlotte Roze, Laurence Danlos, and Philippe Muller. 2012. Lexconn: A french lexicon of discourse connectives. *Discours*.

Kenji Sagae. 2009. Analysis of discourse structure with syntactic dependencies and data-driven shift-reduce parsing. *Proceedings of IWPT*.

Tobias Scheffer. 1999. *Error Estimation and Model Selection*. Ph.D. thesis, Technischen Universitet Berlin, School of Computer Science.

Anders Sogaard. 2013. *Semi-supervised learning and domain adaptation in natural language processing*. Morgan & Claypool.

Caroline Sporleder and Alex Lascarides. 2008. Using automatically labelled examples to classify rhetorical relations, an assessment. *Natural Language Engineering*.

Caroline Sporleder. 2008. Lexical models to identify unmarked discourse relations: Does Wordnet help? *Lexical-Semantic Resources in Automated Discourse Analysis*.

Sudhir Varma and Richard Simon. 2006. Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*.

Yannick Versley. 2013. Subgraph-based classification of explicit and implicit discourse relations. In *Proceedings of IWCS*.

WenTing Wang, Jian Su, and Chew Lim Tan. 2010. Kernel based discourse relation recognition with temporal ordering information. In *Proceedings of ACL*.

Xun Wang, Sujian Li, Jiwei Li, and Wenjie Li. 2012. Implicit discourse relation recognition by selecting typical training examples. In *Proceedings of COLING (Technical Papers)*.

Zhi-Min Zhou, Yu Xu, Zheng-Yu Niu, Man Lan, Jian Su, and Chew Lim Tan. 2010. Predicting discourse connectives for implicit discourse relation recognition. In *Proceedings of COLING (Posters)*.

Reinforcement Learning of Cooperative Persuasive Dialogue Policies using Framing

Takuya Hiraoka, Graham Neubig, Sakriani Sakti, Tomoki Toda, Satoshi Nakamura
Nara Institute of Science and Technology (NAIST), Nara, Japan
{takuya-h,neubig,ssakti,tomoki,s-nakamura}@is.naist.jp

Abstract

In this paper, we apply reinforcement learning for automatically learning cooperative persuasive dialogue system policies using framing, the use of emotionally charged statements common in persuasive dialogue between humans. In order to apply reinforcement learning, we describe a method to construct user simulators and reward functions specifically tailored to persuasive dialogue based on a corpus of persuasive dialogues between human interlocutors. Then, we evaluate the learned policy and the effect of framing through experiments both with a user simulator and with real users. The experimental evaluation indicates that applying reinforcement learning is effective for construction of cooperative persuasive dialogue systems which use framing.

1 Introduction

With the basic technology supporting dialogue systems maturing, there has been more interest in recent years about dialogue systems that move beyond the traditional task-based or chatter bot frameworks. In particular there has been increasing interest in dialogue systems that engage in persuasion or negotiation (Georgila and Traum, 2011; Georgila, 2013; Paruchuri et al., 2009; Heeman, 2009; Mazzotta and de Rosis, 2006; Mazzotta et al., 2007; Nguyen et al., 2007; Guerini et al., 2003). We concern ourselves with *cooperative* persuasive dialogue systems (Hiraoka et al., 2013), which try to satisfy both the user and system goals. For these types of systems, creating a system policy that both has persuasive power and is able to ensure that the user is satisfied is the key to the system's success.

In recent years, reinforcement learning has gained much attention in the dialogue research community as an approach for automatically learning optimal dialogue policies. The most popular framework for reinforcement learning in dialogue models is based on Markov decision processes (MDP) and partially observable Markov decision processes (POMDP). In these frameworks, the system gets a reward representing the degree of success of the dialogue. Reinforcement learning enables the system to learn a policy maximizing the reward. Traditional reinforcement learning requires thousands of dialogues, which are difficult to collect with real users. Therefore, a user simulator which simulates the behavior of real users is used for generating training dialogues. Most research in reinforcement learning for dialogue system policies has been done in slot-filling dialogue, where the system elicits information required to provide appropriate services for the user (Levin et al., 2000; Williams and Young, 2007).

There is also ongoing research on applying reinforcement learning to persuasion and negotiation dialogues, which are different from slot-filling dialogue (Georgila and Traum, 2011; Georgila, 2013; Paruchuri et al., 2009; Heeman, 2009). In slot-filling dialogue, the system is required to perform the dialogue to achieve the user goal, eliciting some information from a user to provide an appropriate service. A reward corresponding to the achievement of the user's goal is given to the system. In contrast, in persuasive dialogue, the system convinces the user to take some action achieving the system goal. Thus, in this setting, reward corresponding to the achievement of both the user's and the system's goal is given to the system. The importance of each goal will vary depending on the use case of the system. For

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1706–1717, Dublin, Ireland, August 23-29 2014.

example, a selfish system could be rewarded with an emphasis on only achievement of the system goal, and a cooperative system could be rewarded with an emphasis on achievement of both of the goals. In addition, negotiation dialogue could be considered as a kind of the persuasive dialogue where the user also tries to convince the system to achieve the user's goal.

In this paper, our research purpose is learning better policies for cooperative persuasive dialogue systems using *framing*. We focus on learning a policy that tries to satisfy both the user and system goals. In particular, two elements in this work set it apart from previous works:

- We introduce framing (Irwin et al., 2013), which is known to be important for persuasion and a key concept of this paper, as a system action. Framing uses emotionally charged words to explain particular alternatives. In the context of research that applies reinforcement learning to persuasive (or negotiation) dialogue, this is the first work that considers framing as a system action.
- We use a human-to-human persuasive dialogue corpus of Hiraoka et al. (2014) to train predictive models for achievement of a human persuadee's and a human persuader's goals, and introduce these models to reward calculation to enable the system to learn a policy reflecting knowledge of human persuasion.

To achieve our research purpose, we construct a POMDP where the reward function and user simulator are learned from a corpus of human persuasive dialogue. We define system actions based on framing and general dialogue acts. In addition, the system dialogue state (namely, belief state) is defined for tracking the system's rewards. Then, we evaluate the effect of framing and learning a system policy. Experimental evaluation is done through a user simulator and real users.

2 Reinforcement learning

Reinforcement learning is a machine learning technique for learning a system policy. The policy is a mapping function from a dialogue state to a particular system action. In reinforcement learning, the policy is learned by maximizing the reward function. Reinforcement learning is often applied to models based on the framework of MDP or POMDP.

In this paper, we follow a POMDP-based approach. A POMDP is defined as a tuple $\langle S, A, P, R, O, Z, \gamma, b_0 \rangle$ where S is the set of states (representing different contexts) which the system may be in (the system's world), A is the set of actions of the system, $P : S \times A \rightarrow P(S, A)$ is the set of transition probabilities between states after taking an action, $R : S \times A \rightarrow \Re$ is the reward function, O is a set of observations that the system can receive about the world, Z is a set of observation probabilities $Z : S \times A \rightarrow Z(S, A)$, and γ a discount factor weighting longterm rewards. At any given time step i the world is in some unobserved state $s_i \in S$. Because s_i is not known exactly, we keep a distribution over states called a belief state b, thus $b(s_i)$ is the probability of being in state s_i, with initial belief state b_0. When the system performs an action $\alpha_i \in A$ based on b, following a policy $\pi : S \rightarrow A$, it receives a reward $r_i(s_i, \alpha_i) \in \Re$ and transitions to state s_{i+1} according to $P(s_{i+1}|s_i, \alpha_i) \in P$. The system then receives an observation o_{i+1} according to $P(o_{i+1}|s_{i+1}, \alpha_i)$. The quality of the policy π followed by the agent is measured by the expected future reward also called Q-function, $Q^\pi : S \times A \rightarrow \Re$.

In this framework, it is critical to be able to learn a good policy function. In order to do so, we use Neural fitted Q Iteration (Riedmiller, 2005) for learning the system policy. Neural fitted Q Iteration is an offline value-based method, and optimizes the parameters to approximate the Q-function. Neural fitted Q Iteration repeatedly performs 1) sampling training experience using a POMDP through interaction and 2) training a Q-function approximator using training experience. Neural fitted Q Iteration uses a multi-layered perceptron as the Q-function approximator. Thus, even if the Q-function is complex, Neural fitted Q Iteration can approximate the Q-function better than using a linear approximation function[1].

3 Persuasive dialogue corpus

In this section, we give a brief overview of Hiraoka et al. (2014)'s persuasive dialogue corpus between human participants that we will use to estimate the models described in later sections.

[1]In a preliminary experiment, we found that Neural fitted Q Iteration had high performance compared to using the linear approximation of the Q-function in this domain.

Table 1: The beginning of a dialogue from the corpus (translated from Japanese)

Speaker	Transcription	GPF Tag
Cust	Well, I am looking for a camera, do you have camera B?	PROPQ
Sales	Yes, we have camera B.	ANSWER
Sales	Did you already take a look at it somewhere?	PROPQ
Cust	Yes. On the Internet.	ANSWER
Sales	It is very nice. Don't you think?	PROPQ
Cust	Yes, that's right, yes.	INFORM

Table 2: Sytem and user's GPF tags

Inform	Answer	Question	PropQ
SetQ	Commisive	Directive	

Table 3: An example of positive framing

(Camera A is) able to achieve performance of comparable single-lens cameras and can fit in your pocket, this is a point.

3.1 Outline of persuasive dialogue corpus

As a typical example of persuasive dialogue, the corpus consists of dialogues between a salesperson (persuader) and customer (persuadee). The salesperson attempts to convince the customer to purchase a particular product (decision) from a number of alternatives (decision candidates). This type of dialogue is defined as "sales dialogue." More concretely, the corpus assumes a situation where the customer is in an appliance store looking for a camera, and the customer must decide which camera to purchase from 5 alternatives.

Prior to recording, the salesperson is given the description of the 5 cameras and instructed to try to convince the customer to purchase a specific camera (the persuasive target). In this corpus, the persuasive target is camera A, and this persuasive target is invariant over all subjects. The customer is also instructed to select one preferred camera from the catalog of the cameras, and choose one aspect of the camera that is particularly important in making their decision (the determinant). During recording, the customer and the salesperson converse and refer to the information in the camera catalog as support for their dialogues. The customer can close the dialogue whenever they want, and choose to buy a camera, not buy a camera, or reserve their decision for a later date.

The corpus includes a role-playing dialogue with participants consisting of 3 salespeople from 30 to 40 years of age and 19 customers from 20 to 40 years of age. All salespeople have experience working in an appliance store. The total number of dialogues is 34, and the total time is about 340 minutes. Table 1 show an example transcript of the beginning of one dialogue. Further examples are shown in Table 8 in the appendix.

3.2 Annotated dialogue acts

Each utterance is annotated with two varieties of tags, the first covering dialogue acts in general, and the rest covering framing.

As a tag set to represent traditional dialogue acts, we use the general-purpose functions (GPF) defined by the ISO international standard for dialogue act annotation (ISO24617-2, 2010). All annotated GPF tags are defined to be one of the tags in this set (Table 2).

More relevant to this work is the *framing* annotation. Framing uses emotionally charged words to explain particular alternatives. It has been suggested that humans generally evaluate decision candidates by selecting based on several determinants weighted by the user's preference, and that framing is an effective way of increasing persuasive power. This corpus focuses on negative/positive framing (Irwin et al., 2013; Mazzotta and de Rosis, 2006), with negative framing using negative words and positive framing using positive words.

In the corpus, framing is defined as a tuple $\langle a, p, r \rangle$ where a represents the target alternative, p takes value NEG if the framing is negative, and POS if the framing is positive, and r represents whether the framings contains a reference to the persuadees preferred determinant (for example, the performance or price of a camera), taking the value TRUE if contained, and FALSE if not contained. The user's preferred determinant is annotated based on the results of a questionnaire.

Table 3 shows an example of positive framing (p=POS) about the performance of Camera A (a=A). In this example, the customer answered that his preference is the price of camera, and this utterance does

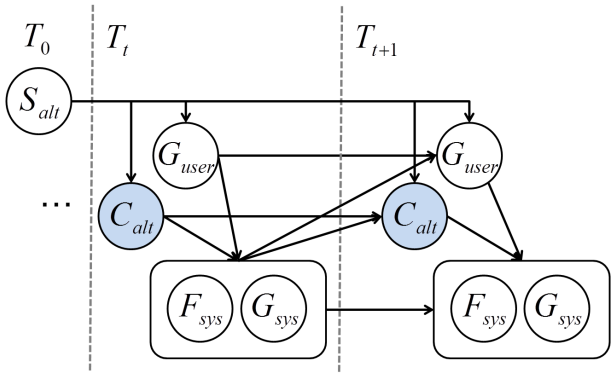


Figure 1: Dynamic Bayesian network of the user simulator. Each node represents a variable, and each edge represents a probabilistic dependency. The system cannot observe the shaded variables.

not contain any description of price. Thus, r=NO is annotated. Further examples of positive and negative framing are shown in Tables 9 and 10 in the appendix.

In this paper, we re-perform annotation of the framing tags and evaluate inter-annotator agreement, which is slightly improved from Hiraoka et al. (2014). Two annotators are given the description and examples of tags (e.g. what a positive word is), and practice with these manuscripts prior to annotation. In corpus annotation, at first, each annotator independently chooses the framing sentences. Then, framing tags are independently annotated to all utterances chosen by the two annotators. The inter-annotator agreement of framing polarity is 96.9% (kappa=0.903).

4 User simulator

In this section, we describe a statistical dialogue model for the user (customer in Section 3). This model is used to simulate the system's conversational partner in applying reinforcement learning.

The user simulator estimates two aspects of the conversation:

1. The user's general dialogue act.
2. Whether the preferred determinant has been conveyed to the user (conveyed preferred determinant; CPD).

The users' general dialogue act is represented by using GPF. For example, in Table 1, PROPQ, ANSWER, and INFORM appear as the user's dialogue act. In our research, the user simulator chooses one GPF described in Table 2 or $None$ representing no response at each turn. CPD represents that the user has been convinced that the determinant in the persuader's framing satisfies the user's preference. For example, in Table 3, the "performance" is contained in the clerk's positive framing for camera A. If the persuadee is convinced that the decision candidate satisfies his/her preference based on this framing, we say that CPD has occurred (r=YES)[2]. In our research, the user simulator models CPD for each of the 5 cameras. This information is required to calculate reward described in the following Section 5.1. Specifically, GPF and CPD are used for calculating naturalness and persuasion success, which are part of the reward function.

The user simulator is based on an order one Markov chain, and Figure 1 shows its dynamic Bayesian network. The user's GPF G^{t+1}_{user} and CPD C^{t+1}_{alt} at turn $t+1$ are calculated by the following equations.

$$P(G^{t+1}_{user}|G^t_{user}, F^t_{sys}, G^t_{sys}, S_{alt}) \quad (1)$$

$$P(C^{t+1}_{alt}|C^t_{alt}, F^t_{sys}, G^t_{sys}, S_{alt}) \quad (2)$$

G^t_{sys} represents the system GPF at time t. F^t_{sys} represents the system framing at t. These two variables correspond to system actions, and are explained in Section 5.2. G^t_{user} represents the user's GPF at t. C^t_{alt} represents the CPD at t. S_{alt} represents the users's original evaluation of the alternatives. In our

[2]Note that the persuader does not necessarily know if r=YES because the persuader is not certain of the user's preferred determinants.

research, this is the camera that the user selected as a preferred camera at the beginning of the dialogue[3]. We use the persuasive dialogue corpus described in Section 3 for training the user simulator, considering the customer in the corpus as the user and the salesperson in the corpus as the system. In addition, we use logistic regression for learning Equations (1) and (2).

5 Learning cooperative persuasion policies

Now that we have introduced the user model, we describe the system's dialogue management. In particular, we describe the reward, system action, and belief state, which are required for reinforcement learning.

5.1 Reward

We follow Hiraoka et al. (2014) in defining a reward function according to three factors: user satisfaction, system persuasion success, and naturalness. As described in Section 1, we focus on developing cooperative persuasive dialogue systems. Therefore, the system must perform dialogue to achieve both the system and user goals. In our research, we define three elements of the reward function as follows:

Satisfaction The user's goal is represented by subjective user satisfaction. The reason why we use satisfaction is that the user's goal is not necessarily clear for the system (and system creator) in persuasive dialogue. For example, some users may want the system to recommend appropriate alternatives, while some users may want the system not to recommend, but only give information upon the user's request. As the goal is different for each user, we use abstract satisfaction as a measure, and leave it to each user how to evaluate achievement of the goal.

Persuasive success The system goal is represented by persuasion success. Persuasion success represents whether the persuadee finally chooses the persuasive target (in this paper, camera A) at the end of the dialogue. Persuasion success takes the value SUCCESS when the customer decides to purchase the persuasive target at the end of dialogue, and FAILURE otherwise.

Naturalness In addition, we use naturalness as one of the rewards. This factor is known to enhance the learned policy performance for real users (Meguro et al., 2011).

The reward at each turn t is calculated with the following equation[4].

$$r_t = (Sat^t_{user} + PS^t_{sys} + N^t)/3 \quad (3)$$

Sat^t_{user} represents a 5 level score of the user's subjective satisfaction (1: Not satisfied, 3: Neutral, 5: Satisfied) at turn t scaled into the range between 0 and 1. PS^t_{sys} represents persuasion success (1: SUCCESS, 0: FAILURE) at turn t. N_t represents bi-gram likelihood of the dialogue between system and user at turn t as follows.

$$N_t = P(F^t_{sys}, G^t_{sys}, G^t_{user} | F^{t-1}_{sys}, G^{t-1}_{sys}, G^{t-1}_{user}) \quad (4)$$

In our research, Sat and PS are calculated with a predictive model constructed from the human persuasion dialogue corpus described in Section 3. In constructing these predictive models, the persuasion results (i.e. persuasion success and persuadee's satisfaction) at the end of dialogue are given as the supervisory signal, and the dialogue features in Table 4 are given as the input. In the reward calculation, the dialogue features used by the predictive model are calculated by information generated from the dialogue of the user simulator and the system. Table 4 shows all features used for reward calculation at each turn[5]. Note that, for the calculating TOTAL TIME, average speaking time corresponding to speakers and dialogue acts is added at each turn.

[3]Preliminary experiments indicated that the user behaved differently depending on the first selection of the camera, thus we introduce this variable to the user simulator.

[4]We also optimized the policy in the case where the reward (Equation (3)) is given only when dialogue is closed. However, the convergence of the learning was much longer, and the performance was relatively bad.

[5]Originally, there are more dialogue features for the predictive model. However as in previous research, we choose significant dialogue features by step-wise feature selection (Terrell and Bilge, 2012).

Table 4: Features for calculating reward. These features are also used as the system belief state.

Sat_{user}	Frequency of system commisive
	Frequency of system question
PS_{sys}	Total time
	C_{alt} (for each 6 cameras)
	S_{alt} (for each 6 cameras)
N	System and user current GPF
	System and user previous GPF
	System framing

Table 5: System framing. Pos represents positive framing and Neg represents negative framing. A, B, C, D, E represent camera names.

Pos A	Pos B	Pos C	Pos D	Pos E	None
Neg A	Neg B	Neg C	Neg D	Neg E	

Table 6: System action.

<None, ReleaseTurn>	<None, CloseDialogue>
<Pos A, Inform>	<Pos A, Answer>
<Neg A, Inform>	<Pos B, Inform>
<Pos B, Answer>	<Pos E, Inform>
<None, Inform>	<None, Answer>
<None, Question>	<None, Commissive>
<None, Directive>	

5.2 Action

The system's action $\langle F_{sys}, G_{sys} \rangle$ is a framing/GPF pair. These pairs represent the dialogue act of the salesperson, and are required for reward calculation (Section 5.1). There are 11 types of framing (Table 5), and 9 types of GPF which are expanded by adding RELEASETURN and CLOSEDIALOGUE to the original GPF sets (Table 2). The number of all possible GPF/framing pairs is 99, and some pairs have not appeared in the original corpus. Therefore, we reduce the number of actions by filtering. We construct a unigram model of the salesperson's dialogue acts $P(F_{sales}, G_{sales})$ from the original corpus, then exclude pairs for which the likelihood is below 0.005[6]. As a result, the 13 pairs shown in Table 6 remained[7]. We use these pairs as the system actions.

5.3 Belief state

The current system belief state is represented by the features used for reward calculation (Table 4) and the reward calculated at previous turn. Namely, the features for the reward calculation and calculated reward are also used as the next input of the system policy. Note that the system cannot directly observe C_{alt}, thus the system estimates it through the dialogue by using the following equation.

$$P(\hat{C_{alt}^{t+1}} | \hat{C_{alt}^{t}}, F_{sys}^{t}, G_{sys}^{t}, S_{alt}) \tag{5}$$

where $\hat{C_{alt}^{t+1}}$ represents the estimated CPD at $t+1$. $\hat{C_{alt}^{t}}$ represents the estimated CPD at t. The other variables are the same as those in Equation (2). In contrast, we assume that the system can observe G_{user} and S_{alt}. G_{user} is not usually observable because traditional dialogue systems have automatic speech recognition/Spoken language understanding errors. However, in this work, we use Wizard of Oz in place of automatic speech recognition/Spoken language understanding (Section 6.2). Thus, we can ignore these factors[8].

6 Experimental evaluation

In this section, we describe the evaluation of the proposed method for learning cooperative persuasive dialogue policies. Especially, we focus on examining how the learned policy with framing is effective for persuasive dialogue. The evaluation is done both using a user simulator and real users.

[6]We chose this threshold by trying values from 0.001 to 0.01 with incrementation of 0.001. We select the threshold that resulted in the number of actions closest to previous work (Georgila, 2013).

[7]Cameras C and D are not popular, and don't appear frequently in the human persuasive dialogue corpus, and are therefore excluded in filtering.

[8]In addition to this reason, the G_{user} is not so essential to our research (GPF is general dialogue act), and we want to focus the CPD. This is the other reason that we assume that G_{user} is observable.

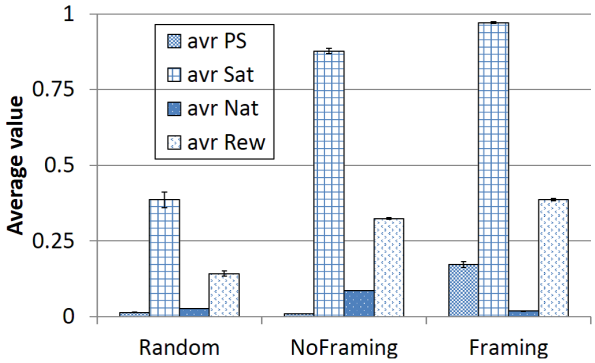


Figure 2: Average reward of each system. Error bars represents 95% confidence intervals. Rew represents the reward, Sat represents the user satisfaction, PS represents persuasion success, and Nat represents naturalness.

6.1 Policy learning and evaluation using the user simulator

For evaluating the effectiveness of framing and learning the policy through the user simulator, we prepare the following 3 policies.

Random A baseline where the action is randomly output from all possible actions.

NoFraming A baseline where the action is output based on the policy which is learned using only GPFs. For constructing the actions, we remove actions whose framing is not $None$ from the actions described in Section 5.2. The policy is a greedy policy, and selects the action with the highest Q-value.

Framing The proposed method where the action is output based on the policy learned with all actions described in Section 5.2 including framing. The policy is also a greedy policy.

For learning the policy, we use Neural fitted Q Iteration (Section 2). For applying Neural fitted Q Iteration, we use the Pybrain library (Schaul et al., 2010). We set the discount factor γ of learning to 0.9, and the number of nodes in the hidden layer of the neural network for approximating the Q-function to the sum of number of belief states and actions (i.e. Framing: 53, NoFraming: 47). The policy in learning is the ϵ-greedy policy ($\epsilon = 0.3$). These conditions follow the default Pybrain settings. We consider 50 dialogues as one epoch, and update the parameters of the neural network at each epoch. Learning is finished when number of epochs reaches 200 (10000 dialogues), and the policy with the highest average reward is used for evaluation.

We evaluate the system on the basis of average reward per dialogue with the user simulator. For calculating average reward, 1000 dialogues are performed with each policy.

Experimental results (Figure 2) indicate that 1) performance is greatly improved by learning and 2) framing is somewhat effective for the user simulator. Learned policies (Framing, NoFraming) get a higher reward than Random. Particularly, both of the learned policies better achieve user satisfaction than Random. On the other hand, only Framing is able to achieve better persuasion success than Random. This result indicates that framing is effective for persuasive success. In contrast, naturalness of Framing is not improved from Random. One of the reasons for this is that variance of Nat is smaller than those of the other factors, and the optimization algorithm favored the other two factors which had a higher variance.

6.2 Real user evaluation based on Wizard of Oz

To test whether the gains shown on the user simulator will carry over to an actual dialogue scenario, we perform an experiment with real human users. In addition to the policies described in Section 6.1, we add the following policy.

Human An oracle where the action is output based on human selection. In this research, the first author (who has no formal sales experience, but experience of about 1 year in analysis of camera sales dialogue) selects the action.

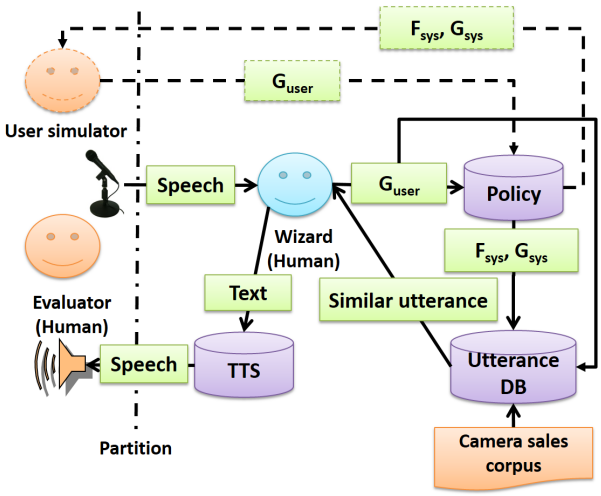


Figure 3: The experimental environment based on Wizard of Oz. The rectangle represents information, and the cylinder represents a system module. The information flow (dashed line) in the experiment through the user simulator is also shown for comparison.

Experimental evaluation is conducted, based on the Wizard of Oz framework. In the experiment, the wizard plays the salesperson, and the evaluator plays the customer. Dialogue is performed between the wizard and the evaluator. The wizard and evaluator are divided by a partition, and the evaluator cannot see or detect what the wizard is doing. The evaluator selects his/her preferred camera from the catalog before starting evaluation. Then, the evaluator starts the dialogue with the wizard who is obeying one of the policies (Figure 3). In particular, dialogue between wizard and evaluator proceeds based on the following steps.

1. The evaluator talks to the wizard using the mic. In this step, the evaluators can close the dialogue if they want.
2. The wizard listens to the evaluator's utterance, translating the utterance into the appropriate G_{user}. Then, the wizard inputs G_{user} to the policy module.
3. The policy module decides action sequences (F_{sys}, G_{sys}) based on G_{user}, then outputs the action to the utterance database module. This module is constructed from the camera sales corpus (Section 3).
4. The utterance database module searches for similar sentences that match the history of input actions and G_{user} so far, then outputs the top 6 similar utterances to the wizard.
5. The wizard generates the system utterance (Text) using the retrieved sentences. The wizard selects one sentence which best matches the context[9]. If the wizard determines the sentence is hard to understand, the wizard can correct the sentence to be more natural.
6. The wizard inputs the system utterance to text-to-speech, then waits for the next evaluator utterance (back to step 1).

Finally, the evaluator answers the following questionnaire for calculating the evaluation measures in Section 5.1.

Satisfaction The evaluator's subjective satisfaction defined as a 5 level score of customer satisfaction (1: Not satisfied, 3: Neutral, 5: Satisfied).

Final decision The camera that the customer finally wants to buy.

We use SofTalk (cncc, 2010) as text-to-speech software.

Evaluation criteria are basically same to those of previous section (described in Section 5.1). Note that in the previous section, Sat_{user} and PS_{sys} are estimated from the simulated dialogue. In contrast to the previous section, Sat_{user} and PS_{sys} are calculated from the result of the real user's questionnaire

[9]Note that the wizard is not allowed to create the utterance with complete freedom, and selects an utterance from the utterance database even when Human policy is used.

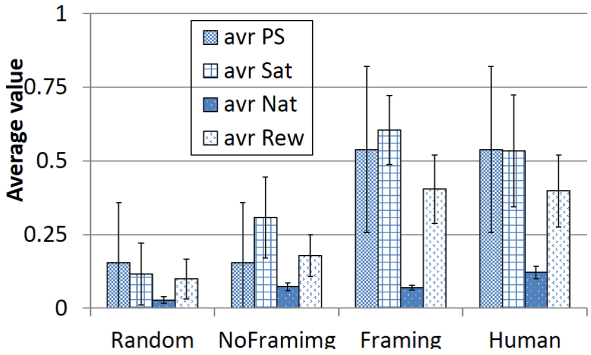


Figure 4: Evaluation results for real users. Error bars represent 95% confidence intervals. Rew represents the reward, Sat represents the user satisfaction, PS represents persuasion success, and Nat represents naturalness.

Table 7: Part of a dialogue between Framing and an evaluator (translated from Japanese)

Speaker	Transcription	Fra	GPF
Wiz	Which pictures do you want to take? Far or near?	None	QUESTION
Wiz	Camera B has 20x zoom, and this is good.	Pos B	ANSWER
Wiz	How about it?		RELEASET
Eva	I think B sounds good.		ANSWER
Wiz	Yes, B is popular with zoom,	Pos B	INFORM
Wiz	But, A has extremely good performance. Camera A has almost the same parts as a single lens camera, and is more reasonably priced than a single lens-camera.	Pos A	ANSWER
Wiz	How about it?		RELEASET

(described in the previous paragraph)[10] based on the definition of Sat_{user} and Sat_{user} in Section 6.1. The naturalness is automatically calculated by the system, in the same manner as described in the previous section. Finally, reward is calculated considering Sat_{user}, PS_{sys} and naturalness according to Equation 3.

Participants consist of 13 evaluators (3 female, 10 male) and one wizard. Evaluators perform one dialogue with the wizard obeying each policy (a total of 4 dialogues) in random order.

Experimental results (Figure 4) indicate that framing is effective in persuasive dialogues with real users, and that the reward of Framing is higher than NoFraming and Random, and almost equal to Human. In addition, the score of NoFraming is almost equal to Random. This indicates that despite the fact that it performed relatively well in the simulation experiment, NoFraming is not an effective policy for real users. In addition, the score of NoFraming is lower than the score given by the user simulator. In particular, persuasion success is drastically decreased. This indicates that framing is important for persuasion.

We can see that some features in human persuasive dialogue appear in the dialogue between users and the wizard using the Framing policy. An example of a typical dialogue of Framing is shown in Table 7. The first feature is that the system also recommends camera B when the system does positive framing of camera A, which is the persuasive target. This feature was found by Hiraoka et al. (2014) to be an indicator of persuasion success in the camera sales corpus. The second feature is that the system asks the user about the user's profile at the first stage of the dialogue. This feature is often found when user satisfaction is high. The second feature also appeared in the dialogue with NoFraming. However, NoFraming does not use framing, and asks the user to make a decision (DIRECTIVE). An example utterance from the DIRECTIVE class is "Please, decide (which camera you want to buy) after seeing the catalog".

Considering the evaluation result of the previous section, we can see that Sat and PS differ between the user simulator and the real users ($p < .05$). While the general trend of showing improvements for

[10]Note that, though systems estimate the satisfaction and evaluator's decision at each turn for the belief state, the human evaluator answers the questionnaire only when the dialogue is closed.

satisfaction and persuasive success is identical in Figures 2 and 4, the systems are given excessively high Sat in simulation. In addition, systems (especially Framing) are given underestimated PS in simulation. One of the reasons for this is that the property of dialogue features for the predictive model for reward differs from previous research (Hiraoka et al., 2014). In this paper, dialogue features for the predictive model are calculated at each turn. In addition, persuasion success and user satisfaction are successively calculated at each turn. In contrast, in previous research, the predictive model was constructed with dialogue features calculated at end of the dialogue. Therefore, it is not guaranteed that the predictive model estimates appropriate persuasion success and user satisfaction at each turn. Another reason is that the simulator is not sufficiently accurate to use for reflecting real user's behavior. Compared to other works (Meguro et al., 2010; Misu et al., 2012), we are using a relatively small sized corpus for training the user simulator. Therefore, the user simulator cannot be trained to accurately imitate real user behavior. Improving the user simulator is an important challenge for future work.

7 Related work

There are a number of related works that apply reinforcement learning to persuasion and negotiation dialogue. Georgila and Traum (2011) apply reinforcement learning to negotiation dialogue using user simulators divided into three types representing individualist, collectivist, and altruist. Dialogue between a florist and a grocer are assumed as an example of negotiation dialogue. In addition, Georgila (2013) also applies reinforcement learning to two-issue negotiation dialogue where participants have a party, and decide both the date and food type. A handcrafted user simulator is used for learning the policy of each participant. Heeman (2009) models negotiation dialogue, assuming a furniture layout task, and Paruchuri et al. (2009) model negotiation dialogue, assuming the dialogue between a seller and buyer.

Our research differs from these in three major ways. The first is that we use framing, positive or negative statements about the particular item, which is known to be important for persuasion (Irwin et al., 2013). By considering framing, the system has the potential to be more persuasive. While there is one previous example of persuasive dialogue using framing (Mazzotta et al., 2007), this system does not use an automatically learned policy, relying on handcrafted rules. In contrast, in our research, we apply reinforcement learning to learn the system policy automatically.

In addition, in these previous works, rewards and belief states are defined with heuristics. In contrast, in our research, reward is defined on the basis of knowledge of human persuasive dialogue. In particular, we calculate the reward and belief state using the predictive model of Hiraoka et al. (2014) for estimating persuasion success and user satisfaction using dialogue features. In the real world, it is unclear what factors are important for achieving the dialogue goal in many persuasive situations. By considering these predictions as knowledge of human persuasion, the system can identify the important factors in human persuasion and can track the achievement of the goal based on these.

Finally, these works do not evaluate the learned policy, or evaluate only in simulation. In contrast, we evaluate the learned policy with real users.

8 Conclusion

We apply reinforcement learning for learning cooperative persuasive dialogue system policies using framing. In order to apply reinforcement learning, a user simulator and reward function is constructed based on a human persuasive dialogue corpus. Then, we evaluate the learned policy and effect of framing using a user simulator and real users. Experimental evaluation indicates that applying reinforcement learning is effective for construction of cooperative persuasive dialogue systems that use framing.

In the future, we plan to construct a fully automatic persuasive dialogue system using framing. In this research, automatic speech recognition, spoken language understanding and natural language generation are performed by a human Wizard. We plan to implement these modules and evaluate system performance. In addition, in this research, corpus collection and evaluation are done in a role-playing situation. Therefore, we plan to evaluate the system policies in a more realistic situation. We also plan to consider non-verbal information (Nouri et al., 2013) for estimating persuasive success and user satisfaction.

References

cncc. 2010. SofTalk. http://www35.atwiki.jp/softalk/.

Kallirroi Georgila and David Traum. 2011. Reinforcement learning of argumentation dialogue policies in negotiation. *Proceedings of INTERSPEEECH*.

Kallirroi Georgila. 2013. Reinforcement learning of two-issue negotiation dialogue policies. *Proceedings of the SIGDIAL*.

Marco Guerini, Oliviero Stock, and Massimo Zancanaro. 2003. Persuasion model for intelligent interfaces. *Proceedings of the IJCAI Workshop on Computational Models of Natural Argument*.

Peter A. Heeman. 2009. Representing the reinforcement learning state in a negotiation dialogue. *Proceedings of ASRU*.

Takuya Hiraoka, Yuki Yamauchi, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2013. Dialogue management for leading the conversation in persuasive dialogue systems. *Proceedings of ASRU*.

Takuya Hiraoka, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Construction and analysis of a persuasive dialogue corpus. *Proceedings of IWSDS*.

Levin Irwin, Sandra L. Schneider, and Gary J. Gaeth. 2013. All frames are not created equal: A typology and critical analysis of framing effects. *Organizational behavior and human decision processes 76.2*.

ISO24617-2, 2010. *Language resource management-Semantic annotation frame work (SemAF), Part2: Dialogue acts. ISO*.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. 2000. A stochastic model of human-machine interaction for learning dialog strategies. *Proceedings of ICASSP*.

Irene Mazzotta and Fiorella de Rosis. 2006. Artifices for persuading to improve eating habits. *AAAI Spring Symposium: Argumentation for Consumers of Healthcare*.

Irene Mazzotta, Fiorella de Rosis, and Valeria Carofiglio. 2007. PORTIA: a user-adapted persuasion system in the healthy-eating domain. *Intelligent Systems*.

Toyomi Meguro, Ryuichiro Higashinaka, Yasuhiro Minami, and Kohji Dohsaka. 2010. Controlling listening-oriented dialogue using partially observable Markov decision processes. *Proceedings of COLING*.

Toyomi Meguro, Yasuhiro Minami, Ryuichiro Higashinaka, and Kohji Dohsaka. 2011. Wizard of oz evaluation of listening-oriented dialogue control using pomdp. *Proceedings of ASRU*.

Teruhisa Misu, Kallirroi Georgila, Anton Leuski, and David Traum. 2012. Reinforcement learning of question-answering dialogue policies for virtual museum guides. *Proceedings of the 13th Annual Meeting of SigDial*.

Hien Nguyen, Judith Masthoff, and Pete Edwards. 2007. Persuasive effects of embodied conversational agent teams. *Proceedings of HCI*.

Elnaz Nouri, Sunghyun Park, Stefan Scherer, Jonathan Gratch, Peter Carnevale, Louis-Philippe Morency, and David Traum. 2013. Prediction of strategy and outcome as negotiation unfolds by using basic verbal and behavioral features. *Proceedings of INTERSPEECH*.

Praveen Paruchuri, Nilanjan Chakraborty, Roie Zivan, Katia Sycara, Miroslav Dudik, and Geoff Gordon. 2009. POMDP based negotiation modeling. *Proceedings of the first MICON*.

Martin Riedmiller. 2005. Neural fitted Q iteration - first experiences with a data efficient neural reinforcement learning method. *Machine Learning: ECML*.

Tom Schaul, Justin Bayer, Daan Wierstra, Yi Sun, Martin Felder, Frank Sehnke, Thomas Rückstieß, and Jürgen Schmidhuber. 2010. Pybrain. *The Journal of Machine Learning Research*.

Allison Terrell and Mutlu Bilge. 2012. A regression-based approach to modeling addressee backchannels. *Proceedings of the 13th Annual Meeting of SIGDIAL*.

Jason D. Williams and Steve Young. 2007. Scaling POMDPs for spoken dialog management. *IEEE Transactions on Audio, Speech, and Language Processing*.

Appendix

Table 8: The summary of one dialogue in the corpus (translated from Japanese)

Speaker	Transcription	GPF Tag
Customer	Hello.	INFORM
Customer	I'm looking for a camera for traveling. Do you have any recommendations?	PROPQ
Clerk	What kind of pictures do you want to take?	SETQ
Customer	Well, I'm the member of a tennis club, and want to take a picture of landscapes or tennis.	ANSWER
Clerk	O.K. You want the camera which can take both far and near. Don't you?	PROPQ
Clerk	Well, have you used a camera before?	PROPQ
Customer	I have used a digital camera. But the camera was cheap and low resolution.	ANSWER
Clerk	I see. I see. Camera A is a high resolution camera. A has extremely good resolution compared with other cameras. Although this camera does not have a strong zoom, its sensor is is almost the same as a single-lens camera.	INFORM
Customer	I see.	INFORM
Clerk	For a single lens camera, buying only the lens can cost 100 thousand yen. Compared to this, this camera is a bargain.	INFORM
Customer	Ah, I see.	INFORM
Customer	But, it's a little expensive. right?	PROPQ
Customer	Well, I think, camera B is good at price.	INFORM
Clerk	Hahaha, yes, camera B is reasonably priced.	ANSWER
Clerk	But its performance is low compared with camera A.	INFORM
Customer	If I use the two cameras will I be able to tell the difference?	PROPQ
Clerk	Once you compare the pictures taken by these cameras, you will understand the difference immediately. The picture itself is very high quality. But, camera B and E are lower resolution, and the picture is a little bit lower quality.	ANSWER
Customer	Is there also difference in normal size pictures?	PROPQ
Clerk	Yes, whether the picture is small or large, there is a difference	ANSWER
Customer	Considering A has single-lens level performance, it is surely reasonable.	INFORM
Clerk	I think so too.	INFORM
Clerk	The general price of a single-lens is about 100 or 200 thousand yen. Considering these prices, camera A is a good choice.	INFORM
Customer	Certainly, I'm interested in this camera.	INFORM
Clerk	Considering its performance, it is a bargain.	INFORM
Customer	I think I'll go home, compare the pictures, and think a little more.	COMMISIVE
Clerk	I see. Thank you.	DIRECTIVE

Table 9: Example positive framing of a salesperson's utterance $\langle a_i = \mathrm{B}, p_i = \textsc{pos}, r_i = \textsc{yes}\rangle$. In this example, the customer has indicated price as the preferred determinant.

Hahaha, yes, camera B is reasonably priced.

Table 10: Example negative framing of a salesperson's utterance $\langle a_i = \mathrm{B}, p_i = \textsc{neg}, r_i = \textsc{no}\rangle$. In this example, the customer has indicated price as the preferred determinant.

But, considering the long term usage, you might care about picture quality. You might change your mind if you only buy a small camera (Camera B).

Towards multimodal modeling of physicians' diagnostic confidence and self-awareness using medical narratives

Joseph Bullard[†] **Cecilia Ovesdotter Alm**[‡]
Qi Yu[†] **Pengcheng Shi**[†] **Anne Haake**[†]
[†]College of Computing and Information Sciences
[‡]College of Liberal Arts
Rochester Institute of Technology
jtb4478@cs.rit.edu
coagla|qi.yu|spcast|arhics@rit.edu

Abstract

Misdiagnosis is a problem in the medical field, often related to physicians' cognitive errors. Overconfidence is considered a major cause of such errors. Intelligent diagnostic support systems could benefit from understanding how aware physicians are of their performance when they estimate their confidence in a diagnosis (i.e. a physician's *diagnostic self-awareness*). Shedding light on the cognitive processes related to such awareness could also help improve medical education. We use a multimodal dataset of medical narratives to computationally model diagnostic confidence and self-awareness based on physicians' linguistic and eye movement behaviors. Dermatologists viewed images of cutaneous conditions, providing a description, diagnosis, and certainty level for each image case, while their speech and eye movements were recorded. We define both a generalized and a personalized approach to binning confidence levels, used in classification experiments. We also introduce truly multimodal features, which focus on combining linguistic and eye movement data into multimodal attributes. Results indicate that combinations of multiple modalities can outperform their constituent modalities in isolation for these problems.

1 Introduction

Misdiagnosis in the medical field is estimated to be as high as 10%-15% (Berner and Graber, 2008; Croskerry, 2009). Such errors can result in incorrect or delayed treatment, causing patients to experience additional suffering. Graber et al. (2002) describe three types of diagnostic errors: *no-fault* errors, resulting from atypical disease presentation or limitations of medical knowledge; *system* errors, resulting from problems with the health care system; and *cognitive* errors, resulting from biases or faulty interpretation on the part of a physician. Cognitive errors in particular have potential for substantial reduction through education and training aimed at developing clinicians' metacognitive skills. Understanding the cognitive processes of physicians during diagnosis is also of critical importance for building human-centered diagnostic support systems, which could help detect and flag problematic diagnostic self-awareness cases. Examples of cognitive errors include settling on a final diagnosis too early, without ever considering the correct diagnosis (Berner and Graber, 2008), or confirmation bias, in which only evidence to confirm a diagnostic hypothesis is considered (Croskerry, 2003). Overconfidence is generally thought to be a major cause of such errors (Berner and Graber, 2008; Croskerry, 2008). For example, an overconfident physician may not question her original thoughts or explore alternative diagnoses until later in the treatment process. In general, overconfidence may be a systemic problem, reinforced by patients' preferences for confident doctors, and by a professional environment that favors decisive actions (Katz, 1984). Similarly, underconfidence can erode patients' trust in their providers. In this study, we view the interplay between confidence[1] and correctness as a two-dimensional problem (see Figure 1). Ideally, physicians would have high confidence when correct and low confidence when incorrect, indicated by the upper-left and lower-right quadrants in Figure 1.

[1]For consistency, this paper uses the term *confidence*, treated as interchangeable with *certainty* and similar synonymous expressions which may have been used by clinicians in the medical narratives, such as *sure*, *certain*, *confident*, etc.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1718–1727, Dublin, Ireland, August 23-29 2014.

	Correct	Incorrect
Confident	Appropriate Confidence	Overconfidence
Not confident	Underconfidence	Appropriate Confidence

Figure 1: Two-dimensional view of the confidence and correctness relationship as it relates to *diagnostic self-awareness*. A similar conceptual model is presented by Pon-Barry and Shieber (2011). Ideally, physicians should have high confidence when they are correct and low confidence when incorrect.

Contribution Diagnostic self-awareness is an important phenomenon with implications for clinical training and practice, yet has received little focus from a computational perspective. We report on computational modeling for predicting the confidence and correctness interplay in diagnosis using features of physicians' speech, eye movements, and combinations thereof, as dermatologists performed medical image inspection tasks while narrating their diagnostic thought process. In dermatology, visual expertise and clinical knowledge are both important. A motivation behind our multimodal approach is that medical image inspection relies on both the physician's visual perceptual expertise and conceptual knowledge base, each of which can be regarded as expressed by eye movement behavior and linguistic behavior, respectively. We aim to apply this decision modeling to intelligent diagnostic support and clinical tutoring systems. Here we solve a foundational problem by successfully modeling the complex relationship between physicians' confidence in and correctness of their diagnoses. We also make contributions in multimodal and linguistic feature analysis: carefully assessing feature modalities that represent physicians' behaviors, and introducing a novel multimodal feature type that focuses on fusing eye movement and verbal data.

2 Previous Work

Although there are many causes of diagnostic errors (Graber et al., 2005), those resulting from cognitive errors may be the most challenging to reduce (Croskerry, 2003; Graber et al., 2002), while their reduction provides high impact. Examples of such errors include flawed perception, biased heuristics, and settling on a final diagnosis too early (Graber et al., 2002), all of which can be caused by overconfidence (Berner and Graber, 2008; Croskerry, 2008). Underconfidence may also be a problem if it prevents a physician from pursuing a correct diagnosis (Friedman et al., 2005).

There is evidence for links between speech and confidence in terms of prosodic features, such as pitch and loudness (Scherer et al., 1973; Pon-Barry and Shieber, 2011; Kimble and Seidel, 1991), as well as other characteristics of spoken language, such as speech disfluencies (Womack et al., 2012) and hedges (Smith and Clark, 1993). Prosodic features have been identified and successfully used in intelligent tutoring systems (Liscombe et al., 2005), where a student's confidence (or lack thereof) can play a key role in effective system response. In medical diagnosis, prosodic and lexical features have been useful indicators of physicians' confidence and diagnostic correctness, individually (Womack et al., 2013; McCoy et al., 2012). Other potentially useful information may be evident in speech as well. In a study by Womack et al. (2012) on a similar dataset, the authors found a relationship between speech characteristics and physician experience: attending (experienced) physicians used more filled pauses and spoke more than resident (in-training) physicians. Additionally, verbal features may expose differences in diagnostic reasoning that may be useful predictors of confidence. Rogers (1996) analyzed a dataset of spoken chest X-ray examinations by radiologists, remarking that reasoning styles influence physicians' expectations, and confirmations or contradictions of those expectations can affect their self-reported confidence levels.

Most relevant literature focuses on linguistic features. Language, as the primary form of human expression, is certainly critical. However, analyzing meaning may require going beyond linguistic inference, depending on the context or application. Previous studies have successfully incorporated multiple expressive modalities when examining linguistic and cognitive processes, such as facial expressions for video sentiment analysis (Pérez-Rosas et al., 2013) and pointing gestures for referring actions (Gatt and Paggio, 2013). In such studies, the additional modalities were carefully chosen based on the nature of the performed tasks. Here, we deal with experts (dermatologists) inspecting images (skin conditions) for diagnostic purposes, a task that heavily involves their use of visual perceptual expertise, in addition to conceptual domain knowledge. For this reason, we incorporate features of their eye movements in our study. There is evidence for ties between perceptual expertise and eye movements during image inspection tasks (Li et al., 2012b), and we explore if such ties may also relate to a physician's confidence and diagnostic self-awareness.

Integrating different expressive modalities is challenging. Previous work involving multimodality has predominantly treated each in isolation. We further address this challenge by identifying and exploring truly *multimodal* features that focus on combining verbal and eye movement data into complex multimodal attributes, as it seems reasonable that the two modalities together could be more informative if linked, and that such complex features represent a natural interactive extension of multimodal semantics. Evidence for ties between speech and eye movements specifically was found by Li et al. (2012a), in which sequences of fixations and saccadic eye movements were identified to predominantly align with particular conceptual units of thought (e.g. primary lesion type) expressed verbally in medical narratives.

3 Data Description and Analysis

This study takes advantage of a dataset previously reported on by Womack et al. (2013), which is briefly described here for clarity, as Womack et al.'s work ignored the eye movement data. A group of 29 dermatologists (11 attending physicians, 18 residents) were each shown a series of 30 images of dermatological conditions in random order and asked to narrate their diagnosis of each condition. They were asked to provide a description of the case, a list of differential diagnoses to consider, a final diagnosis, and their certainty of their final diagnosis, as a percentage. The physicians' verbal descriptions were recorded as audio and later manually transcribed in detail, including pauses, disfluencies, and other speech phenomena.[2] During this process, the physicians' eye movements were also tracked. Each image was displayed on a 22" LCD monitor (1650x1050 pixels) with an attached 250Hz SensoMotoric Instruments RED remote eye-tracker while IViewX software was recording the eye movements.

In this study, the time-aligned pair of verbal description and eye movements for one physician viewing one image is henceforth called a *narrative*. Figure 2a shows an example of a verbal description for one narrative and Figure 2b shows a visualization of the corresponding eye movements. The correct diagnoses for all images were known for the experiment and each narrative was assigned a binary label of *correct* or *incorrect*.[3] For the purposes of this multimodal study, 238 of the 870 narratives were excluded due to technical issues that had occurred with the eye tracking or audio capture equipment, or because the physicians had provided no confidence values for their diagnoses. The remaining 632 narratives were used for the analysis and experimentation reported on in this paper.

3.1 Case Studies towards Understanding Physicians' Confidence and Correctness

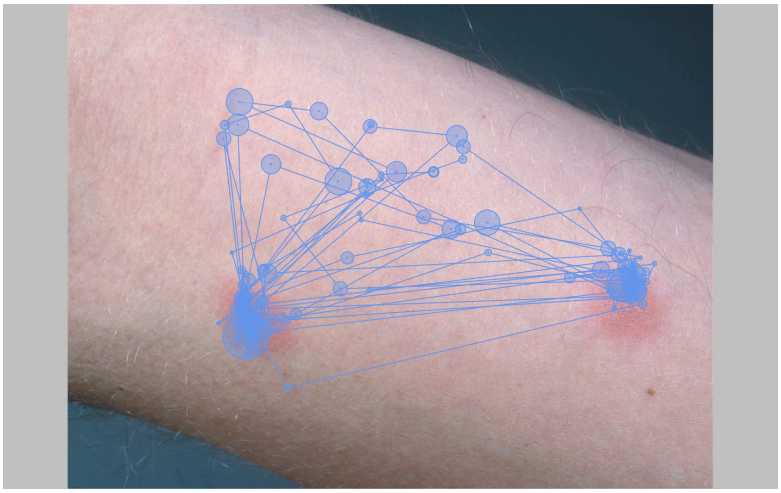
The physicians tended to evaluate their confidence towards the upper end of the spectrum, with a median of 70% confident over all narratives. But diagnostic confidence may be affected by many factors, including professional experience, case difficulty, and personality. We examine both individual images and physicians at the extremes of confidence to gain insight into the relationship between confidence and correctness in the dataset. Table 1 summarizes information for the three image cases that received the

[2] Some transcription imperfections may occur.

[3] A limited number of narratives in the dataset were labeled *half* correct if one of two final diagnoses given was correct, and *partially* correct if the final diagnosis was too broad. Here, we consider *half* to be correct, because in such cases the correct diagnosis was still identified, but *partial* to be incorrect, because the correct diagnosis was technically not identified.

... um two ... pa- ... pink to purple macules ... on the ... volar wrist differential diagnosis ... um fixed drug eruption ... bites ... urticaria ... uh ... diagnosis fixed drug eruption percent certainty fifty percent next ...

(a) Sample verbal description. Ellipses ("...") show pauses.



(b) Sample eye movement visualization. Circles represent *fixations*, where the center is the point of fixation and the radius is proportional to the time fixating at that point. Lines represent *saccades* (movements) between fixation points.

Figure 2: Sample verbal description and eye movements for one narrative. The final diagnosis is correct and the physician was 50% confident.

Confidence	Conf.	% Correct	Rank
Highest	100	100	2
	90	100	5
	90	100	1
Lowest	50	24	25
	50	35	29
	45	0	20

Table 1: Images receiving highest and lowest median confidence values. Difficulty ranking provided by a dermatology expert with 1 reflecting the easiest image and 30 the most difficult.

Confidence	Conf.	% Correct	Exp.
Highest	90	53	R
	85	50	A
	85	41	R
Lowest	38	39	A
	30	48	R
	15	37	R

Table 2: Most and least confident physicians by median confidence values given over all images. The last column shows experience level: experienced attending (A) or resident (R) physician.

highest median confidence values and the three that received the lowest. A domain expert (dermatologist and clinical educator) who was not a subject in the experiment gave each image a unique difficulty ranking from 1 to 30, where the image ranked number 1 was considered the easiest to support a correct diagnosis, and 30 the most difficult. As expected, the highest confidence images were among the easiest, and vice versa. Accordingly, the higher confidence images were correctly diagnosed by every physician, while those receiving the lowest confidence were correctly diagnosed much less often. The negative correlation between image difficulty and median physician confidence was significant using Spearman's rank correlation ($r_s = -0.544$, $p < 0.005$). In other words, higher levels of case difficulty were associated with lower levels of physician confidence. In contrast, examination of the most and least confident physicians yields less intuitive results. The physicians with the highest and lowest median confidence values are shown in the top and bottom halves of Table 2, respectively. Notably, each of the two groups contained both resident dermatologists-in-training and attending physicians with careers spanning multiple decades. Also, the most confident physicians were only correct roughly half of the time, and the least confident physicians' correctness appears quite similar. While this may reflect the sample size, the observation is interesting nonetheless. Clearly, this points to how complicated diagnostic self-awareness is, and how potentially useful it would be to computationally infer a physician's self-awareness for diagnostic cases based on their behaviors.

3.2 Confidence Binning

Nearly all confidence values given were multiples of five, or simply numbers close to 100, such as 99%.[4] This makes discretization preferable to using real-numbered values for confidence. Additionally, the analyses in Section 3.1 revealed patterns of over- or underconfidence in individual physicians. What this indicates is that "high" and "low" confidence involve different numerical values in the minds of different physicians. This subjectivity could be problematic in doctor-patient interactions and it adds complexity for predictive modeling involving confidence. To explore the impact, we devise two alternative binary binning schemes: *generalized* bins, based on the performance of all physicians in the dataset, and *personalized* bins, based on each individual physician's performance in the training data only. In terms of application, consider a diagnostic support system which could establish a history for each physician who uses it. Such a system could implement a generalized binning scheme and predictive model for new users, and later, after learning from repeated exposure to a given physician, switch to a model based on that physician's individual performance. In addition, binning choice may be influenced by context: in a clinical tutoring system, it may be preferable to compare learners to experienced physicians as a target population. For the *generalized* binning scheme, a confidence value greater than or equal to the median over all physicians is considered *high*, while a value below is considered *low*. This results in a slight imbalance towards *high* confidence (56% of narratives).[5] We construct the *personalized* binning scheme similarly, but using a given physician's own median confidence in the training data as the dividing line. In this case, *high* confidence accounts for 58% of the narratives, similar to that of the generalized bins. Calling a physician's median confidence *high* lets us better distinguish the problem cases: cases of underconfidence should be strictly less than their "typical" confidence, while cases of overconfidence should be at or above typical. The binning scheme used does not affect the correctness value for each narrative, but it does change the distribution of high and low confidence, with the *generalized* scheme favoring over- and underconfidence, and the *personalized* scheme favoring appropriate confidence. Arguably, the latter is a better reflection of the expected: over- and underconfidence as the minority classes.

4 Approach and Methodology

There are many ways to approach the problem of predicting physicians' diagnostic self-awareness. Here we formulate two classification problems, each tested under both binning schemes, yielding a total of four classification models. We also outline the performance evaluation experiments for the models.

4.1 Classification Problems

We define two classification problems based on the chart in Figure 1 (above). First, we define *Confidence Only*, which ignores correctness (the horizontal dimension of Figure 1) and predicts only confidence as a binary *high* or *low*. Intuitively, low confidence might be considered a warning sign for a diagnosis, alerting a physician to seek additional insight or information.[6] This first problem was used as a stepping stone to explore and better understand confidence, before incorporating correctness. Next, we define *Confidence & Correctness*, which relates confidence with the correctness of the diagnosis (considering all four quadrants in Figure 1, individually) to better address the more problematic, but interesting, cases. Distinguishing these four classes could be of use to intelligent tutoring or clinical support systems, which could respond differently to over- or underconfident users. In general, the full separation of these classes could ultimately allow for deeper analysis of physician self-awareness.

4.2 Model Evaluation

Before any development took place, the 632 narratives were randomly divided into three subsets: 442 (70%) for training (*dev-train*), 95 (15%) for testing during development and tuning (*dev-test*), and 95

[4]There were only a few exceptions: one physician gave three values of 3%, another gave a 33% and a 66% (rounded down from "two-thirds"), and a third gave a 33%. The latter three cases could also seem intuitive depending on how many conditions were listed in the differential diagnosis. For example, 66% might indicate that one disease seemed twice as likely as a another.

[5]Other simple binning schemes dividing up the 0-100% range were explored, but this binary version allowed for a more systematic approach to both generalized and personalized binning, without sacrificing performance.

[6]Normally, a physician would likely administer tests after the differential diagnosis, before reaching a final diagnosis.

(15%) for final summative evaluation after all development was completed (*heldout-test*). All three subsets have similar class distributions. Each of the four classification models were evaluated in two ways: (1) by training the model on the union of the *dev-train* and *dev-test* sets and testing on the *heldout-test* set, and (2) by running 50 randomized iterations of 10-fold cross-validation on the entire collection of 632 narratives. The first evaluation experiment addresses the problem of overfitting by excluding the *heldout-test* set from all development, while the second addresses the problem of sampling bias in the initial set divisions. The results are described in Section 5.2.

5 Models and Results

Here we describe the development and performance of each of the four computational models outlined in Section 4. We report on logistic regression, which had the best performance in all metrics for all experiments, after dimensionality reduction (see Section 5.1). The feature selection and modeling was implemented in Python with the `scikit-learn` machine learning library (Pedregosa et al., 2011).

5.1 Feature Extraction and Selection

A total of 60 features were examined (see Table 3). The features represented three modalities, motivated by the task the physicians performed and knowledge about dimensions of clinical expertise in this domain: *verbal*, composed of lexical, prosodic, and structural features of the narratives; *eye movement*, consisting of features of fixations and saccadic eye movements; and truly *multimodal* features, consisting of overlapping or simultaneously occurring features from the other two modalities, to reflect integrated multimodal semantics. Continuing with the theme of personalization, we also created a fourth category of *personal* features, with demographics of the physician and statistics about their confidence and correctness in the training data, in order to model their "past" performance. The latter simulates how a system could learn from experience with a particular physician.

As discussed in Section 2, verbal features of confidence have been studied before, and many of the verbal features used here are inspired by previous work. Some verbal features are based on word choice, such as *amplifiers* (e.g. *definitely*, *sure*) and *modals* (e.g. *could*, *might*),[7] while other have to do with silences (or pauses) or prosody. The eye movement and multimodal features are mostly concerned with fixations, as it seems intuitive that fixation may be associated with thoughtfulness about a particular area of the image, which may in turn reflect a physician's confidence.

Initial feature selection was performed on the development data (*dev-train* and *dev-test*) using `scikit-learn`'s random forest ensemble classifier. This allowed for human-friendly inspection of useful features. Random forests (Breiman, 2001) are an ensemble method in which numerous decision trees are constructed, each trained on a randomized subset of the development data, which allows for the utility of features to be evaluated on many sub-distributions of the data. The importance of a feature can then be approximated as the sum of the error reduction at each node that splits on that feature, weighted by the population size at that node. This reflects the fact that features used near the root of the tree often handle a larger number of individuals. The importance values for all features will sum to 1. We consider any feature that appeared in the top 20 of the ranked features for any model to be important, and all such types of features are marked in bold in Table 3. Interestingly, the useful features for all classification models were almost the same, with a few transpositions in the ordering. The exception was *past confidence*, which was useful under generalized, but disappeared under personalized, as expected, since the personalized scheme effectively normalizes each physician's confidence values.

Interpreting the results for the verbal features, *silence duration* (statistics about the durations of all silences) and the *duration of narrative* were most useful. Intuitively, this may relate to thoughtfulness or contemplation. Additionally, *words per second*, or speech rate, was also useful, again perhaps relating to more careful or thorough inspection/diagnosis. As discussed earlier, ties between speech and confidence have been well-studied, while eye movements are underreported. It seems intuitive that eye movement

[7] Such word-choice features were mostly based on lexical lists, and some overlap may occur. The *cutaneous conditions* feature contained multiword expressions. These could be improved by using resources such as UMLS (`http://www.nlm.nih.gov/research/umls/`) or WordNet (`http://wordnet.princeton.edu/`).

Verbal (29)		Multimodal (14)
Duration of narrative	Pronouns 1st (n, %)	**% of initial silence time fixating**
Number of silences	Pronouns 3rd (n, %)	**% of total silent time fixating**
Silence duration ($\underline{\Sigma}$, $\underline{\mu}$, $\underline{\sigma}$)	Modals (n, %)	**% of total fixation time silent**
Duration of initial silence	Amplifier words (n, %)	Words per second during fixation
Number of filled pauses	Speculative words (n, %)	**Pitch during fixations** ($\underline{\mu}$, $range$)
Word type-token ratio	Negations (n, %)	**Intensity during fixations** ($\underline{\mu}$, $range$)
Words per second	**Pitch** ($\underline{m}$, M, μ)	Pitch of filled pauses (m, M, μ)
Cutaneous conditions (n, %)	Intensity (m, M, μ)	Intensity of filled pauses (m, M, μ)

Eye movement (11)		Personal (6)	
Fixation duration ($\underline{\Sigma}$, μ, σ)	**Number of fixations**	Attending vs. Resident	Past correctness
Saccade duration ($\underline{\Sigma}$, $\underline{\mu}$, $\underline{\sigma}$)	**% image area fixated**	Years of experience	
Saccade amplitude ($\underline{\Sigma}$, $\underline{\mu}$, $\underline{\sigma}$)		**Past confidence** ($\underline{m}$, M, μ)	

Table 3: Features examined for classification (60 total), grouped by modality. Symbols in parentheses indicate statistics over all occurrences of a feature in a narrative: raw count (n), raw count divided by the total number of words (%), sum (Σ), mean (μ), standard deviation (σ), min (m), max (M), range ($range$). Useful features are boldfaced. If a feature has multiple statistics, the useful ones are underlined.

features may be more related to correctness. For example, the most useful eye movement feature was *% image area fixated*, computed using a grid overlaid onto the image. If more of the image was fixated upon, then it may have contained more areas of interest, or more visual evidence may have been sought, which may also be related to case difficulty. Similarly, features of *saccade amplitude* (the angle of a saccadic eye movement) may reflect physicians feeling a need to explore additional visual evidence by switching focus between distant areas in an image. It is not surprising that the useful individual features from verbal and eye movement modalities were also useful when combined as multimodal features. In particular, simultaneous silence and fixation were the most useful, which again might indicate contemplation and analytical cognitive processing. This suggests that expression of confidence and diagnostic self-awareness is at least partially a multimodal phenomenon.

Although the random forest method could be used for dimensionality reduction, we instead use Principle Component Analysis (PCA) in evaluation below, as it gave better performance gains in development. The purpose of the random forest method was to examine which verbal, eye movement, and multimodal features were most informative for classification, as we are interested in understanding how these modalities relate to confidence and correctness. The latent features resulting from PCA are linear combinations of the features, and thus would not allow for such inspection. The number of PCA components was optimized for classification accuracy in cross-validation for each of the four classification models. Each problem had a different number of principal components, indicating that both the binning scheme and the classification problem type affected which features were identified as more collectively discriminative by PCA.

5.2 Results and Evaluation

Heldout narratives We addressed the problem of overfitting by withholding 15% ($n = 95$) of the narratives as an unseen final evaluation set. All predictive models performed well above their respective majority class baselines (see Table 4). The Confidence Only models were able to reach higher accuracy, precision, and recall than the joint Confidence & Correctness models. The exception is the accuracy relative to baseline for personalized Confidence Only, which may be due to its higher baseline. As mentioned in Section 3.2, the generalized binning scheme is biased towards over- and underconfidence, and the personalized towards appropriate confidence. The per-class metrics (not shown here) reflect this fact, with overconfidence having higher precision and recall under generalized binning than under personalized. Additionally, under the personalized scheme underconfidence is particularly underrepresented and thus more difficult to predict.

Binning	Problem	N	Majority Class	% BL	% Acc.	P	R
Generalized	Conf. Only	2	High Confidence	53	76 (+23)	0.76	0.76
	Conf. & Corr.	4	Overconfidence	37	53 (+16)	0.42	0.42
Personalized	Conf. Only	2	High Confidence	65	77 (+12)	0.75	0.73
	Conf. & Corr.	4	Appropriate High	37	53 (+16)	0.38	0.42

Table 4: Performance metrics for the *heldout-test* set under each binning scheme with logistic regression and PCA. All four models performed well above the majority class baselines (% BL) of their respective problems (each with N many class labels). Precision (P) and recall (R) are each macro-averaged.

Random cross-validation A potential drawback of the initial development strategy used here is that the initial random splits may bias classification models. To address this problem, after the heldout testing, 50 randomized iterations of 10-fold cross-validation were performed on the total collection of narratives, the results of which are in Table 5. The personalized binning scheme was designed to mimic a system that could adapt to a physician's performance history, and thus the statistics used for personalized confidence binning were recomputed on the training data within each individual cross-validation fold. It is therefore not possible to establish a baseline for the personalized confidence binning outside of a given fold. Instead, we take the mean of the percent accuracy *above baseline* from each test fold ($\frac{1}{k}\sum_{i=1}^{k}(accuracy_i - baseline_i)$). All models performed well above their respective baselines, which is in line with observations from heldout testing.

Binning	Generalized		Personalized	
Problem	C.O.	C&C	C.O.	C&C
Acc. above baseline	+14	+9	+13	+12
Precision	0.70	0.25	0.69	0.32
Recall	0.70	0.38	0.57	0.37

Table 5: Performance metrics for logistic regression with 50 randomized iterations of cross-validation using all narratives for Confidence Only (C.O.) and Confidence & Correctness (C&C). We average the accuracy above baseline from each individual fold. Precision and recall are each macro-averaged for each problem.

Feature modality	Generalized		Personalized	
	C.O.	C&C	C.O.	C&C
V	+13	+9	+12	+11
E	+7	+6	+11	+10
MM	+7	+4	+6	+5
V+E	+13	+9	+13	+11
V+MM	+14	+8	+11	+11
E+MM	+10	+6	+13	+11
V+E+MM	+14	+9	+13	+12

Table 6: Modality study with cross-validation for Verbal (V), Eye movement (E), and Multimodal (MM) features, measured in accuracy above respective baselines, averaged over all folds. Most modality combinations equaled or slightly improved on constituent modalities in isolation.

Modality study We also performed a study within the cross-validation testing to investigate the impact of different feature modality combinations on classification (see Table 6). Importantly, the verbal modality alone was more powerful than the eye movement or multimodal features, but most combinations of modalities resulted in slightly higher or equal accuracy compared to their isolated constituent modalities. This suggests that, as we projected, considering multiple modalities of a physician's behavior can help reveal their confidence and self-awareness, but also that verbal features are the most informative, likely since verbal expression is the primary means to tap into physicians' rich and tacit conceptual understanding of a diagnostic case. The multimodal features, which focused on combining verbal and eye movement data, did not improve performance over baselines as much as the simple combination of the individual verbal and eye movement features. One reason for this could be that a person's speech and eye movements are not perfectly temporally aligned (Vaidyanathan et al., 2012), and this asynchronous relationship may affect the meaningfulness of our multimodal feature measurements. Additionally, these eye movement features may be at a much finer spatial or temporal scale than the verbal features.

6 Conclusions

This study examined a dataset of medical narratives consisting of verbal descriptions, eye movements, and self-reported confidence values, and used it to model physicians' confidence in diagnosis, as well as their diagnostic self-awareness. The Confidence Only problem involves the expression of confidence based on clinicians' belief, but it is important to understand the relationship to clinicians' actual diagnostic performance. This distinction is key because, while predicting confidence alone is a stepping stone, self-awareness is the ability to additionally align one's confidence with unknown correctness, which involves human intuitive and analytical reasoning (another topic of interest to the medical field, see Hochberg et al. (2014)). Case studies of the most and least confident physicians revealed a complex relationship between confidence and correctness, and highlighted the need for exploring clinical self-awareness. We also defined a personalized binning scheme for physician confidence levels, taking into account a physician's past confidence when drawing the line between high and low confidence, and compared this to a generalized binning scheme based on performance of all physicians. In tandem, these approaches to confidence binning could be used by an intelligent diagnostic support system.

We incorporated previously unused eye movement information from this dataset, and introduced truly multimodal features which directly combined physicians' verbal and eye movement behaviors. While physicians' eye movement and multimodal features were not individually as powerful as verbal features, combinations of the three groups mostly produced classification improvements that were slightly better than, or at least as good as, their constituent feature groups in isolation. The best performance for the majority of models was achieved by considering features from all three modalities. This suggests that eye movements help convey confidence and diagnostic self-awareness. The multimodal features did not help as much, which we believe is explained by the more flexible temporal relationship between speech and eye movements in the human mind. We leave the multimodal alignment challenge to future work. Some pitch features implemented without speaker-dependent analysis were useful for classification, but future work may benefit from pitch feature representations that adapt to demographic variation. Another area for future work beyond the scope of this study includes examining alternative ways of combining confidence and correctness classes, such as merging the diagonals of Figure 1 into a binary classification of appropriate vs. inappropriate (i.e. the union of over- and underconfidence). Such alternatives may present additional challenges for classification, but could also provide benefits for simpler clinical support applications that may not be concerned with differentiating all four classes.

Acknowledgements

This work was supported by a seed award, and its dissemination partially by a Kodak Endowed Chair award, both from the Golisano College of Computing and Information Sciences at RIT. The original data collection was supported by NIH grant 1 R21 LM01003901A1. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. The authors also thank Rui Li, and appreciate the helpful comments from reviewers.

References

Eta S. Berner and Mark L. Graber. 2008. Overconfidence as a cause of diagnostic error in medicine. *The American Journal of Medicine*, 121(5A):S2–S23.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45:5–32.

Pat Croskerry. 2003. The importance of cognitive errors in diagnosis and strategies to minimize them. *Academic Medicine*, 78(8):775–780, August.

Pat Croskerry. 2008. Overconfidence in clinical decision making. *The American Journal of Medicine*, 121(5A):S24–S29.

Pat Croskerry. 2009. A universal model of diagnostic reasoning. *Academic Medicine*, 84(8):1022–1028, August.

Charles P. Friedman, Guido G. Gatti, Timothy M. Franz, Gwendolyn C. Murphy, Frederic M. Wolf, Paul S. Heckerling, Paul L. Fine, Thomas M. Miller, and Arthur S. Elstein. 2005. Do physicians know when their diagnoses are correct? *Journal of General Internal medicine*, 20:334–339, April.

Albert Gatt and Patrizia Paggio. 2013. What and where: An empirical investigation of pointing gestures and descriptions in multimodal referring actions. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 82–91, Sofia, Bulgaria, August 8-9.

Mark Graber, Ruthanna Gordon, and Nancy Franklin. 2002. Reducing diagnostic errors in medicine: What's the goal? *Academic Medicine*, 77(10):981–992, October.

Mark L. Graber, Nancy Franklin, and Ruthanna Gordon. 2005. Diagnostic error in internal medicine. *Archives of Internal Medicine*, 165:1493–1499, July 11.

Limor Hochberg, Cecilia Ovesdotter Alm, Esa M. Rantanen, Caroline M. DeLong, and Anne Haake. 2014. Decision style in a clinical reasoning corpus. BioNLP 2014.

Jay Katz. 1984. Why doctors don't disclose uncertainty. *Hastings Center Report*, 14:35–44.

Charles E. Kimble and Steven D. Seidel. 1991. Vocal signs of confidence. *Journal of Nonverbal Behavior*, 15:99–105.

Rui Li, Jeff Pelz, Pengcheng Shi, Cecilia Ovesdotter Alm, and Anne Haake. 2012a. Learning eye movement patterns for characterization of perceptual expertise. In *ETRA 2012 Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 393–396, Santa Barbara, CA, March 28-30.

Rui Li, Jeff Pelz, Pengcheng Shi, and Anne Haake. 2012b. Learning image-derived eye movement patterns for characterization of perceptual expertise. In *Proceedings of CogSci 2012*, pages 1900–1905.

Jackson Liscombe, Julia Hirschberg, and Jennifer J. Venditti. 2005. Detecting certainness in spoken tutorial dialogues. In *Proceedings of Interspeech 2005*, pages 1837–1840, Lisbon, Portugal.

Wilson McCoy, Cecilia Ovesdotter Alm, Cara Calvelli, Jeff B. Pelz, Pengcheng Shi, and Anne Haake. 2012. Linking uncertainty in physicians' narratives to diagnostic correctness. In *Proceedings of the ACL-2012 Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics (ExProM-2012)*, pages 19–27, Jeju, Republic of Korea, 13 July.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Verónica Pérez-Rosas, Rada Mihalcea, and Louis-Phillippe Morency. 2013. Utterance-level multimodal sentiment analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 973–982, Sofia, Bulgaria, August 4-9.

Heather Pon-Barry and Stuart M. Shieber. 2011. Recognizing uncertainty in speech. *EURASIP Journal on Advances in Signal Processing*, 2011(251753).

Erika Rogers. 1996. A study of visual reasoning in medical diagnosis. In *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*, pages 213–218, La Jolla, California, 12-15 July.

Klaus R. Scherer, Harvey London, and Jared J. Wolf. 1973. The voice of confidence: Paralinguistic cues and audience evaluation. *Journal of Research in Personality*, 7:31–44, June.

Vicki L. Smith and Herbert H. Clark. 1993. On the course of answering questions. *Journal of Memory and Language*, 32(1):25–38.

Preethi Vaidyanathan, Jeff Pelz, Wilson McCoy, Cara Calvelli, Cecilia Ovesdotter Alm, Pengcheng Shi, and Anne Haake. 2012. Visualinguistic approach to medical image understanding. In *Proceedings of the AMIA 2012 Annual Symposium*, Chicago, Illinois, November.

Kathryn Womack, Wilson McCoy, Cecilia Ovesdotter Alm, Cara Calvelli, Jeff B. Pelz, Pengcheng Shi, and Anne Haake. 2012. Disfluencies as extra-propositional indicators of cognitive processing. In *Proceedings of the ACL-2012 Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics (ExProM-2012)*, pages 1–9, Jeju, Republic of Korea, 13 July.

Kathryn Womack, Cecilia Ovesdotter Alm, Cara Calvelli, Jeff B. Pelz, Pengcheng Shi, and Anne Haake. 2013. Markers of confidence and correctness in spoken medical narratives. In *Proceedings of Interspeech 2013*, pages 2549–2553, Lyon, France, August 25-29.

Towards Semantic Validation of a Derivational Lexicon

Britta D. Zeller* **Sebastian Padó†** **Jan Šnajder§**
*Heidelberg University, Institut für Computerlinguistik
69120 Heidelberg, Germany
†Stuttgart University, Institut für maschinelle Sprachverarbeitung
70569 Stuttgart, Germany
§University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
zeller@cl.uni-heidelberg.de pado@ims.uni-stuttgart.de jan.snajder@fer.hr

Abstract

Derivationally related lemmas like *friend*$_N$ – *friendly*$_A$ – *friendship*$_N$ are derived from a common stem. Frequently, their meanings are also systematically related. However, there are also many examples of derivationally related lemma pairs whose meanings differ substantially, e.g., *object*$_N$ – *objective*$_N$. Most broad-coverage derivational lexicons do not reflect this distinction, mixing up semantically related and unrelated word pairs.

In this paper, we investigate strategies to recover the above distinction by recognizing semantically related lemma pairs, a process we call *semantic validation*. We make two main contributions: First, we perform a detailed data analysis on the basis of a large German derivational lexicon. It reveals two promising sources of information (distributional semantics and structural information about derivational rules), but also systematic problems with these sources. Second, we develop a classification model for the task that reflects the noisy nature of the data. It achieves an improvement of 13.6% in precision and 5.8% in F1-score over a strong majority class baseline. Our experiments confirm that both information sources contribute to semantic validation, and that they are complementary enough that the best results are obtained from a combined model.

1 Introduction

Morphological processing forms the first step of virtually all linguistic processing toolchains in natural language processing (NLP) and precedes other analyses such as part of speech tagging, parsing, or named entity recognition. There are three major types of morphological processes: (a) *Inflection* modifies word forms according to the grammatical context; (b) *derivation* constructs new words from individual existing words, typically through affixation; (c) *composition* combines multiple words into new lexical items. Computational treatment of morphology is often restricted to normalization, such as *lemmatization* (covering inflection only) or *stemming* (covering inflection and derivation heuristically, Porter (1980)).

An important reason is that English is morphologically a relatively simple language. Composition is not marked morphologically (*zoo gate*) and an important derivational pattern is *zero derivation* where the input and output terms are identical surface forms (*a fish / to fish*). Thus, lemmatization or stemming go a long way towards treating the aspects of English morphology relevant for NLP. The situation is different for languages with a complex morphology that calls for explicit treatment. In fact, recent years have seen a growing body of computational work in particular on derivation, which is a very productive process of word formation in Slavic languages but also in languages more closely related to English, like German (Štekauer and Lieber, 2005).

Derivation comprises a large number of distinct patterns, many of which cross part of speech boundaries (nominalization, verbalization, adjectivization), but some of which do not (gender indicators like *master / mistress*, approximations like *red / reddish*). A simple way to conceptualize derivation is that it partitions a language's vocabulary into *derivational families* of derivationally related lemmas (cf. Zeller et al. (2013), Gaussier (1999)). In WordNet, this type of information has been included to some extent by so-called "morpho-semantic" relations (Fellbaum et al., 2009), and the approach has been applied to languages other

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1728–1739, Dublin, Ireland, August 23-29 2014.

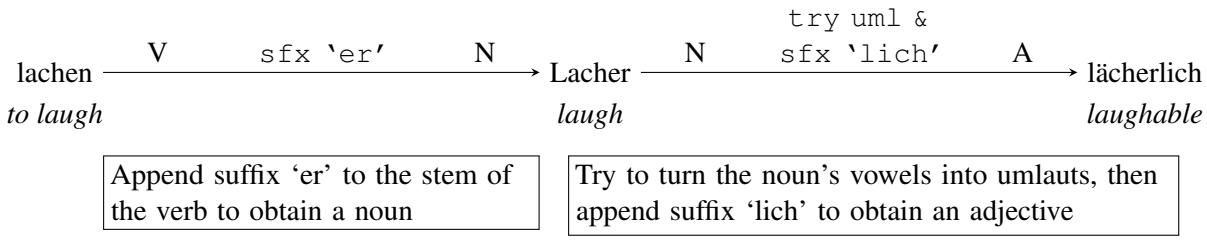


Figure 1: (Part of) a derivational family from DERIVBASE including derivational rules

than English (Bilgin et al., 2004; Pala and Hlaváčková, 2007). Another source of derivational information are stand-alone derivational lexicons such as CatVar (Habash and Dorr, 2003) for English, DERIVBASE (Zeller et al., 2013) for German, or the multilingual CELEX (Baayen et al., 1996).

Recent work has demonstrated that NLP can benefit from derivational knowledge. Shnarch et al. (2011) employ derivational knowledge in recognizing English textual entailment to better gauge the semantic similarity of text and hypothesis. Padó et al. (2013) improve the prediction of German semantic similarity judgments for lemma pairs by backing off to derivational families for infrequent lemmas. Luong et al. (2013) and Lazaridou et al. (2013) improve distributional semantic representations.

Note that all of these applications make use of derivational knowledge to address various *semantic* tasks, working on the assumption that derivationally related words, as represented in derivational lexicons, are strongly semantically related. This assumption is not completely warranted, though. The development of wide-coverage derivational lexicons is generally driven by morphological information, using for example finite-state technology (Karttunen and Beesley, 2005) to characterize known derivational patterns in terms of string transformations. Even though there is a strong correlation in derivation between morphology and semantics, it is not perfect. The absence of (synchronic) semantic relatedness can have a number of reasons, including accidental instantiation of derivational patterns (*corn – corner*) and diachronic meaning drift (*dog* (*animal*) – *dogged* (*determined*)). In other words, a substantial number of the lemma pairs in those lexicons are false positives regarding the level of semantic relatedness.

Our goal in this paper is to ameliorate this situation by developing strategies for the *semantic validation* of derivational lexicons, i.e., methods to determine, for lemma pairs that are derivationally related at the morphological level, whether they are in fact semantically related. We base our study on the German derivational lexicon DERIVBASE, and start by assessing which strategies can be used for its semantic validation (Section 2). In Sections 3 and 4, we analyze the contributions of semantic information (distributional semantics) as well as structural information (derivational rules). On the basis of our observations, we train a classifier that is able to semantically validate DERIVBASE at 89.9% F_1-score (Section 5), significantly outperforming a majority-class baseline of 84.1%. Section 6 reviews related work. Section 7 concludes the paper and outlines future work.

2 A Lexicon for German Derivation

2.1 DERIVBASE

DERIVBASE (Zeller et al., 2013) is a freely available derivational lexicon for German.[1] We used a rule-based framework to define derivation rules that cover suffixation, prefixation, and zero derivation as well as stem changes. Following the work of Šnajder and Dalbelo Bašić (2010), derivational processes are defined using derivational rules and higher-order string transformation functions. The only requirements for this method are (a) a comprehensive set of lemmas and (b) knowledge about admissible derivational rules, which can be gathered, for example, from linguistics textbooks.

Figure 1 shows a small sample from a derivational family with three lemmas and two derivational rules, one turning a verb into the corresponding event noun (in this case a semelfactive), and one turning the event into an adjective associated with it. Note that there are two perspectives on such a family: It can

[1] http://www.cl.uni-heidelberg.de/~zeller/res/derivbase/

DERIVBASE release	"Positive" class	Precision %	Recall %
1.2 (Zeller et al., 2013)[3]	R and M	83.0	71.0
1.4 (our analysis)	R and M	85.1	91.4
1.4 (our analysis)	R only	76.7	93.8

Table 1: DERIVBASE evaluation across releases on the DERIVBASE release 1.2 P and R samples

either be seen as a set of lemmas, or as a set of (independent) lemma pairs. We will assume the latter perspective in this paper, leaving questions of global coherence for future work.

DERIVBASE is a good example for the problems sketched in Section 1. It is defined purely on morphological grounds, without semantic validation of derivational families. Consequently, it contains a substantial number of words that are not semantically related.

2.2 Morphological and Semantic Relatedness in DERIVBASE

Our original evaluation of the quality of DERIVBASE in Zeller et al. (2013) was based on manually classified samples of lemma pairs. We introduce two samples, the "R sample", drawn from a large population of lemma pairs with high string similarity, in order to calculate recall, and the "P sample", drawn from the DERIVBASE families, in order to compute precision. Each lemma pair was classified into one of five categories (**R**: morphologically and semantically related; **M**: only morphologically related; **N**: not related; **L**: lemmatization errors; **C**: compounds) and inter-annotator agreement was checked to be substantial.[2] The overall best model (L123) showed 83% precision and 71% recall. However, this evaluation is limited in two important respects. First, it refers to DERIVBASE release 1.2 from 2013. Since then, we have extended DERIVBASE, e.g., with rules covering particle verbs, a very productive area of German derivation. Secondly, and more seriously, the previous evaluation considered all instances of **R** *and* **M** as true positives. In other words, in Zeller et al. (2013) we only evaluated the morphological relatedness of the lemma pairs but not the semantic relatedness.

We therefore start by presenting an evaluation of DERIVBASE focusing on the **R** instances in Table 1, reusing the DERIVBASE 1.2 "P" and "R" samples introduced in Zeller et al. (2013, see there for evaluation details). Between DERIVBASE 1.2 and 1.4, precision increased marginally and recall substantially, due mainly to the inclusion of rules that cover particle verbs. However, the numbers change substantially when only **R** (truly semantically related pairs) are counted as true positives. Recall increases by about 2.5%, but precision drops about 8.5%. Almost one quarter of all pairs in the lexicon are *not* semantically related.

A possible confounder of this analysis is that the "P sample" was drawn on DERIVBASE 1.2 and therefore does not include the novel items in DERIVBASE 1.4. We therefore created a novel DERIVBASE *1.4 extended sample* by combining the existing "P sample" with those pairs from the "R sample" that are in the coverage of a DERIVBASE rule as of DERIVBASE 1.4, resulting in 2,545 lemma pairs.

This DERIVBASE 1.4 extended sample will form the basis of all our analyses in this paper. The class distribution in the new sample is similar, but not identical, to the old P sample, as shown in Table 2. The relative frequency of **R** drops another 2%. Since this number also corresponds to the precision of the resource, the precision of the extended sample is 74.6%.

There are almost no compound errors **C**, which is not surprising given the rule-based construction of the lexicon, and only a relatively small number (about 5%) of lemmatization errors **L**, which fall outside the scope of our work. In contrast, both **N** and **M** occur with substantial frequency: Each class accounts for around 10% of the pairs. An analysis of **N** shows many cases of rule overgeneration: These are often pairs of lemmas whose stems are sufficiently similar that they might be related, e.g., by stem-changing derivation rules. Although such rules are valid in other contexts ($Verkauf_N$ – $Verkäufer_N$ (*selling – seller*)),

[2] Although we believe semantic relatedness to be fundamentally a graded scale, we adopt a binary notion of it as a convenient operational simplification that is supported by the good inter-annotator agreement for manual labeling in DERIVBASE.

[3] DERIVBASE 1.2 corresponds to DERIVBASE "L123" in (Zeller et al., 2013, p. 1207).

	R	M	N	L	C
Frequency	1899	265	240	131	8
Percentage overall	74.6	10.4	9.5	5.2	0.3
Percentage on dev. set	75.5	10.3	9.0	4.8	0.3
Percentage of test set	72.6	10.6	10.6	5.9	0.3

Table 2: Class distribution in our new DERIVBASE 1.4 extended sample

erroneous application leads to **N** cases like *Blase*$_N$ – *Bläser*$_N$ (*bubble – blower*). Also, we find false matches of common noun rules with named entities (*Empire*$_N$ – *Empirismus*$_N$ (*Empire – empiricism*)).

In contrast, many cases of **M** (as sketched in Section 1) refer to different senses of the same stem. As an example, consider *beruhen*$_V$ – *unruhig*$_A$ (*to rest on – restless*), both related to *Ruhe*$_N$ *(rest)*. In other cases, one of the two lemmas appears to have undergone a meaning shift (*Rappel*$_N$ – *rappeln*$_V$ (*craze – to rattle*)). This is particularly prominent for particle verbs (*bauen*$_V$ – *erbaulich*$_A$ (*build – edifying*)).

We divide the DERIVBASE 1.4 extended sample into a development and a test partition (70:30 ratio); the subsequent analyses consider only the development set.

2.3 Hypotheses for Semantic Validation

The preceding analysis of DERIVBASE has established that the lexicon contains a substantial number (around one fourth) of lemma pairs that are not semantically related. Therefore, it is in need of *semantic validation*, i.e., a computational procedure that can filter out semantically unrelated words.

In this paper, we frame semantic validation as a binary classification task that classifies all lemma pairs within one derivational family as either semantically related or unrelated. We consider this a first step towards splitting the current, morphologically motivated, DERIVBASE families into smaller, semantically coherent, families. We base our work on two general hypotheses about the types of information that might be helpful in this endeavor.

Hypothesis 1. *Distributional similarity indicates semantic relatedness between derivationally related words.* The instances of polysemy and meaning shift that we observe, in particular in the **M** class, motivate the use of distributional similarity (Turney and Pantel, 2010) since we expect these lemma pairs to be distributionally less related than cases of true semantic relatedness.

Hypothesis 2. *Derivational rules differ in their reliability.* Both the evidence from **M** and **N** indicate that some rules are more meaning-preserving than others. We expect this to be tied to both lexical properties of the rules (particle verbs are more likely than diminutives to radically change meaning) as well as structural properties (more specific rules are presumably more precise than generic rules).

In the two following Sections, we will operationalize these hypotheses and analyze the development set of the DERIVBASE 1.4 extended sample with respect to their empirical adequacy.

3 Analysis 1: Distributional Similarity for Semantic Validation

3.1 Measuring Distributional Similarity

We examine semantic similarities as predicted by simple bag-of-words semantic space models built from the lemmatized SDeWaC (Faaß et al., 2010), a large German web corpus containing about 880 million words. We compute vectors for all words covered in DERIVBASE using a window of $\pm$5 words within sentence boundaries and considering the 10k most frequent lemma-part of speech combinations of nouns, verbs, and adjectives in SDeWaC as contexts. Distributional vectors are built from co-occurrences which are measured with Local Mutual Information (Evert, 2005). The semantic similarity is measured by the cosine similarity between the vectors. Despite the size of the corpus, many lemmas from DERIVBASE occur very infrequently, and due to the inflection in German, it is important to retrieve as many occurrences of each lemma as possible.

We therefore use a very permissive two-step lemmatization scheme that starts from lemmas from the lexicon-based TreeTagger (Schmid, 1994), which provides reliable lemmas but with relatively low coverage, and supplements them with lemmas and parts of speech produced by the probabilistic MATE toolkit (Bohnet, 2010) when TreeTagger abstains.

3.2 Frequency Considerations

The advantage of the string transformation-based construction of DERIVBASE is its ability to include infrequent lemmas in the lexicon, and in fact DERIVBASE includes more than 250,000 content lemmas, some of which occur not more than three times in SDeWaC. However, this is a potential problem when we build distributional representations for all lemmas in DERIVBASE since it is known from the literature that similarity predictions for infrequent lemmas are often unreliable (Bullinaria and Levy, 2007).

Our data conform to expectations in this regard – infrequent lemmas are indeed problematic for validating the semantic relatedness of lemma pairs. More specifically, the semantic similarity of *related* lemmas (**R**) is systematically underestimated, because the lemma pairs from our sample are often too infrequent to share any dimensions. Consequently, they receive a low or zero cosine even when they are semantically strongly related. For example, each of the lemmas *Drogenverkauf*$_N$ – *Drogenverkäufer*$_N$ (*drug selling – drug seller*) has only nine lemmas as dimensions, and those are completely disjoint. This underestimation constitutes a general trend. The model assigns cosine scores below 0.1 to 64% of the related pairs in the development set, cosines below 0.2 to 81%, and cosines below 0.3 to 87%. Such low scores are problematic for separating related from unrelated pairs.

Two-step lemmatization is important for the proper handling of infrequent words. Compared to just using TreeTagger, the TreeTagger+MATE vectors for *auferstehen*$_V$ – *auferstehend*$_A$ (*to resurrect – resurrecting*) share seven more dimensions, including *Jesus*, *Lord*, *myth*, and *suffering*. Correspondingly, the cosine value of this pair rises by 50%. Generally, the amount of zero cosines in the DERIVBASE 1.4 extended sample drops by 45% using two-step lemmatization compared to one-step TreeTagger lemmatization.

3.3 Conceptual Considerations

In addition to the frequency considerations discussed above, we find three conceptual phenomena that affect distributional similarity independently of the frequency aspects.

The first one is the influence of *parts of speech*. Derivational rules often change the part of speech of the input lemma, and the parts of speech of its context words change as well. This decreases context overlap. For example, *Überschätzung*$_N$ – *überschätzt*$_A$ (*overestimate – overestimated*) is assigned a cosine of merely 0.09. The upper half of Table 3 shows the top ten individual and shared context words for this pair, ranked by LMI. The context words of the noun are mainly nominal heads of genitive complements (*overestimation of possibility/force/...*), while the context words of the adjective comprise many adverbs (*totally, widely, ...*). None of the shared contexts rank among of the top ten for both target lemmas. This is even more surprising considering that German adjectives and adverbs have the same surface realization (as opposed to English) and are more likely to form matching context words.

The second phenomenon that we identified as influencing semantic similarity is *markedness* (Battistella, 1996). A considerable number of derivational rules systematically produce marked terms. A striking example is the feminine suffix "*-in*" as in *Entertainer*$_N$ – *Entertainerin*$_N$: Although the lemmas are intuitively very similar, their cosine is as low as 0.1. The reason is that the female versions tend to be used in contexts where the gender of the entertainer is relevant. This is illustrated in the lower half of Table 3. The first two contexts for both words (*actor, singer*) stem from frequent enumerations (*actor and entertainer X*) and are almost identical, but again the female versions are marked for gender. We also find two female given names. As a result, the target lemmas receive a low distributional similarity.

The third example are cases of mild meaning shifts that were tagged by the annotators as **R**. These are lemmas where the semantic relatedness is intuitively clearly recognizable but may be accompanied by pretty substantial changes in the distribution of contexts. Consider the semantically related pair *Absteiger*$_N$ – *absteigend*$_A$ (*descender (person) – descending/decreasing*). It achieves only a cosine of

word pair (l_1, l_2)	context(l_1)	context(l_2)	shared contexts(l_1, l_2)
Überschätzung – überschätzt (*overestimation – overestimated*), $cos = 0.09$	eigen (*own*) warnen (*to alert*) Möglichkeit (*possibility*) führen (*to lead*) Kraft (*force*) Bedeutung (*meaning*) Fähigkeit (*ability*) Leistungsfähigkeit (*performance*) neigen (*to tend*) Einfluß (*influence*)	völlig (*totally*) Problem (*problem*) Gefahr (*danger*) Autor (*author*) weit (*widely*) total (*totally*) ernst (*seriously*) überhöht (*excessive*) gnadenlos (*mercilessly*) Hollywood (*Hollywood*)	völlig (*totally*) Möglichkeit (*possibility*) Bedeutung (*meaning*) Gefahr (*danger*) Einfluß (*influence*) überhöht (*excessive*) Macht (*power*) gnadenlos (*mercilessly*) Kraft (*force*) häufig (*frequent*)
Entertainer – Entertainerin (*entertainer – female entertainer*), $cos = 0.1$	Sänger (*singer*) Schauspieler (*actor*) Musiker (*musician*) Harald (*male given name*) Moderator (*anchorman*) Schmidt (*surname*) groß (*big*) Künstler (*artist*) Talent (*talent*) gut (*good*)	Sängerin (*female singer*) Schauspielerin (*actress*) Helga (*female given name*) Mutter (*mother*) berühmt (*famous*) brillant (*brilliant*) Lisa (*female given name*) Künstlerin (*female artist*) verstorben (*deceased*) Talent (*talent*)	Schauspieler (*actor*) Musiker (*musician*) **Talent** (*talent*) bekannt (*well-known*) Sängerin (*female singer*) beliebt (*popular*) groß (*big*) berühmt (*famous*) Sportler (*sportsman*) Schauspielerin (*actress*)

Table 3: Top ten individual and shared context words for *Überschätzung*$_N$ *– überschätzt*$_A$ (*overestimation – overestimated*) and *Entertainer*$_N$ *– Entertainerin*$_N$. Individual context words are ranked by LMI, shared context words by the product of their LMIs for the two target words. Shared context words that occur in the top ten contexts for both words are marked in **boldface**.

0.005, because *Absteiger* is almost exclusively used to refer to relegated sport teams while *absteigend* is used as a general verb of scalar change.

3.4 Ranking of Distributional Information

Given the results reported above, the standard distributional approach of using plain cosine scores to measure the absolute amount of co-occurrences does not seem very promising, due to the low absolute numbers of shared dimensions of the two lemmas. We expect other similarity measures, e.g., the Lin measure (Lin, 1998), to perform equally poorly since they do not change the fundamental approach. Also, although using a large corpus for semantic space construction might ameliorate the situation, we would prefer to make improvements on the modeling side of semantic validation.

We follow the ideas of Hare et al. (2009) and Lapesa and Evert (2013) who propose to consider semantic similarity in terms of ranks rather than absolute values. The advantage of rank-based similarity is that it takes the density of regions in the semantic space into account. That is, a low cosine value does not necessarily indicate low semantic relatedness – provided that the two words are located in a "sparse" region. Conversely, a high cosine value can be meaningless in a densely populated region. A second conceptual benefit of rank-based similarity is that it is directed: It is possible to distinguish the "forward" rank (the rank of l_1 in the neighborhood of l_2) and the "backward" rank (the rank of l_2 in the neighborhood of l_1). The previous studies found rank-based similarity to be beneficial for the prediction of priming results. In our case, it suggests a refined version of our Hypothesis 1:

Hypothesis 1'. *High* rank-based *distributional similarity indicates semantic relatedness between derivationally related words.*

4 Analysis 2: Derivational Rules for Semantic Validation

As discussed in Section 2.3, a second source of information that should be able to complement the problematic distributional similarity is provided by the derivational rules that are encoded in DERIVBASE (cf. the arrows in Figure 1). Our intuition is that some rules are "semantically stable", meaning that they reliably connect semantically similar lemmas, while other rules tend to cause semantic drifts. To examine

this situation, we perform a qualitative analysis on all lemma pairs connected by rule paths of length one ("simplex paths"), which are easy to analyze. Longer paths ("complex paths") are considered below.

We find that rules indeed behave differently. For example, the "*-in*" female marking rule from Section 3.3 is very reliable: every lemma pair connected by this rule is semantically related. At the other end of the scale, there are rules that consistently lead to semantically unrelated lemmas, e.g., the *"ver-"* noun-verb prefixation: *Zweifel*$_N$ – *verzweifeln*$_V$ (*doubt – to despair*). Foreign suffixes like "*-ktiv*" in *instruieren*$_V$ – *instruktiv*$_A$ (*to instruct – instructive*) retain semantic relatedness in most cases, but sometimes link actually unrelated lemmas (**N**, **C**, **L**). For example, *Objektiv*$_N$ – *Objektivismus*$_N$ (*lens – objectivism*), is an **N** pair for the suffix "*-ismus*". Finally, zero derivations and very short suffixes are less reliable: Since they easily match, they are often applied to incorrectly lemmatized words (**L**). For example, the "*-n*" suffix, which relates nationalities with countries (*Schwede*$_N$ – *Schweden*$_N$ (*Swede – Sweden*)). It matches many wrongly lemmatized nouns due to its syncretism with the plural dative/accusative suffix *-n*, as in *Schweineschnitzel*$_N$ – *Schweineschnitzeln*$_N$ (*pork cutlet – pork cutlets*$_{dat/acc\text{-}pl}$). This suggests that *rule-specific reliability* is a promising feature for semantic validation. Fortunately, due to its construction, DERIVBASE provides a rule chain for each lemma pair so that these reliabilities can be "read off". For other rules, however, the variance of the individual lemma pairs that instantiate the rule is large, and the applicability of the rule is influenced by the particular combination of rule and lemma pair. Such cases suggest that distributional knowledge and structural rule information should be combined, a direction that we will pursue in the next section.

On word pairs that are linked by "complex paths", i.e., more than one rule (*lachen*$_V$ – *lächerlich*$_A$ in Figure 1), our main observation in this respect is that rule paths show a clear "weakest link" property. One unreliable rule can be sufficient to cause a semantic drift, and only a sequence of reliable rules is likely to link two semantically related words. We will act on this observation in the next section.

5 A Machine Learning Model for Semantic Validation

5.1 Classification

The findings of our analyses suggest that the decision to classify lemma pairs as semantically related or unrelated can draw on a range of considerations. We therefore decided to adopt a machine learning approach and phrase semantic validation as a binary classification task, using the analyses we performed in Sections 3 and 4 as motivation for feature definition.

We train a classifier on the development portion of the DERIVBASE 1.4 extended sample (1,780 training instances, cf. Section 2.2). We learn a binary decision: Semantic relatedness (**R**) vs. non-semantic relatedness (**M**, **N**, **C**, **L**) within derivationally related pairs. For classification, we use a nonlinear model: Support Vector Machine (SVM) with a Radial Basis Function (RBF) kernel. Using the RBF kernel allows us to capture the non-linear dependencies between the features.[4] We rely on LIBSVM (Chang and Lin, 2011), a well-known SVM implementation. We optimize the C and γ hyperparameters of the SVM model using 3-fold cross-validation on the training data (i.e., the development portion of the extended sample).

5.2 Features

Our analyses motivate three feature groups comprising 35 individual features: Distributional, derivation rule-based ("structural"), and hybrid features. Table 4 gives a list.

Distributional features. All distributional features apply to the lemma or pair level. They are calculated from our BOW model with permissive lemmatization (Section 3.1). We use absolute and rank-based cosine similarity (Section 3.4) as well as the number of shared contexts (computed with LMI, cf. Section 3.3) and lemma frequency. To speed up processing, we compute the forward rank similarity for a lemma pair (l_1, l_2) not on the complete vocabulary but by pairing l_1 with a random sample of 1,000 lemmas from DERIVBASE (plus l_2 if it is not included). We do the computation analogously for the backward rank.

[4]The nonlinear SVM model outperforms a linear SVM. The difference is 0.8% F-Score, statistically significant at p=0.05.

Feature group (# features)	Type	Feature name (# features)	Description
Distributional (6)	l	Lemma frequency (2)	Normalized SDeWaC corpus lemma frequencies
	p	Cosine similarity	Standard cosine lemma similarity
	p	Dimensions shared	Number of shared context words
	p	Cos. rank similarity (2)	Rank-based forward and backward similarity
Structural (25)	r	Rule identity (11)	Indicator features for the top ten rules in the dev set + one aggregate feature for the rest
	r	Rule reliability	Percentage of rule applications on **R** pairs among all applications of the rule in dev set
	r	Rule frequency rank (2)	Rank-based rule frequency in DERIVBASE
	r	Avg. string distance (2)	Avg. Levenshtein distance for all rule instances
	p	POS combinations (6)	Indicator features for lemma POS combinations
	p	Path length	Length of the shortest path between the lemmas
	p	String distance (2)	Dice bigram coefficient; Levenshtein distance
Hybrid (4)	r	Average rank sim (2)	Frequency-weighted average rank similarity of rules on shortest path
	p	Rank sim deviation (2)	Difference between lemma pair rank similarity and average rule rank similarity

Table 4: Features used to characterize derivationally related lemma pairs. "Type" indicates the level at which each feature applies: *l* lemma level, *p* pair level, *r* rule level.

Structural features. The structural features encode properties of the rules and paths in DERIVBASE. Most features apply to the level of derivation rules. This includes the identity of the rule; its reliability (estimated as the ratio of its application on **R** pairs among all its applications on the dev set); its frequency rank among all rules (as a measure of specificity)[5]; and the average Levenshtein distance between the input and output lemmas (estimating rule complexity by measuring the amount of string modification).

For lemma pairs linked by complex paths (i.e., more than one rule, cf. Figure 1), the question arises how the rule-level features should be computed. Following our observations on "weakest link" behavior in Section 4, we always combine the feature values for the individual rules adopting the most pessimistic combination function (e.g., minimum in the case of reliability, maximum in the case of frequency rank).

Three more structural features are computed directly at the lemma pair level: their part of speech combination (e.g., *"NV"* for $oxide_N - oxidate_V$), the length of the shortest path connecting them, and the Levenshtein and Dice string distances between the two lemmas.

Hybrid features. Hybrid features combine rule-based and distributional information to avoid their respective shortcomings. We work with two hybrid features, one at rule level and one at pair level. The rule-level feature models the reliability of the rule. It is the average rank similarity for each rule (computed as a log frequency-weighted average over rule instances). This feature is a counterpart to rule reliability that is unsupervised in that it does not require class labels. We compute it by randomly drawing 200 lemma pairs for each rule from DERIVBASE (less if the rule has fewer instances). The pair-level feature is the difference between the rule's average rank similarity and the rank similarity for the current pair. It measures the rank of a pair relative to the rule's "baseline" rank and indicates how similar and dissimilar lemma pairs are compared to the rule average. In parallel to the structural features, values for complex rule paths are computed by minimum. Since the rank similarity is directional, we compute both hybrid features in two variants, one for each direction.[6]

[5]We compute this feature once only on simplex paths and once on all instances of the rule in DERIVBASE, trading reliability against noise.

[6]We also tested hybrid features based on raw cosine; however, this yielded worse results than the rank-based hybrid features.

Validation method	Precision	Recall	F_1	Accuracy
Majority baseline	72.6	**100**	84.1	72.6
Classifier, *only "cosine similarity" feature*	72.6	100	84.1	72.6
Classifier *only "similarity rank" feature*	80.3	90.3	85.0	76.8
Classifier, *only "rule identity" feature*	73.7	99.5	84.6	73.8
Classifier, *hybrid group*	80.4	95.3	87.2	79.7
Classifier, *distributional group*	80.5	96.6	87.8	80.5
Classifier, *structural group*	82.7	93.1	87.6	80.9
Classifier, *hybrid + distributional groups*	82.6	93.3	87.6	80.9
Classifier, *hybrid + structural groups*	84.9	93.7	89.1	83.4
Classifier, *distributional + structural groups*	85.3	94.6	89.7	84.3
Classifier, *all features*	**86.2**	93.9	**89.9**	**84.7**

Table 5: Accuracy, precision, recall, and F_1 on the test portion of the DERIVBASE 1.4 extended sample.

5.3 Results and Discussion

We applied the trained classifier to the test portion of the DERIVBASE 1.4 extended sample (cf. Section 2.2). Table 5 summarizes precision, recall, and F_1-score of the classifier for various combinations of features and feature groups. Recall that since our motivation is semantic validation, i.e., the removal of false positives, we are in particular interested in improving the *precision* of our predictions. We test significance of F_1 differences among models with bootstrap resampling (Efron and Tibshirani, 1993).

Our baseline is the majority class in the sample, **R**. Due to the sample's skewed class distribution (cf. Table 2), the frequency baseline is quite high (precision 72.6, F_1-score 84.1). We next consider the three most prominent individual features: Distributional similarity measured as cosine, distributional similarity measured as similarity rank, and rule identity. As expected from our analyses, the cosine similarity on its own is not reliable; in fact, it performs at baseline level. The rank-based similarity already leads to a considerable gain (precision +7.7%), but only a slight F_1-score increase of 0.9% that is not statistically significant at p=0.05. These results provide good empirical evidence for Hypothesis 1' (Section 3.4) and underscore that 1' is a more accurate statement than Hypothesis 1 (Section 2.3). On the structural side, rule identity alone improves the precision by 1.1%, with an F_1-score increase in 0.5% (again not significant).

We now proceed to complete feature groups, all of which perform at least 3% F_1-score better than the baseline, proving that the features within these groups are complementary. The hybrid feature group is the worst among the three. The distributional feature group is able to improve only slightly over the individual rank-based similarity feature in precision (80.5 vs. 80.3), but gains 6.3% in recall. This is sufficient for a significant improvement in F_1 (+3.7%, significant at p=0.01). The structural feature group performs surprisingly well, given that these features are very simple and most are computed only on the relatively small training set. It yields by far the highest precision (82.7), and its F_1-score is only slightly lower than the one of the distributional group (87.6 vs. 87.8). We take this as further evidence for the usefulness of structural information, as expressed by Hypothesis 2 (cf. Section 2.3).

Ultimately, all three feature groups turn out to be complementary. We obtain an improvement in F_1-score for two out of the three feature group combinations, and a clear improvement in precision in all cases. Finally, the best overall result is shown by the combination of all three feature groups. It attains an F_1-score of 89.9, an improvement of 5.8% over the baseline and 2.1% over the best feature group (both differences significant at p=0.01). Crucially, this model gains over 13% in precision while losing only 6% of recall compared to the baseline. This corresponds to a reduction of false positives in the sample by about half (from 27% to 14%) while the true positives were reduced only by 5% (from 73% to 68%).

Table 6 shows a breakdown of the predictions by the best model in terms of the five gold standard classes

	R	M	N	L	C	total
Gold annotation	554	81	81	45	2	763
Classified as **R**	520	36	16	29	2	603
Classified as **not R**	34	45	65	16	0	160

Table 6: Predictions on the test set of the *all features* Classifier per annotation class.

(**R**, **M**, **N**, **L**, **C**). Ignoring compounds (**C**), of which there are too few cases to analyze, we first find that the classifier achieves a high **R** recall. It is also very good in filtering out unrelated cases (**N**), of which it discards around 80%. The model recognizes morphologically but not semantically related word pairs (**M**) fairly well and manages to remove more than half of these. It has the hardest time with lemmatization errors (**L**), of which only about 35% were removed. However, this is not surprising: Lemmatization errors do not form a coherent category that would be easy to retrieve with the kinds of features that we have developed. We believe that such errors should be handled in an earlier stage, i.e., during preprocessing.

6 Related Work

Given that many derivational lexicons were only developed in recent years, we are only aware of one study (Jacquemin, 2010) that semantically validates the output of an existing derivational lexicon (Gaussier, 1999) to apply it to Question Answering. In contrast to our study, it requires elaborate dictionary information to look up which derivations are permitted for a specific lemma, as well as word sense disambiguation to determine the meaning of ambiguous words in context. Other related work comes from two areas: unsupervised morphology induction and semantic clustering.

Unsupervised morphology induction is concerned with the automatic identification of morphological relations (cf. Hammarström and Borin (2011) for an overview). Most approaches in this area do not differentiate between the inflectional and derivational level of morphology (Gaussier (1999) is an exception) and restrict themselves to the string level. Only a small number of studies (Schone and Jurafsky, 2000; Baroni et al., 2002) take distributional information into account.

Semantic clustering is the task of inducing semantic classes from (broadly speaking) distributional information (Turney and Pantel, 2010; im Walde, 2006). Boleda et al. (2012) include derivational properties in their feature set to learn Catalan adjective classes. However, the input to such studies is almost always a set of words from the same part of speech with no prior morphological constraints, while our input lemmas are morphologically preselected (via derivational rules), are often extremely infrequent, and exhibit systematical variation in parts of speech. To our knowledge, this challenging situation has not been addressed in previous studies.

Recent work has also considered the opposite problem, namely using derivational morphology for improving distributional similarity predictions. Luong et al. (2013) use recursive neural networks to learn representations of morphologically complex words and demonstrate the usefulness of their approach on word similarity tasks across different datasets. Similarly, Lazaridou et al. (2013) improve the word representations of derivationally related words by composing vector space representations of stems and derivational suffixes.

7 Conclusions

Almost all existing derivational lexicons do not distinguish between only morphologically related words on one hand and words that are both morphologically and semantically related words on the other hand. In this paper, we have addressed the task of recovering this distinction and called it *semantic validation*. We have used DERIVBASE, a German derivation lexicon, as the basis of our investigation.

We have made two contributions: (a) providing a detailed analysis of the types of information available for this task (distributional similarity as well as structural information about derivation rules) and the problems associated with each information type; and (b) training a machine learning classifier on linguistically

motivated features. The classifier, although not perfect, can substantially improve the precision of the word pairs in DERIVBASE and thus help to filter the derivational families in the lexicon. We are making this semantic validation information available in the DERIVBASE lexicon by attaching a probability for the class **R** to each lemma pair (see footnote 1 for the DERIVBASE URL).

The approach that we have described should transfer straightforwardly to other derivational lexicons and other languages on the conceptual level. The practical requirements are an appropriate corpus (for the distributional features) and derivational rule information (for the structural features).

There are two clear directions for future work. First, we plan to broaden our attention from word pairs to clusters and use the relatedness probabilities to cluster the derivational families in DERIVBASE into semantically coherent subfamilies. Second, we will demonstrate the impact of semantic validation on applications of derivational knowledge such as derivation-driven smoothing of distributional models (Padó et al., 2013).

Acknowledgments. We gratefully acknowledge partial funding by the European Commission (project EXCITEMENT (FP7 ICT-287923), first and second authors) as well as the Croatian Science Foundation (project 02.03/162: "Derivational Semantic Models for Information Retrieval", third author). We thank the reviewers for their valuable feedback.

References

Harald R. Baayen, Richard Piepenbrock, and Leon Gulikers. 1996. *The CELEX Lexical Database. Release 2. LDC96L14*. Linguistic Data Consortium, University of Pennsylvania, Philadelphia, Pennsylvania.

Marco Baroni, Johannes Matiasek, and Harald Trost. 2002. Unsupervised Discovery of Morphologically Related Words Based on Orthographic and Semantic Similarity. *Computing Research Repository*, cs.CL/0205006.

Edwin L. Battistella. 1996. *The Logic of Markedness*. Oxford University Press.

Orhan Bilgin, Ozlem Çetinoğlu, and Kemal Oflazer. 2004. Morphosemantic relations in and across Wordnets. In *Proceedings of the Global Wordnet Conference*, pages 60–66, Brno, Czech Republic.

Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the International Conference on Computational Linguistics*, pages 89–97, Beijing, China.

Gemma Boleda, Sabine Schulte im Walde, and Toni Badia. 2012. Modeling Regular Polysemy: A Study on the Semantic Classification of Catalan Adjectives. *Computational Linguistics*, 38(3):575–616.

John A. Bullinaria and Joe P. Levy. 2007. Extracting Semantic Representations from Word Co-occurrence Statistics: A Computational Study. *Behavior Research Methods*, 39(3):510–526.

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

Bradley Efron and Robert J. Tibshirani. 1993. *An Introduction to the Bootstrap*. Chapman and Hall, New York.

Stefan Evert. 2005. *The Statistics of Word Cooccurrences Word Pairs and Collocations*. Ph.D. thesis, University of Stuttgart.

Gertrud Faaß, Ulrich Heid, and Helmut Schmid. 2010. Design and Application of a Gold Standard for Morphological Analysis: SMOR in Validation. In *Proceedings of the Conference on Language Resources and Evaluation*, pages 803–810, Valletta, Malta.

Christiane Fellbaum, Anne Osherson, and Peter Clark. 2009. Putting semantics into WordNet's "morphosemantic" links. In *Proceedings of Human Language Technology. Challenges of the Information Society*, pages 350–358, Poznań, Poland.

Éric Gaussier. 1999. Unsupervised learning of derivational morphology from inflectional lexicons. In *ACL Workshop Proceedings on Unsupervised Learning in Natural Language Processing*, pages 24–30, College Park, Maryland.

Nizar Habash and Bonnie Dorr. 2003. A categorial variation database for English. In *Proceedings of the North American Association for Computational Linguistics*, pages 96–102, Edmonton, Canada.

Harald Hammarström and Lars Borin. 2011. Unsupervised Learning of Morphology. *Computational Linguistics*, 37(2):309–350.

Mary Hare, Michael Jones, Caroline Thomson, Sarah Kelly, and Ken McRae. 2009. Activating Event Knowledge. *Cognition*, 111(2):151–167.

Sabine Schulte im Walde. 2006. Experiments on the Automatic Induction of German Semantic Verb Classes. *Computational Linguistics*, 32(2):159–194.

Bernard Jacquemin. 2010. A derivational rephrasing experiment for question answering. In *Proceedings of the Conference on Language Resources and Evaluation*, pages 2380–2387, Valletta, Malta.

Lauri Karttunen and Kenneth R. Beesley. 2005. Twenty-five Years of Finite-state Morphology. In *Inquiries into Words, Constraints and Contexts. Festschrift for Kimmo Koskenniemi on his 60th Birthday*, pages 71–83. CSLI Publications, Stanford, California.

Gabriella Lapesa and Stefan Evert. 2013. Evaluating neighbor rank and distance measures as predictors of semantic priming. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 66–74, Sofia, Bulgaria.

Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. 2013. Compositional-ly derived representations of morphologically complex words in distributional semantics. In *Proceedings of the Association for Computational Linguistics*, pages 1517–1526, Sofia, Bulgaria.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the International Conference on Machine Learning*, ICML, pages 296–304, San Francisco, California.

Minh-Thang Luong, Richard Socher, and Christopher D. Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Conference on Natural Language Learning*, pages 104–113, Sofia, Bulgaria.

Sebastian Padó, Jan Šnajder, and Britta Zeller. 2013. Derivational smoothing for syntactic distributional semantics. In *Proceedings of the Association for Computational Linguistics*, pages 731–735, Sofia, Bulgaria.

Karel Pala and Dana Hlaváčková. 2007. Derivational relations in Czech wordnet. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies*, pages 75–81, Prague, Czech Republic.

Martin Porter. 1980. An Algorithm for Suffix Stripping. *Program*, 14(3):130–137.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on Natural Language Processing*, Manchester, UK.

Patrick Schone and Daniel Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. In *Proceedings of the Conference on Natural Language Learning*, pages 67–72. Lisbon, Portugal.

Eyal Shnarch, Jacob Goldberger, and Ido Dagan. 2011. A probabilistic modeling framework for lexical entailment. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 558–563, Portland, Oregon.

Jan Šnajder and Bojana Dalbelo Bašić. 2010. A computational model of Croatian derivational morphology. In *Proceedings of the International Conference on Formal Approaches to South Slavic and Balkan Languages*, pages 109–118, Dubrovnik, Croatia.

Pavol Štekauer and Rochelle Lieber, editors. 2005. *Handbook of Word-Formation*, volume 64 of *Studies in Natural Language and Linguistic Theory*. Springer.

Peter D. Turney and Patrick Pantel. 2010. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Britta Zeller, Jan Šnajder, and Sebastian Padó. 2013. DErivBase: Inducing and evaluating a derivational morphology resource for German. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1201–1211, Sofia, Bulgaria.

Detecting Learner Errors in the Choice of Content Words Using Compositional Distributional Semantics

Ekaterina Kochmar
Computer Laboratory
University of Cambridge
ek358@cl.cam.ac.uk

Ted Briscoe
Computer Laboratory
University of Cambridge
ejb@cl.cam.ac.uk

Abstract

We describe a novel approach to error detection in adjective–noun combinations. We present and release a new dataset of annotated errors where the examples are extracted from learner texts and annotated with error types. We show how compositional distributional semantic approaches can be applied to discriminate between correct and incorrect word combinations from learner data. Finally, we show how the output of the compositional distributional semantic models can be used as features in a classifier yielding good precision and accuracy.

1 Introduction

The task of error detection and correction (henceforth, EDC) in non-native writing in English has been a focus of research in recent years. However, usually research in this area focuses on EDC in the use of function words, such as articles or prepositions (Leacock et al., 2010; Dale et al., 2012), while much less attention has been paid to errors in the choice of content words.

Errors in function words are some of the most common error types in learner writing (Dalgish, 1985; Leacock et al., 2010), so it is important for any EDC system to be able to deal with such errors. Certain properties of these errors facilitate their detection and correction. As function words belong to closed classes, the set of possible corrections is limited by the size of the function word set. Since errors in function words are systematic and highly recurrent, in practice, each article or preposition has an even smaller number of appropriate alternatives. We illustrate this point with the following examples on (1) article and (2) preposition errors:

(1) I am *0*/a* student.

(2) Last October, I came *in*/to* Tokyo.

In (1) an EDC system would consider {*a, an, the*} as possible corrections for the missing article. To correct the preposition *in* in (2), an EDC system would consider the most frequent prepositions {*on, from, for, of, about, to, at, with, by*}, among which *at* or *to* would have a higher chance to be appropriate corrections as these are most often confused with *in*. Confusion sets can be learnt from learner texts, and probabilities can be set up according to the distribution of the confusions (Rozovskaya and Roth, 2011).

EDC is usually cast as a multi-class classification task, with the number of classes equal to the number of target corrections. Detection and correction can occur simultaneously: an error is detected when an EDC system suggests using a word different from the one originally used by the learner, and the suggested word can be used as a correction. Each occurrence of a function word is represented with a feature vector, where features are derived from the surrounding context. This is usually highly informative for function words: for example, a context of *I am* and *student* or a similar noun requires the use of an indefinite article, while the only correct preposition to relate a verb of movement like *come* to a locative like *Tokyo* is *to*.

In this work, however, we focus on errors in the choice of content words, which have received much less attention in spite of being the third most frequent error type in learner writing (Leacock et al., 2010). Errors in content words are more challenging than errors in function words, since the number of possible

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1740–1751, Dublin, Ireland, August 23-29 2014.

confusions and corrections cannot be reduced to a finite set. For example, consider incorrect choice of adjectives in the following sentences extracted from learner data:

(3) A *big*/great* damage has been made to the environment.

(4) I have tried a rock'n'roll dance and a *classic*/classical* dance already.

The confusion in (3) is caused by semantic similarity of the adjectives *big* and *great*, while in (4) it is due to similarity in form between *classic* and *classical*. It is much harder to cast the EDC in content words as multi-class classification, unless we consider the full set of English adjectives as possible classes. The surrounding contexts are much sparser and less informative, and in addition to that, often contain further errors. In this work, we address error detection and focus on adjective-noun combinations (ANs), which are representative of the more general task of EDC in content word combinations and are a frequent error type in learner text.

We have created a dataset of ANs, where the combinations are extracted from learner texts and manually error-coded using a novel annotation scheme. This scheme is motivated by observations about typical learner confusions in the choice of adjectives and nouns – for example, semantically-related or form-related confusions. Since errors in content words are related to semantics, we derive semantically-motivated features through models of compositional distributional semantics and use these features for error detection. We treat error detection as a binary classification task, following the usual convention in EDC.

The original contributions of this paper are that we:

- present and release an error-annotated AN dataset extracted from learner data;
- show how compositional distributional semantic models can be applied to detect semantic anomalies in this dataset;
- demonstrate that the output of these models can be used to derive features for error detection in AN combinations.

2 Previous work

2.1 Error Detection in Content Words

Previous work on EDC for content words has either focused on correction alone assuming that errors are already detected (Liu et al., 2009; Dahlmeier and Ng, 2011), or has reformulated the task as *writing improvement* (Shei and Pain, 2000; Wible et al., 2003; Chang et al., 2008; Futagi et al., 2008; Park et al., 2008; Yi et al., 2008; Östling and Knutsson, 2009).

In the first case, the task is reduced to the search for the most suitable correction among the alternatives typically composed of synonyms, homophones or L1-related paraphrases (Dahlmeier and Ng, 2011), while the more challenging error detection step is omitted. In the second case, error detection is integrated into suggestion of alternatives and their comparison to the originally used word combination according to some metric of collocational strength. Such approaches aim to improve the fluency of non-native texts by correcting erroneous idioms or collocations, where low frequency or low collocational strength clearly signifies an error.

These approaches might be useful for correcting collocations, but they are less suitable for error detection in free word combinations. As they compare original word combinations to their alternatives using corpus statistics, they are not applicable to unseen word combinations, while learner texts contain many previously unseen combinations, not all of which are errors. Moreover, some word combinations may be correct even though less fluent than some of their alternatives. For example, *appropriate concern*, though it is correct, would have lower collocational strength than its alternative *proper concern*, and would, according to this approach, be tagged as an error. From the educational point of view, tagging an acceptable combination as an error is misleading for language learners and should be avoided.

We implement a baseline model inspired by such comparison-based approaches and demonstrate that it cannot be usefully applied to error detection in content word combinations. Then we present an approach that is capable of dealing with unseen data and does not rely on direct corpus-based comparison.

2.2 Semantic Anomaly Detection

Learner errors in content words often result from a semantic mismatch between the chosen words. A similar problem of semantic anomaly detection in content word combinations has been addressed with compositional distributional semantic models.

These models are based on distributional representations for words which are then composed to derive phrase representations. They rely on the assumption that a word meaning can be approximated by its distribution across its contexts of use. Words are represented as vectors in a high-dimensional space with each dimension encoding a word's co-occurrence with one of its contextual elements. Distributional models are less suitable for representing content word combinations directly since these will be very sparse and will often remain unattested even in an extremely large corpus.

A promising solution is provided by compositional distributional semantic models, which combine distributional vectors for the component words using some function over such vectors. Compositional distributional semantic representations have been previously used to detect semantic anomaly in AN combinations (Vecchi et al., 2011). Vecchi *et al.* have applied the *additive* and *multiplicative* models of Mitchell and Lapata (2008) and *adjective-specific linear maps* of Baroni and Zamparelli (2010) to a set of corpus-unattested ANs. They show that there is a distinguishable difference in the compositional semantic representations for the semantically acceptable and anomalous combinations, suggesting that compositional distributional models can be used to detect semantic anomaly without relying directly on corpus statistics.

Kochmar and Briscoe (2013) have applied the same models of semantic composition to distinguish between correct and incorrect ANs extracted from learner texts. Their results support the assumption that there is a distinguishable difference between the composite vectors for the correct and incorrect ANs, but they did not address the question of how to integrate these semantic models into an error detection system.

Recent work by Lazaridou *et al.* (2013) has shown that measures used for quantifying the degree of semantic anomaly in phrases derived from their compositional distributional semantic representations can be used as features by a classifier to help resolve syntactic ambiguities.

Our goals are to test, using a new and larger AN dataset, whether semantic models can distinguish between correct and incorrect AN combinations, which cannot be dealt with using simpler error detection approaches, and to implement an error detection system using these semantically-based features.

3 Data Annotation

We present and release a dataset of AN combinations which, on the one hand, exemplify the typical errors committed by language learners in the choice of content words within such combinations, and, on the other hand, are challenging for an EDC system.

For that, we examined the publicly available CLC-FCE dataset (Yannakoudakis et al., 2011), used the error annotation (Nicholls, 2003), and analysed the typical errors in AN combinations committed by language learners. We have compiled a list of 61 adjectives that are most problematic for learners.

Most typically, learners confuse semantically related words: for example, they are unable to distinguish between synonyms, near-synonyms or co-hyponyms and choose an appropriate one from the set. Our list of adjectives contains some frequent ones that are confused with each other due to their similarity in meaning. For example, the adjectives within the set {*big, large, great*} are frequently confused with each other as in:

(5) *big*/large* quantity

(6) *big*/great* importance

Another common source of error related to the high-frequency adjectives involves using them instead of more specific ones: in such cases, learners are unable to distinguish between the more specific terms and they choose the most frequent adjective, usually encompassing a variety of related meanings, to represent the whole class of similar words. For example, adjectives *big* and *large* encompass a variety of meanings including those of *high*, *wide* or *broad*. As learners often lack intuitions about which of these

more specific adjectives should be chosen, they use the ones with more general meaning. This results in errors like:

(7) *big*/long* history

(8) *bigger*/wider* variety

(9) *greatest*/highest* revenue

(10) *large*/broad* knowledge

The reverse of this – an incorrect selection of a more specific term instead of the more general one – also leads to learner errors.

Form-related confusions represent another typical source of learner errors, and we have included pairs of adjectives such as *classic* and *classical*, *economic* and *economical* and the like in our dataset:

(11) *classic*/classical* dance

(12) *economical*/economic* crisis

Using this set of 61 adjectives, we have extracted AN combinations from the Cambridge Learner Corpus (CLC),[1] a large corpus of texts produced by English language learners, sitting Cambridge Assessment's examinations.[2] We have focused on AN combinations previously unseen in a native English corpus, as we hypothesise that they would have a higher chance of containing an error. Such combinations are more challenging for EDC algorithms since:

- these ANs cannot be effectively handled with simple comparison-based approaches like the ones overviewed in section 2.1;
- language learners are creative in their writing, so there is a substantial number of such previously unseen combinations;
- as no corpus could cover all possible acceptable content word combinations in language, the fact that these combinations are not seen in the corpus cannot be used as definitive evidence of incorrectness.

To summarise, it is important for an EDC algorithm to handle such combinations, but their absence in a native corpus of English makes it impossible to rely on simpler approaches and suggests that semantic analysis of such combinations would be more effective. In our research, we used the British National Corpus (BNC)[3] to select the corpus-unattested combinations.

We have compiled a set of 798 AN combinations.[4] An annotation scheme has been devised to annotate these examples as correct or incorrect, and for the incorrect combinations, to identify the locus of error (adjective, noun or both) and the type of confusion (incorrect synonym, form-related word, or non-related word). The most appropriate corrections are included in the dataset.

We also distinguish between *out-of-context* (*OOC*) and *in-context* (*IC*) annotation. The motivation behind this distinction is as follows: some combinations may appear to be correct when considered out of their original context of use, because there might be other contexts where the same combination would be appropriate. For example, *classic dance* is annotated as correct out of context because one could imagine using it in a context where it would denote some typical dance like:

(13) They performed a *classic Ceilidh dance*.

However, in practice, the AN *classical dance* is used much more frequently, and *classic dance* is most often errorful in context, as in (4) above.

Some ANs in our dataset are represented with more than one context of use, and in that case the *in-context* annotation can be conditioned on each context, or used to derive the most typical annotation for the AN. Both types of information are useful, as EDC systems which make use of the surrounding context should rely on the annotation in each particular context of use and, for example, be able to detect

[1] http://www.cup.cam.ac.uk/gb/elt/catalogue/subject/custom/item3646603/Cambridge-International-Corpus-Cambridge-Learner-Corpus/

[2] http://www.cambridgeenglish.org

[3] http://www.natcorp.ox.ac.uk/

[4] This dataset is released and publicly-available at http://www.ilexir.com/

Type	Cor.	Incor.	LB	UB
OOC	633	165	0.7932	0.8650
IC	394	404	0.5063	0.7467

Table 1: Distribution of correct (cor.) and incorrect (incor.) ANs in the dataset.

that *classic dance* is correct in one specific context, while in others it is incorrect. EDC systems that do not make use of the context can simply rely on the most frequent *in-context* annotation and detect that *classic dance* is typically an error in learner writing.

To create the two-level annotation, the annotators were first presented with an AN combination and asked to tag each word as correct or incorrect depending on whether they can think of some appropriate contexts of use for it. Next, the same combination was presented in its context of use from the CLC and the annotators were asked to annotate it with respect to its context.

The dataset was primarily annotated by a professional linguist. To ensure that the annotation scheme is clear and efficient, the dataset was split into 100 and 698 ANs, and the 100 ANs were first annotated by the same professional annotator and three other annotators. We have measured the inter-annotator agreement for the two levels of annotation using the mean values for the observed agreement within each pair of annotators, as well as mean Cohen's *kappa* value (Cohen, 1960). In Table 1 we report the mean inter-annotator agreement for the correct versus incorrect combinations at the two annotation levels, which represents the upper bound (*UB*) in our experiments. We have obtained the mean *kappa* values of 0.65 and 0.49 at the two levels of annotation, which are interpreted as substantial and medium agreement between annotators and confirm that the annotation scheme is clear.[5] Table 1 presents the distribution of ANs and the majority class baseline which we further use as a lower bound (*LB*).

4 Semantic Models for Error Detection

We replicate the semantic approaches, which have previously shown promising results in detecting semantic anomaly and content word errors (Vecchi et al., 2011; Kochmar and Briscoe, 2013), and test their performance on our dataset of corpus-unattested correct and incorrect AN combinations.

4.1 Experimental Setting

We use the *additive* (*add*) and *multiplicative* (*mult*) models of Mitchell and Lapata (2008), and the *adjective-specific linear maps* (*alm*) of Baroni and Zamparelli (2010).

The first two models derive the composite phrase vector through addition and multiplication of the components of the word vectors. These models have a clear mathematical interpretation and require no training. Their principal weakness is that they are symmetric, and fail to represent the difference in grammatical function of the component words. The *alm* model provides a theoretically more appropriate way of representing ANs based on this asymmetry: nouns are represented by their distributional vectors, while attributive adjectives are functions mapping from noun meanings to a composite noun-like vector for the ANs. Adjectives are represented as weight matrices which are learned from corpus-attested examples of noun–AN mappings, and composition is defined by matrix-by-vector multiplication.

We use the experimental setting previously described (Vecchi et al., 2011; Kochmar and Briscoe, 2013) and populate the semantic space with the constituent nouns and adjectives from the test ANs, frequent nouns and adjectives from the BNC and the AN combinations containing these frequent words. We use about $8K$ nouns, $4K$ adjectives and $64K$ ANs following Kochmar and Briscoe (2013). The semantic space is represented by a matrix encoding word co-occurrences, where the rows represent the $76K$ elements mentioned above, and the columns represent a selected set of $10K$ context elements. The $10K$ context elements include the most frequent nouns, adjectives and verbs from the corpus. The word co-occurrence counts are estimated using the BNC. The corpora have been lemmatized, tagged and parsed with the RASP system (Briscoe et al., 2006; Andersen et al., 2008; Yannakoudakis et al., 2011), and all statistics are extracted at the lemma level.

[5] Further details of the annotation experiment are described in the dataset release.

We transform the raw sentence-internal co-occurrence counts into Local Mutual Information scores (Baroni and Zamparelli, 2010; Evert, 2005), and perform dimensionality reduction applying Singular Value Decomposition to the noun and adjective matrix rows, projecting the AN rows onto the same reduced space following Baroni and Zamparelli (2010). The original $76K \times 10K$ matrix is reduced to a $76K \times 300$ matrix. This allows us to perform training and other calculations in the semantic space more efficiently.

The weight coefficients for the *alm* model are estimated with multivariate partial least squares regression using the RPLS package (Mevik and Wehrens, 2007). The weight matrix is learned for each adjective separately.

4.2 Semantic Cues

In previous work (Vecchi et al., 2011; Kochmar and Briscoe, 2013) several semantic measures for detecting semantic anomaly have been introduced. We reimplement these measures (1 to 8), but also test some additional measures (9 to 13) that we hypothesise can also help distinguish between correct and incorrect word combinations:

1. **Vector length (*VLen*)**: vectors for correct and incorrect combinations may differ with respect to their length, and the latter are expected to be shorter;
2. **Cosine to the input noun (*cosN*)**: the distance between the model-generated AN vector and the input noun vector is expected to be greater for the incorrect combinations, as the noun meaning is typically 'distorted';
3. **Cosine to the input adjective (*cosA*)**: analogical to *cosN* measure, the adjective meaning might be 'distorted' as well, especially as two of the composition functions are symmetric;
4. **Density of the neighbourhood populated by** 10 **nearest neighbours (*dens*)** is calculated as the average distance from the model-generated vector to the 10 nearest neighbours in the original semantic space, and is expected to be higher for the correct ANs;
5. **Density among the** 10 **nearest neighbours (*densAll*)** is a modification of *dens*, which is estimated as an average for the 11 density values calculated for each member within the set consisting of the AN vector and its 10 neighbours;
6. **Ranked density in close proximity (*Rdens*)** relies on the notion of *close proximity*, which is defined as a neighbourhood populated by some very close neighbours (for example, within a distance of ≥ 0.8). It is calculated as: $RDens = \sum_{i=1}^{N} rank_i distance_i$ with N being the total number of close neighbours within close proximity, each with its rank and distance;
7. **Number of neighbours within close proximity (*num*)** is used as another measure, and is assumed to be lower for incorrect combinations, which are expected to be more isolated in the semantic space;
8. **Overlap between the** 10 **nearest neighbours and constituent noun/adjective (*OverAN*)** assumes correct ANs should be surrounded by similar words and combinations. It is calculated as the proportion of the 10 nearest neighbours containing the same constituent words as in the tested ANs;
9. **Overlap between the** 10 **nearest neighbours and input noun (*OverN*)** is a variant of the *OverAN* with only the noun considered;
10. **Overlap between the** 10 **nearest neighbours and input adjective (*OverA*)** is a variant of the *OverAN* with only the adjective considered;
11. **Overlap between the** 10 **nearest neighbours for the AN and constituent noun/adjective (*NOverAN*)** assumes that correct ANs and their constituent words should be placed in similar neighbourhoods. It is calculated as the proportion of the common neighbours among the 10 nearest neighbours for the model-generated AN and the constituent words;

Metric	*add*	*mult*	*alm*
VLen	0.7589	0.7690	0.1676
cosN	0.1621	**0.0248**	**0.0227**
cosA	**0.0029**	0.4782	0.0921
dens	0.6731	0.1182	0.1024
densAll	0.4967	0.1026	0.1176
RDens	0.2786	0.8754	0.1970
num	0.3132	0.4673	0.3765
OverAN	0.8529	0.1622	0.2808
OverA	**0.0151**	0.6377	0.4886
OverN	**0.0138**	0.0764	0.4118
NOverAN	0.3941	0.6730	0.0858
NOverA	**0.0009**	0.3342	0.1575
NOverN	**0.0018**	0.1463	0.1497

Table 2: p values, *out-of-context* annotation

Metric	*add*	*mult*	*alm*
VLen	0.6675	**0.0027**	**0.0111**
cosN	**0.0417**	**0.0070**	0.1845
cosA	**0.00003**	0.1791	0.1442
dens	0.4756	0.7120	0.1278
densAll	0.2262	0.7139	0.5310
RDens	0.8934	0.8664	0.1985
num	0.7077	0.7415	0.4259
OverAN	0.1962	0.8635	0.5669
OverA	**0.00007**	0.7271	0.6229
OverN	**0.0017**	0.9680	0.7733
NOverAN	**0.0227**	0.3473	0.1587
NOverA	**0.000004**	0.3749	0.1576
NOverN	**0.0001**	0.6651	0.2610

Table 3: p values, *in-context* annotation

12. **Overlap between the** 10 **nearest neighbours for the AN and input noun (*NOverN*)** is a variant of the *NOverAN* with only the noun considered;

13. **Overlap between the** 10 **nearest neighbours for the AN and input adjective (*NOverA*)** is a variant of the *NOverAN* with only the adjective considered.

4.3 Results

We evaluate the models and report the results following the procedure that has been used before in Vecchi *et al.* (2011) and Kochmar and Briscoe (2013). For each model and semantic measure, we report the p value denoting statistical significance of the difference between the groups of correct and incorrect ANs. The statistical significance is reported at the $p<0.05$ level, and if a measure applied to the two groups of ANs shows statistically significant difference we interpret that as an ability of this measure to distinguish the correct ANs from the incorrect ones in general. The results for the out-of-context annotation are reported in Table 2, and those for the in-context annotation in Table 3.

The results show that the difference between the vector representations for the correct and incorrect AN combinations can be reliably detected with a number of the proposed measures. Measures which show statistically significant results with at least one model are marked in bold. These results also suggest that the values for the semantic measures can be used to derive discriminative features for a classifier.

5 Error Detection as Classification Task

5.1 Baseline System

We implement a simple comparison-based baseline system inspired by previous work on error detection in content words (see section 2.1). For every AN, we create a set of possible alternatives crossing the confusion set for the adjective with that for the noun, and compare the collocational strength of the original combination with that for each of the alternatives. If an alternative has higher collocational strength than the original combination, the original combination is tagged as an error and the alternative is chosen as a correction. Since semantically related confusions are a rich source of learner errors in content word combinations, we include adjective synonyms in the confusion set for an adjective, and noun synonyms and hyponyms in the confusion set for a noun. All synonyms and hyponyms are retrieved using WordNet 3.0 without word sense disambiguation.

We measure collocational strength using *normalized pointwise mutual information (npmi)* of the adjective a and noun n, which is defined as:

$$npmi(a,n) = \frac{pmi(a,n)}{-log[p(a,n)]} \quad (1) \qquad pmi(a,n) = log\frac{p(a,n)}{p(a)p(n)} \quad (2)$$

All probabilities are estimated from the BNC. This approach performs poorly on the unseen ANs in our dataset, since any alternative AN seen in the BNC would be preferred by this system over the original unseen AN. This ensures that less fluent (in this case, unseen) word combinations are substituted with more fluent (seen) ones. As a result, even though an original AN *important conversation* in our dataset is correct, it is still "corrected" by this system to *serious conversation*. At the same time, some incorrect combinations are not recognised if no appropriate alternative is found (e.g., **high shyness*). It shows that this approach lacks deeper semantic analysis and is also too dependent on the set of alternatives found for a word combination.

We measure *accuracy (acc)* as the proportion of true positives (*TP*) and true negatives (*TN*) to the total number of test items:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \quad (3)$$

Accuracy reflects how often an error detection system correctly identifies that an AN is correct or incorrect. We compare the results to the lower and upper bounds set as the majority class distribution and inter-annotator agreement, respectively (see section 3).

With this approach we get quite low accuracy of 0.3897 on the out-of-context annotation since most of the test items are correct out of context (LB=0.7932), and the baseline system overcorrects many of those. Accuracy of the baseline system on the in-context annotation is 0.5147, which is slightly above the lower bound of 0.5063. These results are used as a baseline and included in Table 4.

Type	Accuracy	Baseline	LB	UB
OOC	**0.8113** $\pm$ 0.0149	0.3897	0.7932	0.8650
IC	**0.6535** $\pm$ 0.0189	0.5147	0.5063	0.7467

Table 4: *Decision Tree* classification results

Type	P (correct)	P (incorrect)
OOC	0.8193	0.7500
IC	0.6241	0.6850

Table 5: Classification precision

5.2 Classification

We implement a supervised classifier which uses output of the semantic models as features. We have tested a number of classifier models but the best results so far have been obtained with the *Decision Tree* classifier using $NLTK$ (Bird et al., 2009). We assume that this classifier effectively learns the inter-dependencies between the features within the small feature set that we use in our experiments. We use feature binning where the whole range of feature values is divided into 10 bins according to the distribution of values for each feature. This feature representation technique combined with the classifier helps generalise over feature values, reducing feature space dimensionality. The order of the feature application to the data is determined by the classifier on the basis of the information gain for the features and their values.

We apply 5-fold cross-validation and report average accuracy over the folds. The 798 ANs are split into 5 subsets with 80% in each of the splits used for training and 20% for testing. We keep the AN error rate in the training and test sets, as well as for each adjective, approximately the same across the splits to avoid any bias. Error detection is cast as a binary classification task. The output of the semantic models is used to derive numeric features for the classifier. Most values are in the range of $[0, 1]$, and we apply normalisation to *VLen*, *RDens* and *num* which originally have a different range.

The full feature set contains 14 features, with 13 features derived from the semantic measures, and 1 feature representing adjective identity. We hypothesise that introduction of this feature might help classifier learn that, for example, an AN containing an adjective *classic* has a higher chance of being incorrect, as most of the ANs with this adjective in the learner data are incorrect and involve confusions with *classical*. We also hypothesise that it facilitates learning correlations between the adjective and other

feature values: it might be the case that ANs with an adjective adj_1, on the average, have higher *cosN* values than ANs with an adjective adj_2. This feature helps the classifier establish such dependencies between the adjective and the values of the semantic measures. For instance, in our data ANs with the adjective *true* have significantly higher cosine between AN vectors and vectors for their constituent nouns than ANs with the adjective *false*: this is in accordance with an intuition that, for example, *true happiness* is more similar to *happiness* than *false happiness* is.

The best results in our experiments have been obtained with the *mult* model. We have performed ablation tests incrementally removing features that did not improve classifier performance in order to find an optimal feature set. The best-performing feature set we found for the *mult* model on the out-of-context annotation uses *adjective*, *cosN* and *RDens* features, while for the in-context annotation the best-performing feature set found uses a combination of features including *adjective*, *VLen*, *densAll*, *NOverA*, *NOverN*, *RDens* and *num* features.

We note that the sets of best performing features in the classification experiments do not coincide with the semantic measures that showed the highest statistically significant difference (Tables 2 and 3). We conclude that although the p values reported in Tables 2 and 3 show that some semantic measures can distinguish one group of ANs from another on the basis of the statistically significant difference between the means of the two groups, when the measures are used as features for a classifier the results depend on how these features interact with each other as well as on their individual discriminativeness across the test dataset. For example, Figure 1 illustrates a small part of the decision tree constructed using the best performing feature set on the in-context annotation:

```
...
    if (num=1.0) == False:
    ...
        if (adjective is 'large') == True:
            if (0.002<=VLen<0.003) == False: return '1' [e.g., 'large jeans']
            if (0.002<=VLen<0.003) == True: return '-1' [e.g., 'large knowledge']
    if (num=1.0) == True: return '1'
...
```

Figure 1: *Decision Tree* classifier pseudocode.

Figure 1 shows how interaction of feature values for *num* and *VLen* in combination with the adjective identity feature can help classify the two ANs containing adjective *large* as correct (`1`) or incorrect (`-1`).

In Table 4 we report results for the *out-of-context* (*OOC*) and *in-context* (*IC*) annotation. The accuracy is reported with its mean $\pm$ standard deviation over the 5 data splits. We compare the *Decision Tree* classifier results to those obtained with the baseline system, as well as to the lower and upper bounds set as before (see section 3). The results show that a classifier that uses output of the semantic models as features outperforms the comparison-based baseline system by a large margin.

6 Discussion

In the previous section, we showed that a classifier that uses output of the semantic models as features outperforms the comparison-based baseline system and shows good accuracy. In this section, we analyse the classifier's performance in more detail.

We note that, from an educational point of view, it is important for an EDC system to have high precision. For example, it has been shown that grammatical error detection systems with high precision maximize learning effect, and that systems with high precision but lower recall are more useful in language learning than systems with high recall and lower precision (Nagata and Nakatani, 2010). This suggests that learners might be misled and confused if they are frequently notified by a system that something is an error when it is not.

Since precision is measured as the proportion of true positives (*TP*) to the sum of true positives and false positives (*FP*):

$$P = \frac{TP}{TP + FP} \tag{4}$$

an EDC system that achieves precision less than 0.5 is, in fact, misleading for language learners: for example, precision of less than 0.5 on the class of errors means that the system misidentifies correct use as an error more frequently than it correctly detects an error.

Our classifier achieves good precision values with respect to both out-of-context and in-context annotations, on correct and incorrect examples. Precision (P) values are reported in Table 5. As precision figures are higher than 0.5 in each case, it shows that the implemented error detection system would, on balance, help guide a learner to text regions in need of reformulation.

With respect to the out-of-context annotation, the error detection system has good precision and recall on correct examples ($P = 0.8193$, $R = 0.9762$). Precision on the incorrect examples is also high ($P = 0.7500$). This is a very encouraging result, suggesting the system would rarely misidentify an originally correct AN combination as an error.

For the in-context annotation, both precision and recall on correct and incorrect examples are quite high: $P = 0.6241$ and $R = 0.7169$ on the correct examples, and $P = 0.6850$ and $R = 0.5849$ on the incorrect examples.

Error analysis on the classifier's output shows that the majority of the incorrect examples misclassified as correct (*missed errors*) contain semantically-related confusions. It appears that the classifier relying on semantically-motivated features misses a number of cases where the original AN and its correction are semantically similar: for example, it misses the errors in *big*/great anger*, *biggest*/greatest painter* and *small*/short speech*. Since the ANs in these pairs are semantically similar, the features based on their semantic representations might not be discriminative enough. In contrast, the classifier is more effective in detecting errors in cases where the original AN and its correction are only similar in form, or not related to each other.

7 Conclusion

We have presented and released a dataset of learner errors in ANs, which has been extracted from learner texts and annotated with error types and corrections. The dataset contains examples not seen in a native corpus of English, and error annotation shows that a substantial number of such examples are correct. Error detection in this dataset is a challenging task, since absence of the ANs in a corpus of English cannot be used as definitive evidence of incorrectness. We have implemented a simple baseline system inspired by previous work on improving content word combinations and shown that such a system would not be effective for error detection in our dataset.

We have cast error detection as a binary classification task and implemented a supervised classifier that uses semantically-motivated features. The features are derived from the compositional distributional semantic representations of the AN combinations. We use a number of semantic measures that describe and distinguish between semantic representations for correct and incorrect combinations. We have introduced new semantic measures in addition to the ones used in previous work and show that they can be effectively applied to this task.

The best results in our experiments are obtained with a *Decision Tree* classifier, and we show that the resulting error detection system can identify errors with high precision and accuracy. We aim to extend this system to perform error correction on ANs, as well as error detection and correction on other types of content word combinations.

Acknowledgments

We are grateful to Cambridge English Language Assessment and Cambridge University Press for supporting this research and for granting us access to the CLC for research purposes. We would like to thank Øistein Andersen for providing us with the annotation tool, Diane Nicholls for undertaking the bulk of the annotation work, and Helen Yannakoudakis and the anonymous reviewers for their valuable comments.

References

Øistein Andersen, Julien Nioche, Ted Briscoe and John Carroll 2008. *The BNC parsed with RASP4UIMA*. In Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC), pp. 865–869.

Marco Baroni and Roberto Zamparelli 2010. *Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space*. In Proceedings of the EMNLP-2010, pp. 1183–1193.

Steven Bird, Ewan Klein, and Edward Loper 2009. *Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit*. O'Reilly Media.

Ted Briscoe, John Carroll and Rebecca Watson 2006. *The Second Release of the RASP System*. In Proceedings of the COLING/ACL-2006 Interactive Presentation Sessions, pp. 59–68.

Yu-Chia Chang, Jason S. Chang, Hao-Jan Chen and Hsien-Chin Liou 2008. *An automatic collocation writing assistant for Taiwanese EFL learners: A case of corpus-based NLP technology*. Computer Assisted Language Learning, 21(3), pp. 283–299.

Jacob Cohen 1960. *A coefficient of agreement for nominal scales*. Educational and Psychological Measurement, 20(1), pp. 37–46.

Robert Dale, Ilya Anisimoff and George Narroway 2012. *HOO 2012: A Report on the Preposition and Determiner Error Correction Shared Task*. In Proceedings of the 7th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 54–62.

Daniel Dahlmeier and Hwee Tou Ng 2011. *Correcting Semantic Collocation Errors with L1-induced Paraphrases*. In Proceedings of the EMNLP-2011, pp. 107–117.

Gerard M. Dalgish 1985. *Computer-assisted ESL research*. In CALICO Journal, 2(2), pp. 32–37.

Stefan Evert 2005. *The Statistics of Word Cooccurrences*. Dissertation, Stuttgart University.

Yoko Futagi, Paul Deane, Martin Chodorow and Joel Tetreault 2009. *A computational approach to detecting collocation errors in the writing of non-native speakers of English*. Computer Assisted Language Learning, 21(4), pp. 353–367.

Ekaterina Kochmar and Ted Briscoe 2013. *Capturing Anomalies in the Choice of Content Words in Compositional Distributional Semantic Space*. In Proceedings of the Recent Advances in Natural Language Processing (RANLP-2013).

Angeliki Lazaridou, Eva Maria Vecchi and Marco Baroni 2013. *Fish transporters and miracle homes: How compositional distributional semantics can help NP parsing*. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1908–1913.

Claudia Leacock, Martin Chodorow, Michael Gamon and Joel Tetreault 2010. *Automated Grammatical Error Detection for Language Learners*. Morgan and Claypool Publishers.

Anne Li-E Liu, David Wible and Nai-Lung Tsao 2009. *Automated suggestions for miscollocations*. In Proceedings of the 4th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 47–50.

Bjørn-Helge Mevik and Ron Wehrens 2007. *The pls package: Principal component and partial least squares regression in R*. Journal of Statistical Software, 18(2), pp. 1–24.

Jeff Mitchell and Mirella Lapata 2008. *Vector-based models of semantic composition*. In Proceedings of ACL, pp. 236–244.

Jeff Mitchell and Mirella Lapata 2010. *Composition in distributional models of semantics*. Cognitive Science, 34, pp. 1388–1429.

Ryo Nagata and Kazuhide Nakatani 2010. *Evaluating performance of grammatical error detection to maximize learning effect*. In Proceedings of COLING 2010, pp. 894–900.

Diane Nicholls 2003. *The Cambridge Learner Corpus: Error coding and analysis for lexicography and ELT*. In Proceedings of the Corpus Linguistics conference, pp. 572–581.

Robert Östling and Ola Knutsson 2009. *A corpus-based tool for helping writers with Swedish collocations.* In Proceedings of the Workshop on Extracting and Using Constructions in NLP, pp. 28–33.

Taehyun Park, Edward Lank, Pascal Poupart, Michael Terry 2008. *Is the sky pure today? AwkChecker: an assistive tool for detecting and correcting collocation errors*. In Proceedings of the 21st annual ACM symposium on User interface software and technology, pp. 121–130.

Alla Rozovskaya and Dan Roth 2011. *Algorithm Selection and Model Adaptation for ESL Correction Tasks*. InProceeding of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies – Volume 1, pp. 924–933.

Chi-Chiang Shei and Helen Pain 2000. *An ESL Writer's Collocation Aid*. Computer Assisted Language Learning, 13(2), pp. 167–182.

Eva Maria Vecchi, Marco Baroni and Roberto Zamparelli 2011. *(Linear) maps of the impossible: Capturing semantic anomalies in distributional space*. In Proceedings of the DISCO Workshop at ACL-2011, pp. 1–9.

David Wible, Chin-Hwa Kuo, Nai-Lung Tsao, Anne Liu and H.-L. Lin 2003. *Bootstrapping in a language-learning environment*. Journal of Computer Assisted Learning, 19(4), pp. 90–102.

Helen Yannakoudakis, Ted Briscoe and Ben Medlock 2011. *A New Dataset and Method for Automatically Grading ESOL Texts*. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 1, pp. 180–189.

Xing Yi, Jianfeng Gao and William B. Dolan 2008. *A Web-based English Proofing System for English as a Second Language Users*. In Proceedings of the third International Joint Conference on Natural Language Processing (IJCNLP), pp. 619–624.

A Novel Distributional Approach to Multilingual Conceptual Metaphor Recognition

Michael Mohler and **Bryan Rink** and **David Bracewell** and **Marc Tomlinson**
Language Computer Corp.
Richardson, Texas, USA
{michael,bryan,david,marc}@languagecomputer.com

Abstract

We present a novel approach to the problem of multilingual conceptual metaphor recognition. Our approach extends recent work in conceptual metaphor discovery by combining a complex methodology for facet-based concept induction with a distributional vector space model of linguistic and conceptual metaphor. In the evaluation of our system in English, Spanish, Russian, and Farsi, we experiment with several state-of-the-art vector space models and demonstrate a clear benefit to the fine-grained concept representation that forms the basis of our methodology for conceptual metaphor recognition.

1 Introduction

The role of metaphor in language has been defined by Lakoff et al. (1980; 1993) as a cognitive phenomenon which operates at the level of mental processes, whereby one concept or domain is viewed systematically in terms of another. For example, the phrase "to cure poverty" is a metaphor which subtly conveys a wide variety of information to the listener. In order to mentally process this phrase, we must first recognize that a metaphor is being used and that "cure" (as a medical term) is being used figuratively. Then, we assume some relationship between "poverty" and "things that can be medically cured" which leads to the conceptual mapping "POVERTY as DISEASE." This conceptual mapping enables the listener to transfer a variety of properties and associations between the two concepts, such as their association with a feeling of helplessness, the existence of sustained efforts to end them, the potential for them to spread, and their mutual relationship with ill-health and death. Therefore, by identifying the conceptual domains associated with this linguistic metaphor, we are able to reason about the target domain (POVERTY) using concepts and terms associated with the source domain (DISEASE).

Any natural language processing system capable of processing metaphor in text with human-level competence must, therefore, overcome three problems in sequence:

1. the identification of metaphorical expressions (also known as linguistic metaphors (LMs))
2. the discovery of a conceptual domain mapping or conceptual metaphor (CM) which consists of
   (a) the conceptual domain of the metaphor target (e.g., POVERTY); and
   (b) the conceptual domain of the metaphor source (e.g., DISEASE)
3. the real-world interpretation of the metaphorical text which uses the conceptual metaphor framework to transfer knowledge between the source and target domains.

While a significant amount of recent work has presented interesting and promising methodologies for multilingual LM identification (Shutova and Sun, 2013; Wilks et al., 2013; Strzalkowski et al., 2013), the work presented in this paper is focused on (2), the problem of multilingual CM recognition, which will be made to serve as the foundation for a more fine-grained interpretation of metaphor.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1752–1763, Dublin, Ireland, August 23-29 2014.

We cast the CM recognition process as a two-part methodology which (a) selects the target domain associated with a particular LM that has been detected; and (b) determines the source domain to which it should be mapped in order to produce a satisfactory interpretation. In this work, we assume that the target domains are known and belong to one of the following conceptual spaces: POVERTY, WEALTH, or TAXATION. Pragmatically speaking, research in CM recognition presupposes some methodology for LM identification, and to this end, we have employed an existing state-of-the-art LM identification system which has been developed to detect linguistic metaphors in four languages: English, Spanish, Russian, and Farsi (Bracewell et al., 2014).

In order to generate a CM which can serve as the basis for an interpretation of an LM, we have developed an approach that is based on the following hypotheses:

> CONCEPTUAL HYPOTHESIS: When an LM has been identified as a pair of lexical items that represent the source (e.g., "cure") and the target (e.g., "poverty"), we can generate a conceptual mapping by selecting the conceptual domains that are, *a priori*, the most likely for the source and target lexemes.[1]
>
> DISTRIBUTIONAL HYPOTHESIS: It is possible to decide which conceptual space better represents a given lexeme by
>
> 1. expanding the lexical space with additional terms (which we call "grammatical co-occurrents") that are strongly associated with the lexeme through grammatical relations such as AGENT, PATIENT, INSTRUMENT, and ATTRIBUTE;
> 2. using these lexical expansions to produce distributional vectors; and
> 3. uncovering the *selectional constraints* of particular domain facets by clustering the distributional vectors within a semantic space.
>
> DOMAIN HYPOTHESIS: The grammatical co-occurrents of the LM are themselves very likely to belong to the same conceptual domain as the lexeme (e.g., "cure patient", "cured of AIDS", and "doctor cured").
>
> MAPPING HYPOTHESIS: The semantic space representations of both the LM source and its grammatically associated terms can be used to produce mappings into a high dimensional space in which source domains are known to exist.

While other computational linguistics research in metaphor has made use of the CONCEPTUAL and DISTRIBUTIONAL hypotheses, to our knowledge the DOMAIN and MAPPING hypotheses have not yet been explored in combination with a distributional approach.

The remainder of this work is organized as follows. In Section 2, we discuss related work in the field of metaphor interpretation and unsupervised concept induction. In Section 3, we introduce the overall architecture of our CM recognition system. In Section 4, we describe our method for representing lexical items and conceptual metaphors in a distributional vector space. Then, in Section 5, we explain our methodology for creating and ranking clusters of LM co-occurrents which are then mapped to conceptual metaphors within our vector space. In Section 6, we describe our experimental setup and provide the results of our experiments. Finally, in Section 7 we present our conclusions.

2 Related Work

Research in metaphor processing can broadly be divided into two categories: metaphor identification and metaphor interpretation. Although some recent work on metaphor interpretation has skirted the issue of conceptual metaphor entirely by casting the problem of metaphor interpretation as an instance of lexical paraphrase (Shutova, 2010; Bollegala and Shutova, 2013) or textual entailment (Mohler et al., 2013), the mapping and modeling of conceptual metaphors has historically served as an important foundation for

[1] If the target domains are pre-selected, this hypotheses is reduced to selecting only the most likely source domain.

more robust interpretation of metaphor. Indeed, a significant amount of research in metaphor interpretation has been concentrated on the development of highly-structured, manually curated representations of both the CM source and CM target domains. Notable in this regard are the KARMA system (Feldman and Narayanan, 2004) which was designed to simulate neurological modeling of verbs – both abstract and metaphorical – and the ISOMETA system (Beust et al., 2003) which made use of differential tables of CM domain lexical items to drive their metaphor interpretation process. The CorMet system (Mason, 2004) sought to model conceptual metaphors by detecting individual source-target mappings that provide evidence for a known CM by quantifying the overlap between clusters of terms with a strong selectional preference to the most representative verbs within the source and target domains. After a manual inspection of the source/target cluster pairs across domains, the directionality and the systematicity of these underlying conceptual mappings were quantified in order to produce an overall confidence in the mapping. As part of their development of the Hamburg Metaphor Database (HMD), Reining and Lönneker-Rodman (2007) performed a a manual categorization of lexical items into conceptual source domains with a facet-level granularity and enriched their domains using a WordNet-based lexical expansion. In the same vein, Chung et al. (2005) chose to model source domains by expanding their lexical items by exploiting the links between WordNet glosses and the SUMO ontology.

In recent years, however, research has focused on automating the modeling and classification of conceptual metaphors as much as possible in order to encourage the scaling up of metaphor research in general. Veale and Hao (2008), as part of the Talking Points system, developed what they refer to as a Slipnet which defines linked chains of meaning that connect a source to a target through shared (or related) attributes and actions. As a step in this process, they combined WordNet relations with pragmatic relations extracted from text and clustered nouns according to their relation (and attribute) similarity in order to define a weak conceptual mapping within the clusters. In a similar way, Shutova et al. (2010), beginning with a seed set of noun/verb linguistic metaphor pairs, performed spectral clustering on a large set of nouns and verbs in order to predict metaphors which participate in the same conceptual metaphor mapping. In particular, she modeled verbs according to their subcategorization frames parameterized by a model of their selectional preferences, while nouns were modeled according to the verbs with which they frequently co-occurred in a dependency relation.

More recently, Gandy et al. (2013) approached the CM discovery problem as a set covering problem. For a given nominal target lexeme, they began by finding all facets (i.e., verbs/adjectives) that share a positive PMI with the target. Then, they would find the set of nouns that also have a positive PMI with those facets, compute their confidence in each association, and heuristically select pairs of concepts (defined as rooted WordNet synset trees) which subsume a large percentage of those nouns and cover a large portion of the overlapping facets. Similarly, Shutova and Sun (2013) detect conceptual mappings by performing hierarchical graph factorization clustering on a graph in which the vertices are defined to be nouns (i.e., concepts) and the edges are weighted using Jensen-Shannon Divergence. For a given input LM source, its likely conceptual metaphors are then discovered by determining its non-literal cluster membership. Finally, Strzalkowski et al. (2013) discovered terms (literal and metaphoric) which often co-occur with an LM source in a corpus and clustered those terms using WordNet and corpus statistics to form "ProtoSources" which could be further inspected to define CM source concepts.

Two vector-based approaches to concept representation are of particular interest in understanding the present work. In the first of these, Schütze (1998) described an approach to word sense identification using second-order co-occurrence vectors which were used to cluster first-order vectors of the in-context terms into senses.[2] Lin (1998), in developing a methodology for evaluating the quality of thesauri, defined a word vector space that moved beyond simple co-occurrence by integrating information about the relations between the word and its co-occurrents. In particular, a word's vector was defined by the number of times that word occurred within a set of (*word, relation, word*) tuples. Our DepVec space represents an extension to Lin's space insofar as we incorporate additional information about relational (i.e., selectional) preference.

[2]While context is critical in word sense disambiguation, we hasten to point out that one mark of metaphoricity is its disconnect from the surrounding literal context.

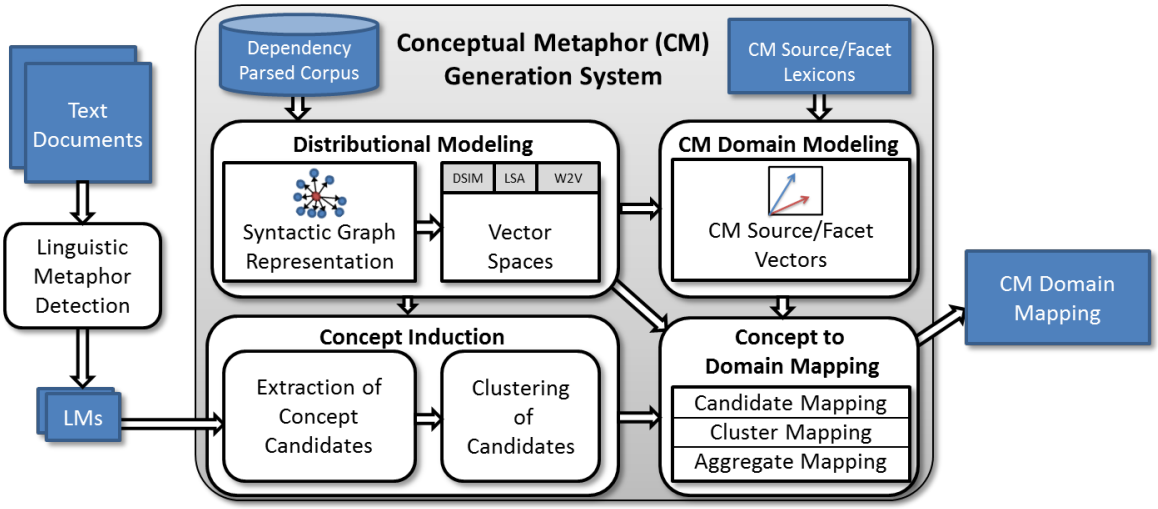


Figure 1: The architecture of our conceptual metaphor recognition system. This system takes a linguistic metaphor as input, induces potential concepts using vector-space clustering, and maps these clusters onto a conceptual metaphor domain.

3 A New Methodology for Conceptual Metaphor Recognition

Figure 1 shows the overall flow of our metaphor processing architecture. We begin with a set of documents gathered from a variety of online news-wire sources. These documents are fed to our state-of-the-art LM detection system which employs a binary logistic regression classifier using a variety of feature modules including imageability and concreteness estimation, topicality modeling, pattern matching, semantic categorization, selectional preference violation, and source/target vector space similarity. The methodology used in this system is beyond the scope of this work, but it is described in detail by Bracewell et al. (2014). The LMs provided by the detection system are validated by a group of native-language experts before being sent for CM recognition system for concept-level interpretation.

Once the LMs have been collected and validated, the CM recognition system begins by extracting, weighting, and clustering the common grammatical contexts of the LM source term. By grammatical context, we refer to the syntactic relations (along with their arguments) which have been found to frequently co-occur with the LM source term in open text. In order to model this grammatical context, we have syntactically parsed a wide collection of documents in each of our focus languages: English, Spanish, Russian, and Farsi. From these parsed documents, we have extracted the most common grammatical co-occurrents of each word in the corpus along with the relation that connects them and the number of times they are connected by that relation. For a given word, we refer to the set of its grammatical co-occurrents as the "concept candidates" associated with that word, as they represent potential concepts within the same conceptual domain as the given word (the DOMAIN HYPOTHESIS). For example, grammatical co-occurrents of the noun "battle" would include many WAR concepts such as "fought", "died in", "waged", "naval", and "losing".

Since a conceptual domain is made up of several interacting concepts, we perform a clustering over the grammatical co-occurrents to produce groups of terms which are likely to represent individual concepts within a domain. The clustering is performed within a high-dimensional, distributional vector space which we describe in Section 4. The clusters are then merged and aligned with a set of 51 predefined source concept domains (see Table 1) that have been found to occur frequently in conceptual metaphors about POVERTY, WEALTH, or TAXATION. For each of these known conceptual domains, we have amassed a collection of lexical items for the purpose of modeling the domains and aligning them to our automatically discovered domains. The collection of lexical items associated with each domain have been further partitioned into three to five *facets* which provide a more fine-grained representation of the domain. For instance, the conceptual domain of ABYSS as been subdivided into facets representing

Full Source Concept List					
A GOD	COMPETITION	ENSLAVEMENT	LIGHT	NATURAL PHYSICAL FORCE	PORTAL
A RIGHT	CONFINEMENT	FOOD	LOW POINT	OBESITY	RESOURCE
ABYSS	CRIME	FORCEFUL EXTRACTION	MACHINE	PARASITE	SCHISM
ACCIDENT	CROP	GAME	MAZE	PATHWAY	STRUGGLE
ADDICTION	DARKNESS	GEOGRAPHIC FEATURE	MEDICINE	PHYSICAL BURDEN	VERTICAL SCALE
ANIMAL	DESTROYER	GOAL DIRECTED	MONSTER	PHYSICAL HARM	VISION
BLOOD SYSTEM	DISEASE	HIGH POINT	MORAL DUTY	PHYSICAL LOCATION	
BODY OF WATER	ENABLER	HUMAN BODY	MOVEMENT	PHYSICAL OBJECT	
BUILDING	ENERGY	IMPURITY	MOVEMENT ON A VERTICAL SCALE	PLANT	
Sample Lexical Items					
ANIMAL	bite, bark, claw, bird, beaver		MEDICINE	dosage, prescription, heal	
ENSLAVEMENT	servant, oppression, ruler		STRUGGLE	enemy, fight, combat, attack	

Table 1: The 51 source conceptual domains along with some sample English lexical items for a subset of them.

DEPTH (e.g., "deep", "bottomless"), ENTRANCE (e.g., "plunged into", "falling into"), and EXIT (e.g., "climb out of").

3.1 Motivating Example

Table 2 shows a sample of the concept candidates associated with the word "cure" along with the relation that connects them. Our methodology for extracting these terms is discussed in Section 5.1.

nsubj	NIH, WHO, therapist, doctor, vaccine, drug, medicine, *chef, butcher*	prep_of	cancer, AIDS, HIV, malaria, influenza, seizures, allergies
dobj	cancer, polio, Goji Berries, man, genetic defects, aging, infant, woman, depression, *meat, fish, garlic*	prep_by	bone marrow transplant, spleen cells, acupuncture, *smoking, salting,* doxycycline, *drying, burying, dipping*
prep_without	surgery, operation, suppuration, *salt* chemotherapy, injections	prep_to^{-1}	need, project, *brine, mineral,* coalition, run, walk, *salt, nitrite*
prep_in	mice, children, baby, *spices, salt,* monkeys, drug trial, breakthrough, *brine, smokehouse, basement, fridge*	prep_for	*grinding, smoking, voyages, lox, transportation, preservation, jerky, sausages, bacon, sale*

Table 2: Terms that are frequently a part of the grammatical context of "cure" along with their associated relations

It is clear from the concept candidates shown that there are at least two coarse-grained senses of "cure" present – corresponding to the domains of MEDICINE and FOOD. Table 3 shows a sample result of clustering these concept candidates. These clusters are organized according to their domain with MEDICINE-related clusters in the left grouping, FOOD-related clusters in the top-right grouping, and clusters not strongly related to either domain in the bottom-right grouping. Each row of the table represents a single cluster. In addition, it can be observed that these clusters correspond to particular semantic facets of the conceptual domain. For instance, there is a cluster that defines "procedures which result in medical cures" ("acupuncture", "surgery", "operation", etc.), one that defines "individuals who cure food products" ("chef", "butcher"), and one that defines "diseases that can (potentially) be cured" ("cancer", "polio", "AIDS", etc.). Our methodology for automatically inducing such clusters is described in Section 5.2.

Once the clusters have been identified, they can be used to define a mapping from the original LM ("cure") onto a pre-defined set of CM source domains (the MAPPING HYPOTHESIS). In particular, individual concept candidates are mapped to CM domains by calculating the distance between the candidate and one or more vectors representing each domain in a high-dimensional distributional vector space.

4 Distributional Representations

Our method for identifying conceptual metaphor domains relies on determining when multiple words should be grouped as belonging to the same conceptual class (the DISTRIBUTIONAL HYPOTHESIS). Previous work in semantic similarity has shown two types of approaches to work well: (a) hand-coded knowledge such as WordNet or SUMO, and (b) distributional approaches which rely on statistics of

NIH, WHO, therapist, doctor	chef, butcher
vaccine, drug, medicine, doxycycline	project, coalition
spleen cells, bone marrow transplant	meat, fish, sausages, jerky, bacon, lox
acupuncture, surgery, operation	garlic, Goji Berries
chemotherapy, injections, suppuration	smoking, salting, drying, dipping
HIV, malaria, influenza	burying
cancer, polio, AIDS	salt, brine, spices, nitrite, mineral
genetic defects, aging, depression	smokehouse, basement, fridge
seizures, allergies	run, walk
drug trial, breakthrough	voyages, transportation
infant, man, woman, children, baby	mice, monkeys

Table 3: Terms from Table 2 grouped into conceptual clusters – one per line. These clusters are organized according to their domain association: MEDICINE (left), FOOD (top-right), unclear (bottom-right).

word usage in corpora. We adopt the distributional approach in order to facilitate research in languages (such as Farsi) for which coverage of existing knowledge bases is limited. The only requirements for our approach are a corpus with documents written in that language and a syntactic parser for the language. We use the Malt dependency parser to obtain syntactic parses for web documents in each language.

Table 2 of Section 3.1 shows some of the words which participate regularly with the word "cure" in a dependency relation. These syntactic contexts of the word "cure" form the basis for one semantic representation we use to find other similar words, which we will call *DepVec*. All of the dependency relations for a word are used to form a vector-based distributional representation for that word. This representation projects words which are semantically similar to one another onto vectors which are near to each other in the vector space. In the following subsection, we describe DepVec along with LSA and word2vec which are alternative vector space models of word meaning. These vector spaces are then used to calculate similarities between words in order to cluster them and to align them with lexicons which model our existing conceptual spaces.

4.1 Dependency Vectors (DepVec) space

In our DepVec vector space model, each word is represented by a vector whose elements correspond to syntactic contexts of the word. Each element of the vector for word w corresponds to the frequency of a unique dependency relation (w, r, w') seen in the corpus. For example, if the relation $(whale, nsubj^{-1}, swim)$ is extracted once, then the vector for "whale" contains a 1 in the element for $(nsubj^{-1}, swim)$, and the vector for "swim" contains a 1 for the element $(nsubj, whale)$. This representation corresponds that proposed by Lin (1998).

However, the use of raw frequency counts in these vectors leads to a situation in which words that are more frequent in the corpus (e.g., "of", "the", "one") will have higher frequencies in the vectors by chance alone, and so a high co-occurrence count for those words is not indicative of a significant relation to the word. We overcome this limitation by replacing the raw frequency counts in each vector with their corresponding G-test scores. The G-test is a measure of statistical significance for proportions, similar to the Chi-square test, which measures the degree to which a particular triple (w, r, w') was found to occur more frequently than expected given all relations (w'', r, w'). If w' occurs far more often with w than it does with other words, then it will receive a high G-test score for w. In particular, the G-test score is computed according to the following equation:

$$G = 2 \sum_i O_i \cdot ln(O_i / E_i)$$

where the index i ranges over the four cells of a 2x2 contingency table, O_i is the observed count in cell i, and E_i is the expected count in the same cell.

Language	Source	# Documents	Language	Source	# Documents
English	ClueWeb	13,361,743	Spanish	ClueWeb	3,682,478
Russian	ruWac	1,173,590	Farsi	Online news sites	835,588

Table 4: Statistics of the corpora used to construct the vector space models

4.2 Latent Semantic Analysis (LSA)

While the DepVec model provides information about the immediate contexts a word can be expected to occur in, it does not directly capture information about the broader contexts typical of that word, such as topical information. Latent Semantic Analysis (LSA) is a well-studied model (Landauer and Dumais, 1997) which does capture such topical information. The LSA model utilizes a singular value decomposition of a TF-IDF weighted matrix representation of the term-document co-occurrences. Terms and documents are then represented in a reduced dimensionality space using only the information from the eigenvectors with the k largest eigenvalues.

4.3 Continuous skip-gram model (W2V)

Mikolov et al. (2013) recently presented a new method for determining distributional word representations based on a shallow neural network model. The values of the latent vector for each word are trained to optimize prediction of the words within a 10 token window. This prediction is performed using the term's latent vector as the input to a series of log-linear classifiers with outputs which correspond to probability distributions over the tokens within the context window. Each position in the context window is assigned its own classifier weights, so that the model used for making predictions about words immediately following the input term is different than the model which makes predictions about the words two tokens after the term, and so on. Because these latent vector representations are in a low dimensionality space (300 dimensions in our case), the training process will tend to move the representations for similar words closer together in this space in order to maximize the predictive accuracy of their contexts.

One benefit of the continuous skip-gram model is that it creates representations which capture some local context as in the DepVec model, which is required to make predictions about the previous and next tokens. However, it must also encode some topical knowledge in order to make accurate predictions about the words seven tokens away. Therefore, using the latent term representations from the continuous skip-gram model as a vector space puts it in a convenient position in between the two others we presented.

4.4 Corpus Processing

The vector models described above were developed using web-scale corpora collected from a combination of frequently used NLP corpora and web crawls on news websites. Table 4 indicates the number of documents used for each language along with their source. These corpora were part-of-speech tagged with in-house POS taggers for English and Spanish, TreeTagger[3] for Russian, and hunpos[4] for Farsi. The open-source MaltParser was used to produce dependency parses for all four languages (Nivre, 2003). Dependency counts for all words occurring fewer than 40 times and for triples occurring fewer than three times were discarded to minimize noise.

5 Concept Induction and CM Recognition

In Section 4, we described our DepVec representation of terms as vectors in a high-dimensional distributional space. These vector representations encode both the dominant grammatical contexts of a term as well as the selectional preference information associated with it in the form of G-test scores. In this section, we describe our methodology for inducing conceptual domains for a linguistic metaphor by adapting techniques for unsupervised word-sense induction (Erk and Padó, 2008; Korkontzelos and Manandhar, 2010; Hope and Keller, 2013). In particular, we induce conceptual domains in an unconstrained manner by extracting the grammatical co-occurrents of an LM source term (i.e., the 'concept candidates') and clustering them into semantically-related concept clusters. Both the clusters and our

[3] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

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

given source domains are then mapped into a distributional vector space, allowing us to compute cluster-to-domain scores. Finally, each source domain is assigned a score based on its affinity to each individual cluster with these affinity scores weighted according to cluster quality. This results in an overall weighted ranking of the given source conceptual domains for the linguistic metaphor.

5.1 Extracting Concept Candidates

Given a linguistic metaphor which consists of a metaphor source, s (e.g., "cure"), and a metaphor target, t (e.g., "poverty"), our system extracts a set of terms (i.e., "concept candidates") from the typical grammatical contexts of s as found in the web-scale corpus described in Section 4.4. In order to extract these candidates, we first determine the syntactic relation, r, which exists between s and t. This relation is the key point of interaction between the domains of the source and the target for the given LM and, as such, it provides an indication of which terms will contribute the most to our understanding of the underlying conceptual mapping. In addition, we make use of a predefined set of relations that are semantically meaningful – specifically the subjects and objects of verbs (i.e., "nsubj", "nsubjpass", and "dobj"),[5] attributes and verbs associated with nouns (i.e., "amod", "dobj^{-1}", "nsubj^{-1}", and "nsubjpass^{-1}"), the terms modified by adjectives or adverbs (i.e., "advmod^{-1}" and "amod^{-1}"), and prepositional relations (e.g., "prep_by", "prep_of", "prep_for"). Using this set of relations, R, we extract the set of candidate terms, X, that have been found to co-occur with the term s within some relation $r_i \in R$ in the preprocessed, web-scale corpus described in Section 4.4 such that $X = \{x|(s, r_i, x)\text{exists in the corpus}\}$.

To improve the quality of our extracted candidates, we apply three criteria to isolate those that best exemplify the underlying non-metaphorical senses of s. First, we anticipate that any term in X which does not co-occur with s at least k times will not be informative,[6] and so we remove such terms from further processing. Next, we predict that poorly imageable terms (i.e., highly abstract terms) are likely to represent metaphorical usages of s and so are unlikely to be integral to a given literal source domain, so these are filtered out as well.[7] Finally, to improve our ability to map these candidates into a conceptual domain, we remove terms that are not significantly related to any of our provided source domains (i.e., those that are off-topic) along with terms that are strongly related to multiple source domains (i.e., those that are ambiguous) as these provide little evidence to distinguish the most appropriate concept for the given LM.[8] We determine the relatedness of a term to a source domain by measuring the similarity of the term and domain vectors in our distributional space as described in Section 5.3.

5.2 Clustering Concept Candidates

Once the candidates have been extracted, they are clustered using a hierarchical agglomerative clustering algorithm with the distance metric defined as the cosine distance between the vectors within one of our distributional vector spaces. Each cluster is then assigned a quality score based on its size (to prefer large clusters with a large amount of semantic evidence), average internal distance (to prefer tighter clusters), and co-occurrence frequency with the LM source (to prefer more closely related terms). Formally, we define the weight associated with a given cluster using the following equation:

$$w(C) = (1 - IDIST(C)) * (S2(C) + FREQ(C) * (1 + S2(C)))$$

$$S2(C) = \frac{\max(SIZE(C)^2, k)}{k}$$

where $IDIST(C)$ represents the average vector distance between all pairs of terms in cluster C, $FREQ(C)$ represents the total co-occurrence frequency of the terms in C with the original LM,

[5] These dependency relation types come from the MaltParser.

[6] We empirically set k to 3.

[7] We estimate candidate imageability by combining the scores of the candidate's most distributionally similar words for which an imageability score is available in the MRC psycholinguistics database (Coltheart, 1981) using the ranked weighting methodology described in Mohler et al. (2014).

[8] Note that filtering by conceptual domain relatedness is only necessary when mapping the induced concepts to a predefined set of source concepts.

$SIZE(C)$ represents the number of unique terms in C, and k is a tuning parameter meant to favor large clusters.[9] Singleton clusters are discarded.

5.3 Assigning Domain Scores to Concept Candidates

We propose two methods for calculating domain scores for candidates – one which attempts to compare candidate vectors to a source domain directly, and and another which attempts to compare them to individual facets of the domain. These two methods rely on representing sources [CentS], or facets [CentF], as centroids which take the average of the vectors of each the lexemes assigned to that source (or facet). Our three vector spaces – DepVec, W2V, and LSA – along with our two methods for mapping terms to domains – CentS and CentF – correspond to six approaches to modeling a CM domain in some vector space.

In each case, the result for a given candidate is a distribution over all source domain scores. This distribution is then normalized by subtracting the mean score between the candidate vector and any of the source concepts. Formally, we define the normalized distribution for concept candidate x as:

$$S(x, D_y) = (1 - DIST(x, D_y)) - \frac{\sum_{D_k \in D} (1 - DIST(x, D_k))}{|D|}$$

where D is defined as the set of all known source domains and $DIST(x, d)$ is the cosine distance from x to a CM domain d in one of our vector spaces.

5.3.1 Assigning Domain Scores to Clusters

Within a given cluster (found as described in Section 5.2), the individual concept domain scores can then be combined to produce cluster-level domain scores. For a given cluster C_x, the score associated with a particular source domain D_y is defined as follows:

$$S(C_x, D_y) = \sum_{i=1}^{N} \frac{S(C_{xi}, D_y)}{\alpha^i}$$

where N represents the number of concepts in C_x with a positive score for the domain D_y, C_{xi} is the i-th highest score associated with any candidate in the cluster, and α is a tuning parameter which bounds the growth of the cluster-level score.[10] Any cluster with a maximum domain score that does not exceed a threshold is discarded as being weakly related to any CM source domain.

5.3.2 Assigning Domain Scores to the Linguistic Metaphor

We then sum the cluster-level source domain scores, scaling each by its associated cluster quality weight $w(c)$ as computed in Section 5.2. By scaling cluster domain scores in this way, we ensure that the most pure and discriminating clusters contribute the most to the overall LM domain scores. The final result measuring the association between the given LM and the source domain D_y is then defined as:

$$S(D_y) = \sum_{C_x \in C} w(C_x) * S(C_x, D_y)$$

Applied across all known domains, we therefore produce a ranked and scored list of CM source domains (i.e., a mapping) that are associated with the given linguistic metaphor and can be used to drive more robust interpretation of the metaphor.

6 Evaluation

We evaluate two aspects of our end-to-end CM recognition system. First, we analyze the impact of our choice of vector space. Specifically, we compare the use of our DepVec space to link concept candidates

[9] In our experiments, k is set to 5.

[10] We have used a value of $\alpha = 2$ which ensures that the result remains within the bounds [0.0,1.0].

with source domains against two off-the-shelf vector space models – the continuous skip-gram model [W2V] (Mikolov et al., 2013) [11] and latent semantic analysis [LSA] (Landauer and Dumais, 1997). Both alternative models were trained over the same corpus as in our DepVec space using a predefined number of dimensions (300 for W2V; 400 for LSA). Second, we have experimented with two different metrics for calculating the distance between a vector and a source concept – the cosine distance to the source-level centroid (CentS) and the cosine distance to the facet-level centroid (CentF).

Our evaluation dataset consists of a held out, unseen set of documents taken from a variety of news articles, opinion pages, and blogs on the open web. These documents consist of 3 to 5 sentences each and cover four of our focus languages.[12] They were then annotated by two native-proficiency speakers in the following way. For each LM, they were instructed to choose the most closely related source concept from our list of 51 provided. Any source concepts selected by at least one annotator were considered correct. Since our CM recognition system produces a ranked list of source concepts, we report both the accuracy associated with our top-ranked concept and the accuracy of the system when allowed to select two.

		Cluster Linking							
		English		Spanish		Russian		Farsi	
Vector Space	**Distance**	Acc@1	Acc@2	Acc@1	Acc@2	Acc@1	Acc@2	Acc@1	Acc@2
DepVec	CentS	28.0%	44.1%	33.3%	43.4%	24.4%	32.6%	16.5%	27.5%
	CentF	25.8%	40.9%	33.3%	49.4%	25.6%	34.9%	26.4%	40.7%
LSA	CentS	34.4%	45.2%	31.0%	41.4%	27.9%	41.9%	22.0%	27.5%
	CentF	**38.7%**	**54.9%**	27.6%	46.0%	29.1%	47.7%	31.9%	44.0%
W2V	CentS	24.7%	36.6%	42.5%	55.2%	31.4%	43.0%	25.3%	34.1%
	CentF	28.0%	44.1%	**46.0%**	**58.6%**	**34.9%**	**48.8%**	**35.2%**	**48.4%**

Table 5: The accuracy of our conceptual interpretation system. We experiment with three vector spaces (LSA, W2V, and DepVec) and two source concept centroid representations – source-level (CentS) and facet-level (CentF).

These results indicate that the continuous skip-gram vector space [W2V] is well suited to the task of cluster-level concept mapping, consistently and significantly outperforming both the LSA space and the DepVec space in every language but English. We believe that this is a result of its probabilistic representation of local context which implicitly collects many of the same relations as the DepVec model while incorporating the advantages associated with dimensionality reduction which has not been incorporated into our DepVec model.[13] We further observe an unmistakable dominance of the facet-level centroid representation over the source-level representation. Based on these results, we believe that we have successfully demonstrated the contribution of our system's vector-space clustering component which groups concept candidates at a facet-level granularity.

7 Conclusion

In this paper, we have presented a novel approach to the problem of multilingual conceptual metaphor recognition which combines facet-based concept induction with a distributional vector space representation of metaphor. We have experimentally demonstrated the advantage of our fine-grained concept induction approach within a variety of vector space models, including our novel DepVec space. Taken together, we hypothesize that a facet-level conceptual model represented in a relational context vector space will serve as a reliable foundation enabling high-quality metaphoric interpretation in future metaphor research. Future work includes expanding the set of concept candidates through higher-order dependency contexts, improved clustering techniques, and evaluating the induced clusters directly.

[11] We make use of the implementation included as part of the gensim python package: `http://radimrehurek.com/gensim/`

[12] This dataset consists of the following counts of documents: English (92), Spanish (86), Russian (85), Farsi (90).

[13] During our pilot experiments, we applied singular value decomposition (SVD) to the DepVec space without any significant improvement to system performance.

Acknowledgments

This research is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-12-C-0025. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

References

Pierre Beust, Stéphane Ferrari, Vincent Perlerin, et al. 2003. NLP model and tools for detecting and interpreting metaphors in domain-specific corpora. In Proceedings of the Corpus Linguistics 2003 conference, volume 16, pages 114–123. Citeseer.

Danushka Bollegala and Ekaterina Shutova. 2013. Metaphor interpretation using paraphrases extracted from the web. PloS one, 8(9):e74304.

D. Bracewell, M. Tomlinson, M. Mohler, and B. Rink. 2014. A tiered approach to the recognition of metaphor. In Computational Linguistics and Intelligent Text Processing.

Siaw-Fong Chung, Kathleen Ahrens, and Chu-Ren Huang. 2005. Source domains as concept domains in metaphorical expressions. International Journal of Computational Linguistics and Chinese Language Processing, 10(4):553–570.

Max Coltheart. 1981. The MRC psycholinguistic database. The Quarterly Journal of Experimental Psychology, 33(4):497–505.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 897–906. Association for Computational Linguistics.

J. Feldman and S. Narayanan. 2004. Embodied meaning in a neural theory of language. Brain and language, 89(2):385–392.

Lisa Gandy, Nadji Allan, Mark Atallah, Ophir Frieder, Newton Howard, Sergey Kanareykin, Moshe Koppel, Mark Last, Yair Neuman, and Shlomo Argamon. 2013. Automatic identification of conceptual metaphors with limited knowledge. In Twenty-Seventh AAAI Conference on Artificial Intelligence.

David Hope and Bill Keller. 2013. MaxMax: a graph-based soft clustering algorithm applied to word sense induction. In Computational Linguistics and Intelligent Text Processing, pages 368–381. Springer.

Ioannis Korkontzelos and Suresh Manandhar. 2010. UoY: Graphs of unambiguous vertices for word sense induction and disambiguation. In Proceedings of the 5th international workshop on semantic evaluation, pages 355–358. Association for Computational Linguistics.

G. Lakoff and M. Johnson. 1980. Metaphors we live by, volume 111. Chicago London.

G. Lakoff. 1993. The contemporary theory of metaphor. Metaphor and thought, 2:202–251.

T.K. Landauer and S.T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological Review; Psychological Review, 104(2):211.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In Proceedings of the 17th international conference on Computational linguistics-Volume 2, pages 768–774. Association for Computational Linguistics.

Z.J. Mason. 2004. CorMet: A computational, corpus-based conventional metaphor extraction system. Computational Linguistics, 30(1):23–44.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Michael Mohler, Marc Tomlinson, and David Bracewell. 2013. Applying textual entailment to the interpretation of metaphor. In IEEE Seventh International Conference on Semantic Computing (ICSC), pages 118–125. IEEE.

Michael Mohler, Marc Tomlinson, David Bracewell, and Bryan Rink. 2014. Semi-supervised methods for expanding psycholinguistics norms by integrating distributional similarity with the structure of WordNet. Language Resources and Evaluation Conference 2014.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In Proceedings of the 8th International Workshop on Parsing Technologies (IWPT. Citeseer.

Astrid Reining and Birte Lönneker-Rodman. 2007. Corpus-driven metaphor harvesting. In Proceedings of the Workshop on Computational Approaches to Figurative Language, pages 5–12. Association for Computational Linguistics.

Hinrich Schütze. 1998. Automatic word sense discrimination. Computational linguistics, 24(1):97–123.

Ekaterina Shutova and Lin Sun. 2013. Unsupervised metaphor identification using hierarchical graph factorization clustering. In Proceedings of NAACL-HLT, pages 978–988.

E. Shutova, L. Sun, and A. Korhonen. 2010. Metaphor identification using verb and noun clustering. In Proceedings of the 23rd International Conference on Computational Linguistics, pages 1002–1010. Association for Computational Linguistics.

Ekaterina Shutova. 2010. Automatic metaphor interpretation as a paraphrasing task. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 1029–1037. Association for Computational Linguistics.

Tomek Strzalkowski, George Aaron Broadwell, Sarah Taylor, Laurie Feldman, Boris Yamrom, Samira Shaikh, Ting Liu, Kit Cho, Umit Boz, Ignacio Cases, et al. 2013. Robust extraction of metaphors from novel data. Meta4NLP 2013, page 67.

T. Veale and Y. Hao. 2008. A fluid knowledge representation for understanding and generating creative metaphors. In Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1, pages 945–952. Association for Computational Linguistics.

Yorick Wilks, Lucian Galescu, James Allen, and Adam Dalton. 2013. Automatic metaphor detection using large-scale lexical resources and conventional metaphor extraction. Meta4NLP 2013, page 36.

Part of Speech Tagging for French Social Media Data

Farhad Nooralahzadeh
École Polytechnique
de Montréal
Montréal, PQ, Canada
nooralahzadeh
@gmail.com

Caroline Brun
Xerox Research Centre
Europe
Meylan, France
caroline.brun
@xrce.xerox.com

Claude Roux
Xerox Research Centre
Europe
Meylan, France
claude.roux
@xrce.xerox.com

Abstract

In the context of Social Media Analytics, Natural Language Processing tools face new challenges on on-line conversational text, such as microblogs, chat, or text messages, because of the specificity of the language used in these channels. This work addresses the problem of Part-Of-Speech tagging (initially for French but also for English) on noisy language usage from the popular social media services like Twitter, Facebook and forums. We employ a linear-chain conditional random fields (CRFs) model, enriched with several morphological, orthographic, lexical and large-scale word clustering features. Our experiments used different feature configurations to train the model. We achieved a higher tagging performance with these features, compared to baseline results on French social media bank. Moreover, experiments on English social media content show that our model improves over previous works on these data.

1 Introduction

There are many challenges inherent to applying standard natural language analysis techniques to social media. On-line conversational texts, such as tweets are quite challenging for text mining tools, and in particular for opinion mining, as they contain very little contextual information and assume too much implicit knowledge. They expose much more language variation and tend to be less grammatical than regular texts such as news articles or books. Furthermore, they contain unusual capitalization, and make frequent use of emoticons, abbreviations and hash-tags, which can form an important part of their inner meaning (Maynard et al., 2012). Conventional natural language processing tools for regular texts have achieved reasonably high accuracy thanks to machine learning techniques on large annotated data set. However, "off the shelf" language processing systems fail to work on social media data and their performance on this domain degrade very fast. For example, in English Part-Of-Speech tagging, the accuracy of the Stanford tagger (Toutanova et al., 2003) falls from 97% on Wall Street Journal text to 85% accuracy on Twitter (Gimpel et al., 2011), similarly the MElt POS tagger (Denis and Sagot, 2012) drops from 97.7% on the French Treebank (called the FTB-UC by (Candito and Crabbé, 2009)) to 85.2% on on-line conversational texts (Seddah et al., 2012). In Named Entity Recognition, the CoNLL-trained Stanford recognizer achieves 44% F-measure (Ritter et al., 2011), down from 86% on the CoNLL test set (Finkel et al., 2005); regarding parsing, see for example (Foster et al., 2011; Seddah et al., 2012), poor performances have been reported for different state-of-the-art parsers applied to English and French social media content.

The main objective of this work is to implement a dedicated Part-Of-Speech (POS) tagger for French social media content such as Twitter, Facebook, blogs, forums and customer reviews. We used the first user-generated content resource for French presented by Seddah et al. (2012), which contains a fine-grained tag set and has been extracted from various social media contents. We have designed and implemented a POS tagger considering one of the well-known *discriminative* type of sequence-based methods; Conditional Random Fields (CRF) (Lafferty et al., 2001). To deal with sparsity and unknown

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1764–1772, Dublin, Ireland, August 23-29 2014.

words, we have applied unsupervised techniques to enrich the feature set. Finally, we have evaluated our tagger performance with different configurations on annotated corpora from French social media.

We will first present related work in Part-Of-Speech tagging (Section 2) on noisy data like social media content. In Section 3, the annotated dataset and its characteristics (e.g., tag set) are described. Section 4 presents the result of applying the MElt POS tagger to user generated text as our baseline (Seddah et al., 2012). In Section 5, we explain how we design and implement our POS tagger. Section 6 is devoted to experiments and performance of our tagger. Section 7 describes the evaluation of the new tagger on English social media texts. Conclusion and future work are given in Section 8.

2 Related work

Online conversational texts, typified by micro-blogs, chat, and text messages, are a challenge for natural language processing. Unlike the highly edited genres for which conventional NLP tools have been developed, conversational texts contain many non-standard lexical items and syntactic patterns. These are the result of unintentional errors, dialectal variation, conversational ellipsis, topic diversity, and creative use of language and orthography (Eisenstein, 2013)

The language technology research community proposes two approaches to deal with noisy texts, namely normalization and domain adaptation, which are briefly described here.

2.1 Normalization

One way to deal with ill-formed language is to turn it into a well-formed language as a pre-processing task: "normalizing" social media or SMS messages to better conform to the language that the technology expects. For example, (Han and Baldwin, 2011) propose the lexical normalization of short text messages, such as tweets, based on string and distributional similarity. They describe a method to identify and normalize ill-formed words. Word similarity and context are exploited to select the best candidate for noisy tokens.

2.2 Domain adaptation

The other approach is instead to adapt the tools to fit the text. A series of papers has followed the mold of "NLP for Twitter," including POS tagging (Gimpel et al., 2011; Owoputi et al., 2013), named entity recognition (Finin et al., 2010; Ritter et al., 2011; Xiaohua et al., 2011), parsing (Foster et al., 2011), dialog modeling (Ritter et al., 2010) and summarization (Hutton and Kalita, 2010). These works adapt various parts of the natural language processing pipeline for social media text, and make use of a range of techniques (Preprocessing, New labeled data, New annotation schemes, Self training, Distributional features, Distance supervision) (Eisenstein, 2013).

Recently, Seddah et al. (2012) followed the second approach on French social media content and provided new labeled data and annotation schemes. They applied the MElt POS tagger (Denis and Sagot, 2012) embedded within text normalization and correction to noisy user generated texts and presented baseline POS tagging and statistical constituency parsing results.

3 Annotated Dataset

A set of 1,700 sentences (38k tokens) has been extracted from various types of French Web 2.0 user generated content (Facebook, Twitter, Video games and medical web forums) by Seddah et al. (2012). They selected these corpora through direct examination of various search queries and ranked the texts according to their distance from the French Treebank style, by measuring noisiness using the kullback-Leibler divergence between the distribution of trigrams of characters in given corpus and the distribution of trigrams of characters in the French Treebank reference. Some properties of this corpora are shown in Table 1.

They targeted the annotation scheme of the FTB-UC in order to annotate the French social media bank. The tagset includes 28 POS tags from FTB-UC and compound tags with additional categories specific to social media, including **HT** for Twitter hashtags and **META** for meta-textual tokens, such as

Twitter's "RT". Twitter at-mention as well as URLs and e-mail addresses have been tagged **NPP** which is the main difference with other works on on-line conversational texts. The inter-annotator agreement rate in this corpora range between 93.4% for **F**ACEBOOK data and 97.44% for **J**EUX**V**IDEOS.COM (Table 1) which indicates an almost perfect agreement on the corpus (Landis and Koch, 1977).

Corpus Name	**# sent.**	**# tokens**	Inter Annotator Agreement %
TWITTER	216	2465	95.40
FACEBOOK	452	4200	93.40
JEUX**V**IDEOS.COM	199	3058	97.44
DOCTISSIMO	771	10834	95.05

Table 1: Annotated datasets

4 Baseline

This section presents the performance of a state-of-the-art POS tagger for French, conducted by Seddah et al. (2012). They used FTB-UC as training, development and test data. First, they applied several correction processes in order to wrap the POS tagger to tag a sequence of tokens as close as possible to standard French and training corpus. Then, the MElt tagger has been used with a set of 15 language-independent rules, that aim at assigning the correct POS to tokens that belong to categories not found in training corpus (e.g., URLs, e-mail addresses, emoticons). The preliminary evaluation experiments with normalization and correction wrapper showed *84.72%* and *85.28%* token accuracy over annotated development and test set respectively.

5 New POS Tagger Development

Conversational style context and 140-character limitation in micro-blogs require users to express their thought or reply to others' messages within a short text. Therefore, without being ambiguous, some words are usually abbreviated with a special spelling. For example, *c t* usually means c'était (it was); *qil* denotes qu' il (that it/he).

Our tagger is based on sequence labeling models (CRF), enabling arbitrary local features to be integrated into a log-linear model. We employed three categories of feature templates to deal with syntactic variations on social media contents and alleviating the data sparseness problem.

5.1 Basic Feature Templates

The feature templates we use here are a superset of the largely language independent features used by (Ratnaparkhi, 1996; Toutanova and Manning, 2000; Toutanova et al., 2003). These features fall into two main categories. A first set of features tries to capture the *lexical form* of the word being tagged: it includes prefixes and suffixes (of at most 10 characters) from the current word, together with binary features based on the presence of special characters such as numbers, hyphens, and uppercase letters, within w_i. A second set of features directly models the context of the current word and tag: it includes the previous tag, surrounding word forms in a 5 tokens window. The detailed list of feature templates we used in this category is shown in Table 2.[1]

Context	
$w_i = X$,$i \in [-2,-1,0,1,2]$	& $t_0 = T$
$w_i w_j = XY$, $(i,j) \in \{(-1,0),(0,1),(-2,0),(0,2)\}$	& $t_0 = T$
$w_i w_j w_k = XYZ$, $(i,j,k) \in \{(-2,-1,0),(0,1,2),(-1,0,1)\}$	& $t_0 = T$
$w_i w_j w_k w_l w_m$=$XYZPQ$, (i,j,k,l,m) =$(-2,-1,0,1,2)$	& $t_0 = T$
t_{-1}	& $t_0 = T$
Lexical and Orthographic	
$f(w_i)$,$i \in [-1,0,1]$,$f \in F$	& $t_0 = T$
$m(w_i)$, $i \in [-1,0,1]$, $m \in M$	& $t_0 = T$

Table 2: Basic Feature Templates

[1] w_0 means the token at the current position while w_{-1} means the previous token.

The model generates the feature space by scanning each pair in the training data with the feature templates given in Table 2. For example, if we consider the following tweet from the training set, the generated features based on the first template can be seen in Table 3, in which the current word is "vous" (position 6) .

Sample tweet : "@Marie Je vais tener De vous produire la vidéo *-* "

word:	@Marie	Je	vais	tener	De	vous	produire	la	vidéo	*-*
Tag:	NPP	CLS-SUJ	V	VINF	P	CLO-A_OBJ	VINF	DET	NC	I
Position:	1	2	3	4	5	6	7	8	9	10

w_0=**vous**	$\&t_0$=**O**
w_{-1}=**De**	$\&t_0$=**O**
w_{-2}=**tener**	$\&t_0$=**O**
w_{+1}=**produire**	$\&t_0$=**O**
w_{+2}=**la**	$\&t_0$=**O**

Table 3: Generated features with template :
$w_i = X$,$i \in [-2, -1, 0, 1, 2]$ $\quad \&t_0 = T$

We defined two sets of operations, F and M. Each operation maps tokens to equivalence classes.
F is a set of regular expression rules that detect specific patterns on w_i and return binary values. The functions $f(w_i) \in F$ include the rules as detailed in the following list (List 1):

List 1: Set of regular expression rules (F)

- ▷ Return "$True$" if the w_i contains Punctuation marks otherwise return "$False$"
- ▷ Return "$True$" if the w_i is list of Punctuation marks otherwise return "$False$"
- ▷ Return "$True$" if the w_i contains digits otherwise return "$False$"
- ▷ Return "$True$" if the w_i number otherwise return "$False$"
- ▷ Return "$True$" if all letters of w_i are capitalized otherwise return "$False$" allNumber
- ▷ Return "$True$" if the w_i starts with capital letter otherwise return "$False$"
- ▷ Return "$True$" if the w_i has"URL" pattern otherwise return "$False$"
- ▷ Return "$True$" if the w_i has "Email" pattern otherwise return "$False$"
- ▷ Return "$True$" if the w_i has "Abbreviation" pattern otherwise return "$False$"
- ▷ Return "$True$" if the w_i has "Arrow" pattern otherwise return "$False$"
- ▷ Return "$True$" if the w_i has "Time " pattern otherwise return "$False$"
- ▷ Return "$True$" if the w_i has "NumberWithCommas" pattern otherwise return "$False$"
- ▷ Return "$True$" if the w_i has symbol representing "RT:retweeting" form otherwise return "$False$"
- ▷ Return "$True$" if the w_i has symbol representing "At-Mention" form otherwise return "$False$"
- ▷ Return "$True$" if the w_i has symbol representing "hash-tagh" form otherwise return "$False$"

M is a set of orthographic transformations that maps a string to another string via a simple surface level transformation. The functions $m(w_i) \in M$ are given in List 2 :

List 2: Set of orthographic transformation (M)

- ▷ Return capitalized type of w_i ,These types are (allCap, shortCap, longCap, noCap, initCap, mixCap)
(e.g.,"Plus-tard" → "initCap" ,"RT" →"allCap,longCap")

- ▷ Return the *type* of w_i, obtained by replacing $[a-z]$ with x, $[A-Z]$ with X, and $[0-9]$ with 9
(e.g.,., "@DJRyan1der" → "@XXXxxx9xxx")

- ▷ Return a vector of Unicode matching of the string w_i
(e.g., "@DJRyan1der" → "$[64-68-74-82-121-97-110-49-100-101-114]$")

- ▷ Return the first n character of x (n-gram prefix), where $1 \leq n \leq 10$
- ▷ Return the last n character of x (n-gram suffix), where $1 \leq n \leq 10$

5.2 Word Clustering Feature Templates

To bridge the gap between high and low frequency words, we employed word clustering to acquire knowledge about paradigmatic lexical relations from large-scale texts. Our work is inspired by the suc-

cessful application of word clustering in supervised NLP models (Miller et al., 2004; Turian et al., 2010; Ritter et al., 2011; Owoputi et al., 2013).

Various clustering techniques have been proposed, some of which, for example, perform automatic word clustering optimizing a maximum likelihood criterion with iterative clustering algorithms. In this work, we focus on distributional word clustering, based on the assumption that the words that appear in similar contexts (especially surrounding words) tend to have similar meanings.

5.2.1 Brown Clustering

We used our unlabeled Twitter corpus (4M tweets) to improve our tagger performance. This corpus has been extracted in the framework of a French government funded ANR project called Imagiweb, whose goal is to develop tools to analyse the brand image of entities (persons or companies) on social media. More specifically, one of the focus of the project is to analyse the brand image of politicians on Twitter. Therefore, data about the two main candidates (F. Hollande and N. Sarkozy) in the last French presidential election in May 2012 have been crawled from Twitter, using Twitter API, from 6 months before to 6 months after the elections. Our unlabeled Twitter data is a sub-set of this corpus.

We obtained hierarchical word clusters via Brown Clustering (Brown et al., 1992) on a large set of unlabeled tweets. This algorithm generates a hard clustering, each word belongs to exactly one cluster. The input to the algorithm is a sequence of words $w_i, \ldots, w_n$. Initially, the algorithm starts with each word in its own cluster. As long as there are at least two clusters left, the algorithm merges the two clusters that maximize the resulting cluster quality. The quality is defined on the class-based bigram language model as follows, where C maps a word w to its class $C(w)$.

$$p(w_i|w_1, \ldots, w_{i-1}) = p(C(w_i)|C(w_{i-1}))p(w_i|C(w_i))$$

We ended up with 500 clusters (the optimal number of clusters according to the performance of the tagger among different number of clusters) with 222,788 word types by keeping the words appearing 10 or more times. Since Brown clustering creates hierarchical clusters in a binary tree, we used the feature template which maps the word w_i to the cluster at depths $2, 4, \ldots, 16$ containing w_i. If w_i was not seen while constructing the clusters and thus does not belong to any cluster we tried to find similar words by computing *Jaro-Winkler distance* (Philips, 1990; Winkler, 2006) and mapped the best match to the cluster depths. Nevertheless, if we couldn't find the best match (the threshold of the similarity score is 0.9), we mapped it to a special $NULL$ cluster. The detailed list of feature templates we used in this category is shown in Table 5.[2]

5.2.2 MKCLS Clustering

We also did some experiments, using another popular clustering method based on the exchange algorithm (Kneser and Ney, 1993). The objective function maximizes the likelihood $\prod_{i=1}^{n} P(w_i|w_1, \ldots, w_{i-1})$ of the training data given a partially class-based bigram model of the form as follows:

$$p(w_i|w_1, \ldots, w_{i-1}) \approx p(C(w_i)|w_{i-1})p(w_i|C(w_i))$$

We use the publicly available implementation MKCLS [3] to train this model on our French Twitter data (4M tweets). This algorithm provides us with 500 word clusters with 2,768,297 different words.

Word Cluster	
$c(w_i) = X$,$i \in [-2, -1, 0, 1, 2]$ and $c \in C$	& $t_0 = T$
$c(w_i)c(w_j) = XY$, $(i, j) \in \{(-1, 0), (0, 1)\}$ and $c \in C$	& $t_0 = T$
$c(w_i)C(w_j)c(w_k) = XYZ$, $(i, j, k) \in \{(-2, -1, 0), (0, 1, 2), (-1, 0, 1)\}$ and $c \in C$	& $t_0 = T$
$c(w_i)c(w_j)c(w_k)c(w_l)c(w_m)$=$XYZPQ$, (i, j, k, l, m) =$(-2, -1, 0, 1, 2)$ and $c \in C$	& $t_0 = T$

Table 5: Word Clustering Feature Templates

[2] $c(w_i) \in C$ map the word w_i to the clusters at depths $2, 4, \ldots, 16$

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

6 Experiments

For the implementation of discriminative sequential model, we chose the *Wapiti* [4] toolkit (Lavergne et al., 2010). Wapiti is a very fast toolkit for segmenting and labeling sequences with discriminative models. It is based on maxent models, maximum entropy Markov models and linear-chain CRF and proposes various optimization and regularization methods to improve both the computational complexity and the prediction performance of standard models. Wapiti has been ranked first on the sequence tagging task for more than a year on MLcomp[5] web site.

6.1 Training and parameter regularization

In the training of log-linear models, regularization is normally required to prevent the model from over fitting on the training data. The two most common regularization methods are called L1 and L2 regularization (Tsuruoka et al., 2009). *Wapiti* uses the elastic-net penalty of the form:

$$\rho_1 * |\theta|_1 + \frac{\rho_2}{2} * ||\theta||_2^2$$

and it is implemented with 3 different algorithms: *Orthant-Wise Limited-memory Quasi-Newton* (OWL-QN: L-BFGS), *Stochastic Gradient Descent* (SGD) and *Block Coordinate Descent*. We trained with *L-BFGS*, a classical Quasi-Newton optimization algorithm with limited memory which minimizes the regularized objective and uses elastic net regularization. Using even a very small L1 penalty excludes many irrelevant or highly noisy features. We carried out a grid search for the regularization values, assessing with F-measure and accuracy. We conducted a first order linear chain CRF model on the French corpora with classical setting (training set: 80%, development set: 10% and test set: 10%) for $L1 \in \{0, 0.0625, 0.125, 0.25, 0.5, 1, 2, 4, 8, 16\}$ and $L2 \in \{0, 0.0325, 0.0625, 0.125, 0.25, 0.5, 1, 2, 4, 8, 16\}$ (Owoputi et al., 2013). In any experiment, the result of the regularization values were close to each other, therefore we selected L1,L2=$(0.25, 0.5)$ achieving 80.4% and 90.6% F-measure and accuracy on the corpora respectively.

6.2 Performance

In order to assess how the results of our tagger based on the current limited corpora could be generalized to an independent data set, a set of 10-fold cross validation experiments has been performed. We investigated the effect of each feature template on the tagging. We used *"c: compact"* option in *Wapiti* which enables model compaction at the end of the training. This removes all inactive observations from the model, leading to a much smaller model when an L1-penalty is used.
Table 6 shows the result of each experiment, measured by token and sentence accuracy. It shows that word clustering is a very strong source of lexical knowledge and significantly increases the performance of our tagger.

Feature Templates	Token Accuracy %	Sentence Accuracy %
B	88.2	45.8
B+C1	90.8	49.9
B+C2	90.3	50.3
B+C1+C2	91.9	51.1

B: Basic Feature Templates
C1: Brown word-Clustering Feature Templates
C2: MKCLS word-Clustering Feature Templates

Table 6: Performance of new tagger based on CRF with different configurations

The CRF model with all set of features (B+C1+C2) is the best model with 91.9% and 51.1% token and sentence accuracy on 10-fold cross validation. All of these tagging accuracies are significantly above previous results on the French social bank (baseline).

[4] http://wapiti.limsi.fr/
[5] http://mlcomp.org/

7 Evaluation on English social media Content

In order to implement a tagger for English dedicated to social media content, we used the publicly available clusters data set (Owoputi et al., 2013) to build Brown clustering features. Moreover we performed the same process as in Section 5.2.2 in order to provide MKCLS clustering features with English Twitter data (1 million tweets obtained from [6]).

We applied our tagger with the best configuration to the annotated dataset provided by Ritter et al. (2011). This dataset contains 800 tweets that have been annotated with the Penn Treebank (PTB) tagset (Marcus et al., 1993). We trained and test our system with 10-fold cross validation. Table 7 shows our tagger performance compared to other state-of-art taggers on this data set.

Tagger	Accuracy%
Our new tagger, CRF with B+C1+C2 configuration	90.1
Ritter et al. (Ritter et al., 2011), CRF tagger	88.3
Owoputi et al. (Owoputi et al., 2013), MEMM tagger	90± 0.5

Table 7: Evaluation on Twitter data with PTB tags

In addition, we evaluated the tagger performance on another English social media data: NPS chat ("Chat with PTB tags" (Forsythand and Martell, 2007)). Due to the large number of tokens (50 K), we trained and tested our tagger with a 5-fold cross validation setup. Our new tagger performance as well as the other taggers results are given in Table 8.

Tagger	Accuracy%
Our new tagger, CRF with B+C1+C2 configuration	92.7
Forsythand and Martell (Forsythand and Martell, 2007), HMM tagger	90.8
Owoputi et al. (Owoputi et al., 2013), MEMM tagger	93.4± 0.3

Table 8: Evaluation on Chat data with PTB tags

8 Conclusion and Future Work

In this paper, we have presented an innovative work on POS tagging for French social media noisy input. Because of the specific phenomena encountered in such data and also because of the lack of large training corpus, we proposed a discriminative sequence labeling model (CRF) enhanced with several type of features. After experimenting different configurations of features, we achieved 91.9% token accuracy on target corpus. Moreover, experiments on English social media contents show that our model obtains further improvement over previous works on these data and could be reproduced for other languages. In the future, we plan to pursue this work in two main directions: (a) Integrate the new tagger with a robust syntactic parser and investigate its impact on dependency parsing applied to social media and (b) evaluate the impact of POS tagging on opinion mining on micro-blogs, since this parser is the core component of an opinion mining system applied in different social-media analytics projects.

References

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.*, 18:467–479.

Marie Candito and Benoît Crabbé. 2009. Improving generative statistical parsing with semi-supervised word clustering. In *Proceedings of the 11th International Conference on Parsing Technologies*, IWPT '09, pages 138–141, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pascal Denis and Benoît Sagot. 2012. Coupling an annotated corpus and a lexicon for state-of-the-art pos tagging. *Language Resources and Evaluation*, 46:721–736.

[6] http://illocutioninc.com/site/products-data.html

Jacob Eisenstein. 2013. What to do about bad language on the internet. *In proc. of NAACL.*

Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in twitter data with crowdsourcing. *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88.

Jenny R. Finkel, Trond Grenager, and Manning Christopher. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. *In Proceedings of ACL*, pages 363–370.

Eric N. Forsythand and Craig H. Martell. 2007. Lexical and discourse analysis of online chat dialog. In *Proceedings of the International Conference on Semantic Computing*, ICSC '07, pages 19–26, Washington, DC, USA. IEEE Computer Society.

Jennifer Foster, Ozlem Cetinoglu, Joachim Wagner, Joseph Le Roux, Joakim Nivre, Deirdre Hogan, and Joseph Van Genabith. 2011. From news to comment: Resources and benchmarks for parsing the language of web 2.0. *In Proceedings of IJCNLP*.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers*, 2:42–47.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: makn sens a #twitter. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 1:365–378.

Beaux Sharifi Mark-Anthony Hutton and Jugal Kalita. 2010. Summarizing microblogs automatically. *In Proceedings of NAACL.*

Reinhard Kneser and Hermann Ney. 1993. Improved clustering techniques for class-based statistical language modeling. In *In Proceedings of the European Conference on Speech Communication and Technology (Eurospeech)*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

J. R. Landis and G. G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, March.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513. Association for Computational Linguistics, July.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, June.

Diana Maynard, Kalina Bontcheva, and Dominic Rout. 2012. Challenges in developing opinion mining tools for social media. In *Proceedings of @NLP can u tag #usergeneratedcontent?! Workshop at International Conference on Language Resources and Evaluation, LREC 2012, 26 May 2012, Istanbul, Turkey*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of HLT*, pages 337–342.

O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N.A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. *In Proceedings of NAACL.*

Lawrence Philips. 1990. Hanging on the metaphone. *Computer Language*, 7:12.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging.

Alan Ritter, Colin Cherry, and Bill Dolan. 2010. Unsupervised modeling of twitter conversations. *In Proceedings of NAACL.*

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. *ACL.*

Djamé Seddah, Benoît Sagot, Marie Candito, Virginie Mouilleron, and Vanessa Combet. 2012. The French Social Media Bank: a Treebank of Noisy User Generated Content. In *COLING 2012 - 24th International Conference on Computational Linguistics*, Mumbai, India, Dec. Kay, Martin and Boitet, Christian.

Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. *In EMNLP/VLC 2000*, pages 63–70.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. *IN PROCEEDINGS OF HLT-NAACL*, pages 252–259.

Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. 2009. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 477–485, Stroudsburg, PA, USA. Association for Computational Linguistics.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *ACL*.

William E Winkler. 2006. Overview of record linkage and current research directions. Technical report, BUREAU OF THE CENSUS.

Liu Xiaohua, Zhang Shaodian, Wei Furu, and Zhou Ming. 2011. Recognizing named entities in tweets. *In Proceedings of ACL*.

Morphological Analysis for Japanese Noisy Text Based on Character-level and Word-level Normalization

SAITO Itsumi, SADAMITSU Kugatsu, ASANO Hisako and MATSUO Yoshihiro
NTT Media Intelligence Laboratories
{saito.itsumi, sadamitsu.kugatsu,
asano.hisako, matsuo.yoshihiro}@lab.ntt.co.jp

Abstract

Social media texts are often written in a non-standard style and include many lexical variants such as insertions, phonetic substitutions, abbreviations that mimic spoken language. The normalization of such a variety of non-standard tokens is one promising solution for handling noisy text. A normalization task is very difficult to conduct in Japanese morphological analysis because there are no explicit boundaries between words. To address this issue, in this paper we propose a novel method for normalizing and morphologically analyzing Japanese noisy text. We generate both character-level and word-level normalization candidates and use discriminative methods to formulate a cost function. Experimental results show that the proposed method achieves acceptable levels in both accuracy and recall for word segmentation, POS tagging, and normalization. These levels exceed those achieved with the conventional rule-based system.

1 Introduction

Social media texts attract a lot of attention in the fields of information extraction and text mining. Although texts of this type contain a lot of information, such as one's reputation or emotions, they often contain non-standard tokens (lexical variants) that are considered out-of-Vocabulary (OOV) terms. We define an OOV as a word that does not exist in the dictionary. Texts in micro-blogging services such as Twitter are particularly apt to contain words written in a non-standard style, e.g., by lengthening them ("goooood" for "good") or abbreviating them ("thinkin' " for "thinking"). This is also seen in the Japanese language, which has standard word forms and variants of them that are often used in social media texts. To take one word as an example, the standard form is おいしい (*oishii*, "It is delicious") and its variants include おいしい いいい(*oishiiii*), おいし ぃ(*oishii*), and おいし ー(*oishii*), where the underlined characters are the differences from the standard form. Such non-standard tokens often degrade the accuracy of existing language processing systems, which are trained using a clean corpus.

Almost all text normalization tasks for languages other than Japanese (e.g., English), aim to replace the non-standard tokens that are explicitly segmented using the context-appropriate standard words (Han et al. (2012), Han and Baldwin (2011), Hassan and Menezes (2013), Li and Liu (2012), Liu et al. (2012), Liu et al. (2011), Pennell and Liu (2011), Cook and Stevenson (2009), Aw et al. (2006)). On the other hand, the problem is more complicated in Japanese morphological analysis because Japanese words are not segmented by explicit delimiters. In traditional Japanese morphological analysis, word segmentation and part-of-speech (POS) tagging are simultaneously estimated. Therefore, we have to simultaneously analyze normalization, word segmentation, and POS tagging to estimate the normalized form using the context information. For example, the input パンケーキぉぃしーぃ(*pan-keiki oishiiii*, "This pancake tastes good") written in the standard form is パンケーキおいしい (*pan-keiki oishii*). The result obtained with the conventional Japanese morphological analyzer MeCab (Kudo (2005)) for this input is **パンケーキ (pancake, noun)**/**おいし (unk)**/**ー (unk)**/**ぃ(unk)**/, where slashes indicate the word segmentations and "unk" means an unknown word. As this result shows, Japanese morphological analyzers often fail to

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1773–1782, Dublin, Ireland, August 23-29 2014.

correctly estimate the word segmentation if there are unknown words, so the pipeline method (e.g., first estimating the word segmentations and then estimating the normalization forms) is unsuitable.

Moreover, Japanese has several writing scripts, the main ones being Kanji, Hiragana, and Katakana. Each word has its own formal written script (e.g., 教科書 (*kyoukasyo*, "textbook") as formally written in Kanji), but in noisy text, there are many words that are intentionally written in a different script (e.g., きょうかしょ (*kyoukasyo*, "textbook") is the Hiragana form of 教科書). These tokens written in different script also degrade the performance of existing systems because dictionaries basically include only the standard script. Unlike the character-level variation we described above, this type of variation occurs on a word—level one. Therefore, there are both character-level and word-level non-standard tokens in Japanese informal written text. Several normalization approaches have been applied to Japanese text. Sasano et al. (2013) and Oka et al. (2011) introduced simple character level derivational rules for Japanese morphological analysis that are used to normalize specific patterns of non-standard tokens, such as for word lengthening and lower-case substitution. Although these approaches handle Japanese noisy text fairly effectively, they can handle only limited kinds of non-standard tokens.

We propose a novel method of normalization in this study that can handle both character- and word-level lexical variations in one model. Since it automatically extracts character-level transformation patterns in character-level normalization, it can handle many types of character-level transformations. It uses two steps (character- and word-level) to generate normalization candidates, and then formulates a cost function of the word sequences as a discriminative model. The contributions this research makes can be summarized by citing three points. First, the proposed system can analyze a wider variety of non-standard token patterns than the conventional system by using our two-step normalization candidate generation algorithms. Second, it can largely improve the accuracy of Japanese morphological analysis for non-standard written text by simultaneously performing the normalization and morphological analyses. Third, it can automatically extract character alignments and in so doing reduces the cost of manually creating many types of transformation patterns. The rest of this paper is organized as follows. Section 2 describes the background to our research, including Japanese traditional morphological analysis, related work, and data collection methods. Section 3 introduces the proposed approach, which includes lattice generation and formulation, as a discriminative model. Section 4 discusses experiments we performed and our analyses of the experimental results. Section 5 concludes the paper with a brief summary and a mention of future work.

2 Background

2.1 Japanese Morphological Analysis

Many approaches to joint word segmentation and POS tagging including Japanese Morphological analysis can be interpreted as re-ranking while using a word lattice (Kaji and Kitsuregawa (2013)). There are two points to consider in the analysis procedure: how to generate the word lattice and how to formulate the cost of each path. In Japanese morphological analysis, the dictionary-based approach has been widely used to generate the word lattice (Kudo et al. (2004), Kurohashi et al. (1994)). In a traditional approach, an optimal path is sought by using the sum of the two types of costs for the path: the cost for a candidate word that reflects the word's occurrence probability, and the cost for a pair of adjacent POS that reflects the probability of an adjacent occurrence of the pair (Kudo et al. (2004), Kurohashi et al. (1994)). A greater cost means less probability. The Viterbi algorithm is usually used for finding the optimal path.

2.2 Related Work

Several studies have been conducted on Japanese morphological analysis in the normalized form. The approach proposed by Sasano et al. (2013) aims to develop heuristics to flexibly search by using a simple, manually created derivational rule. Their system generates normalized character sequence based on the derivational rule, and adding new nodes that are generated from normalized character sequence when generating the word lattice using dictionary lookup. Figure 1 presents an example of this approach. If the non-standard written sentence すーごく楽しぃ (*suugoku tanoshii*, "It is such fun") is input, the

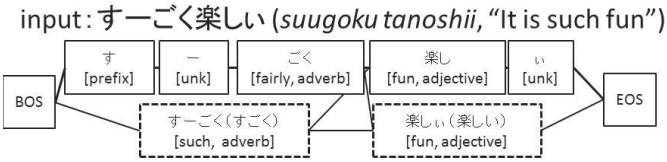


Figure 1: Example of Japanese morphological analysis and normalization

type	non-standard form	standard form
(1) Insertion	ありがと ーー う (*arigatoou*)	ありがとう (*arigatou*, "Thank you")
(2) Deletion	さむ _ (*samu*)	さむい (*samui*, "cold")
(3) Substitution with phonetic variation	かわ ええ (*kawaee*)	かわいい (*kawaii*, "cute")
(4) Substitution with lowercases and uppercases	ぁりがと ぅ (*arigatou*)	ありがとう (*arigatou*, "Thank you")
(5) Hiragana substitution	あいでぃー (*aidei*)	ID (*aidei*, "identification card")
(6) Katakana substitution	アリガトウ (*arigatou*)	ありがとう (*arigatou*, "Thank you")
(7) Any combination of (1) to (6)	かぅんたー (*kaunta*)	カウンター (*kaunta*, "counter")
	あ っ つ ぃ (*attsui*)	あつい (*atsui*, "hot")

Table 1: Types of non-standard tokens and examples of annotated data

traditional dictionary-based system generates Nodes that are described using solid lines, as shown in Fig. 1. Since "すーごく" (*suugoku*, "such") and "楽しぃ" (*tanoshii*, "fun") are OOVs, the traditional system cannot generate the correct word segments or POS tags. However, their system generates additional nodes for the OOVs, shown as broken line rectangles in Fig. 1. In this case, derivational rules that substitute "ー" with "null" and "ぃ" (*i*) with "い" (*i*) are used and the system can generate the standard forms "すごく" (*sugoku*, "such") and "楽しい" (*tanoshii*, "fun") and their POS tags. If we can generate sufficiently appropriate rules, these approaches seem to be effective. However, there are many types of derivational patterns in SNS text and it is difficult to cover all of them by hand. Moreover, it becomes a serious problem how to set the path cost for appropriately re-ranking the word lattice when the number of candidates increases. Our approach is also based on the dictionary-based approach, however, our approach is significantly dissimilar from their approach in two ways. First, we automatically generate derivational patterns (we call them transformation tables) based on the character-level alignment between non-standard tokens and their standard forms. Compared to generating the rules by hand, our approach can generate broad coverage rules. Second, we use discriminative methods to formulate a cost function. Jiang et al. (2008), Kaji and Kitsuregawa (2013) introduce several features to appropriately re-rank the added nodes. This enables our system to perform well even when the number of candidates increases.

On the other hand, several studies have applied a statistical approach. For example, Sasaki et al. (2013) proposed a character-level sequential labeling method for normalization. However, it handles only one-to-one character transformations and does not take the word-level context into account. The proposed method can handle many-to-many character transformations and takes word-level context into account, so the scope for handling non-standard tokens is different. Many studies have been done on text normalization for English; for example Han and Baldwin (2011) classifies whether or not OOVs are non-standard tokens and estimates standard forms on the basis of contextual, string, and phonetic similarities. In these studies it was assumed that clear word segmentations existed. However, since Japanese is an unsegmented language the normalization problem needs to be treated as a joint normalization, word segmentation, and POS tagging problem.

2.3 Data Collection and Analysis of Non-standard Tokens

In previous studies (Hassan and Menezes (2013), Ling et al. (2013), Liu et al. (2011)), the researchers proposed unsupervised ways to extract non-standard tokens and their standard forms. For Japanese text, however, it is very difficult to extract word pairs in an unsupervised way because there is no clear word segmentation. To address this problem we first extracted non-standard tokens from Twitter text and blog

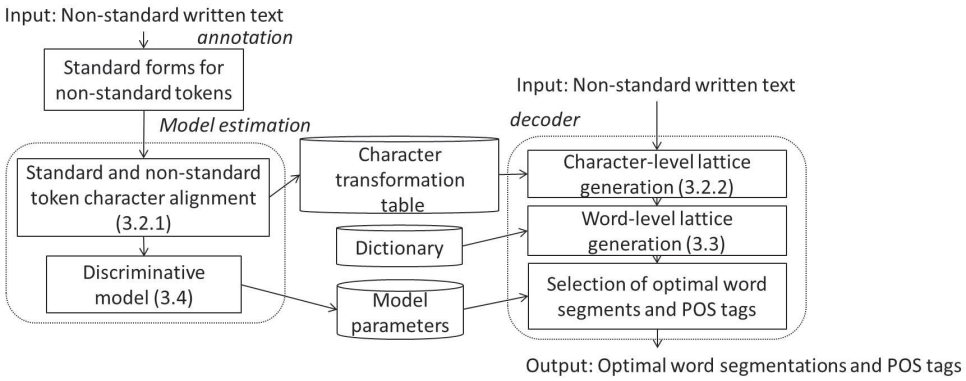


Figure 2: Structure of proposed system

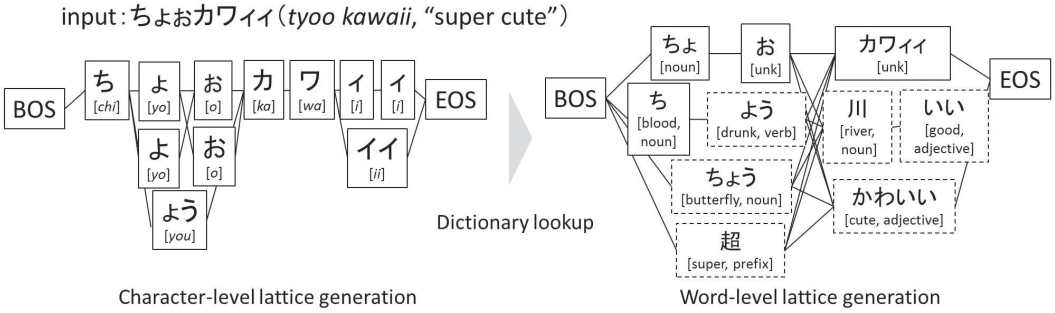


Figure 3: Example of candidate generation

text and manually annotated their standard (dictionary) forms. In total, we annotated 4808 tweets and 8023 blog text sentences. Table 1 lists the types of non-standard tokens that we targeted in this study and examples of the annotated data. Types (1), (2), (3) and (4) are similar to English transform patterns. Types (5) and (6) are distinctive patterns in Japanese. As previously mentioned Japanese has several kinds of scripts, the main ones being Kanji, Hiragana, and Katakana. These scripts can be used to write the same word in several ways. For example, the dictionary entry 先生 (*sensei*, "teacher") can also be written in Hiragana form せんせい (*sensei*) or Katakana form センセイ (*sensei*). Most words are normally written in the standard form, but in informal written text (e.g., Twitter text), these same words are often written in a non-standard form. In examining Twitter data for such non-standard tokens, we found that 55.0% of them were types (1) to (3) in Table 1, 4.5% were type (4), 20.1% were types (5) to (6), 2.7% were type (7), and the rest did not fall under any of these types since they were the result of dialects, typos, and other factors. In other words, a large majority of the non-standard tokens fell under types (1) to (7). We excluded those that did not as targets in this study because our proposed method cannot easily handle them. Types (1) to (4) occur at character-level and so can be learned from character-level alignment, but types (5) to (6) occur at word-level and it is inefficient to learn them on a character—level basis. Accordingly, we considered generating candidates and features on two levels: character-level and word-level.

3 Proposed Method

3.1 Overview of Proposed System

We showed the structure of the proposed system in Fig. 2. Our approach adds possible normalization candidates to a word lattice and finds the best sequence using a Viterbi decoder based on a discriminative model. We introduced several features that can be used to appropriately evaluate the confidence of the added nodes as normalization candidates. We generate normalization candidates as indicated in Fig. 3.

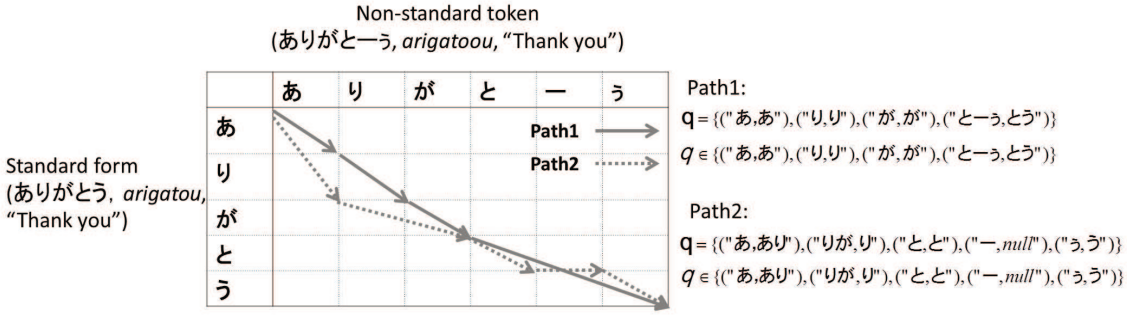


Figure 4: Example of character alignment

We describe the details in the following section.

3.2 Character-level Lattice

3.2.1 Character Alignment between Non-standard Tokens and Their Normalized Forms

We have to create a character-level transformation table to generate the character-level lattice. We used the joint multigram model proposed by Sittichai et al. (2007) to create the transformation table because this model can handle many-to-many character alignments between two character sequences. In observing non-standard tokens and their standard forms, we find there are not only one-to-one character transformations but also many-to-many character transformations. Furthermore, unlike in translation, there is no character reordering so the problems that arise are similar to those in transliteration. Accordingly, we adopted a joint multigram model that is widely used for transliteration problems. The optimal alignment can be formulated as $\hat{\mathbf{q}} = \arg\max_{\mathbf{q} \in K_d} \prod_{q \in \mathbf{q}} p(q)$, where d is a pair of non-standard tokens and its standard form (e.g., d is ありがとーぅ (*arigatoou*), ありがとう (*arigatou*). Here, q is a partial character alignment in d (e.g., q is "とーぅ, とう"), $\mathbf{q}$ is the character alignment q set in d (e.g., $\mathbf{q}$ of path 1 in Fig. 4 is {("あ, あ"), ("り, り"), ("が, が"), ("とーぅ, とう")}. K_d is the possible character alignment sequence candidates generated from d. We generate n-best optimal path for K_d in this study. The maximum likelihood training can be performed using the EM algorithm derivated in Bisani and Ney (2008) and Kubo et al. (2011) to estimate $p(q)$. $p(q)$ can be formulated as follow:

$$p(q) = \gamma_q / \sum_{q \in Q} \gamma_q \tag{1}$$

$$\gamma_q = \sum_{d \in D} \sum_{\mathbf{q} \in K_d} p(\mathbf{q}) n_q(\mathbf{q}) = \sum_{d \in D} \sum_{\mathbf{q} \in K_d} \frac{\prod_{q \in \mathbf{q}} \bar{p}(q)}{\sum_{\mathbf{q} \in K_d q \in \mathbf{q}} \prod \bar{p}(q)} n_q(\mathbf{q}),$$

and where D is the number of the d pair, Q is the set of q, and $n_q(\mathbf{q})$ is the count of q that occurred in $\mathbf{q}$. In our system, we allow for standard form deletions (i.e., mapping of a non-standard character to a null standard character) but not non-standard token deletions. Since we use this alignment as the transformation table when generating a character-level lattice, the lattice size becomes unnecessarily large if we allow for non-standard form deletions. In the calculation step of the EM algorithm, we calculate the expectation (partial counts) γ_q of each alignment in the E-step, calculate the joint probability $p(q)$ that maximizes the likelihood function in the M-step as described before, and repeat these steps until convergence occurs. $\bar{p}(q)$ indicates the result of $p(q)$ calculated in the previous step over the iteration. When generating the character-level lattice, we used alignments that were expected to exceed a predefined threshold. We used γ_q ($q = (c_t, c_v)$) and $r(c_t, c_v)$ as thereshold, where c_t and c_v are the partial character sequence of non-standard token and it's standard form respectively. $r(c_t, c_v)$ is calculated by $r(c_t, c_v) = \gamma_q / n_{c_v}$., where n_{c_v} is the number of occurrences of c_v in the training data. We set the threshold $\gamma_{q\_thres} = 0.5$, and $r(c_t, c_v)_{thres} = 0.0001$ in this study. We also used $r(c_t, c_v)$ as a feature of cost

function in subsection. 3.4.2. When calculating initial value, we set $p(c_t, c_v)$ high if the character c_t and c_v are the same character and the length of each character is 1. We also give the limitation that a Kanji character does not change to a different character and is aligned with same character in the calculation step of the character alignment.

3.2.2 Generation of Character-level Lattice Based on Transformation Table

First, repetitions of more than one letter of "ー", "〜", "-", and "っ" are reduced back to one letter (e.g., ありがと ーーー う (*arigatooooou*, "Thank you") is reduced to ありがと ー う (*arigatoou*)) for the input text. In addition, repetitions of more than three letters other than "ー", "〜", "-", and "っ" are reduced back to three letters (e.g., うれし いいいいい (*uresiiiiii*, "I'm happy") is reduced back to うれし いいい (*uresiii*)). These preprocessing rules are inspired by Han and Baldwin (2011) and determined by taking the Japanese characteristics into consideration. We also used these rules when we estimated the alignments of the non-standard tokens and their standard forms. Next, we generate the character-level normalization candidates if they match the key transformation table in the input text. For example, if the transformation table contains (q, log$p(q)$)= ("よお (*yoo*), よう (*you*)", -8.39), ("お (*o*), お (*o*)", -7.56), and the input text includes the character sequence "ちよお" (*tyoo*), we generate a new sequence "ちょう" (*tyou*) and "ちょお" (*tyoo*). In other words, we add new nodes "ょう" (*you*) and "お" (*o*) in the position of "よお" (*yoo*) and "お" (*o*), respectively (see Fig. 3).

3.3 Generation of Word-level Lattice

We generate the word lattice based on the generated character-level lattice using dictionary lookup. We exploit dictionary lookup by using the possible character sequence of the character-level lattice while the traditional approach exploits it by using only the input character sequence. For example, we exploit dictionary lookup for character sequences such as "ちよおカワィィ" (*tyoo kawaii*) and "ちょう カワィィ" (*tyou kawaii*) and "ち よおカワィィ" (*chiyou kawaii*) and "ちよおカワ イイ" (*tyoo kawaii*) (see Fig. 3)

Furthermore, we use the phonetic information of the dictionary to generate the normalization candidates for Hiragana and Katakana substitution. For example, assume "超" (*tyou*, "super") and "かわいい" (*kawaii*, "cute") are the dictionary words. Then, if the input text contains the character sequences "ちょう" (*tyo*) (which is written in Hiragana) and "カワイイ" (*kawaii*) (which is written in Katakana), we add "超" (*tyo*, "super") and "かわいい" (*kawaii*, "cute") to the word lattice as the normalization candidates since the two character sequences are pronounced identically. By using this two-step algorithm, we can handle any combinational derivational patterns, such as Katakana substitutions or substitutions of lowercases like "カワィィ" (*kawaii*) → "カワイイ" (*kawaii*) → "かわいい" (*kawaii*, "cute") (see Fig. 3). Note that we filtered candidates on the basis of a predefined threshold to prevent the generation of unnecessary candidates. The threshold was defined on the basis of the character sequence cost of normalization, which is described in subsection 3.4.2. Furthermore, we limited the number of character transformations to two per word.

3.4 Decoder

3.4.1 Objective Function

The decoder selects the optimal sequence $\hat{y}$ from $L(s)$ when given the candidate set $L(s)$ for sentence s. This is formulated as $\hat{y} = \arg\min_{y \in L(s)} \mathbf{w} \cdot \mathbf{f}(y)$ (Jiang et al. (2008), Kaji and Kitsuregawa (2013)), where $\hat{y}$ is the optimal path, $L(s)$ is the lattice created for sentence s, and $\mathbf{w} \cdot \mathbf{f}(y)$ is the dot product between weight vector $\mathbf{w}$ and feature vector $\mathbf{f}(y)$. The optimal path is selected according to the $\mathbf{w} \cdot \mathbf{f}(y)$ value.

3.4.2 Features

The proposed lattice generation algorithm generates a lattice larger than that generated in traditional dictionary-based lattice generation. Therefore, we need to introduce an appropriate normalization cost into the objective function. We listed the features we used in Table 2. Let w_i be the ith word candidate and p_i be the POS tag of w_i. p_{i-1} and w_{i-1} are adjacent POS tag and word respectively. We also used the word unigram cost $f_{w_i p_i}$, the cost for a pair of adjacent POS f_{p_{i-1}, p_i} that are quoted from MeCab (Kudo,

Name	Feature
Word unigram cost	$f_{w_i p_i}$
POS bi-gram cost	f_{p_{i-1},p_i}
Word-POS bi-gram cost	$-\log p_{w_{i-1}p_{i-1},w_i p_i}$
Character sequence cost	$\log(p'_s/p'_{t_i})$
	where, $p'_x = p_x^{1/length(x)}$, $p_x = \prod_{j=1}^{n} p(c_j \mid c_{j-5}^{j-1})$, $x \in \{s, t_i\}$
Character transformation cost	$\phi_{trans_i} \cdot (-\log r(c_t, c_v))$
Hiragana substitution cost	$\phi_{h_i} \cdot f_{w_i p_i}$
Katakana substitution cost	$\phi_{k_i} \cdot f_{w_i p_i}$

Table 2: Feature list of the decoder. ϕ_{trans_i} is 1 if w_i is generated by character transformation, otherwise 0. ϕ_{h_i} is 1 if w_i is generated by Hiragana substitution, otherwise 0. ϕ_{k_i} is 1 if w_i is generated by Katakana substitution, otherwise 0.

2005), and five additional types of costs. These are the word-pos bi-gram cost $-\log p_{w_{i-1}p_{i-1},w_i p_i}$ of a blog corpus; the character transformation cost $\phi_{trans_i} \cdot (-\log r(c_t, c_v))$, which is calculated in Section3.2, for nodes generated by character transformation; the Hiragana substitution cost $\phi_{h_i} \cdot f_{w_i p_i}$ for nodes generated by Hiragana substitution; the Katakana substitution cost $\phi_{k_i} \cdot f_{w_i p_i}$ for nodes generated by Katakana substitution; and the character sequence cost $\log(p'_s/p'_{t_i})$ for all the normalized nodes. The character sequence cost reflects the character sequence probability of the normalization candidates. Here, s and t_i are input string and transformed string respectively. (e.g., In Fig. 3, for the normalized node "かわいい" (cute, adjective), s is "ちよおカワイイ" and t_i is "ちよおかわいい"). Then p_s and p_{t_i} are calculated by using the character 5-gram of a blog corpus, which is formulated by $p_s = p(c_1 \cdots c_n) = \prod_{j=1}^{n} p(c_j \mid c_{j-5}^{j-1})$, where c_j is the j th character of character sequence s. p'_{t_i} and p'_s are normalized by using the length of each string s and t_i as $p'_{t_i} = p_{t_i}^{1/length(t_i)}$. We set the threshold $(p'_s/p'_{t_i})_{thres} = 1.5$ for generating a Hiragana or Katakana normalization candidate in this study. Since all those features can be factorized, the optimal path is searched for by using the Viterbi algorithm.

3.4.3 Training

We formulated the objective function for tuning weights $\mathbf{w}$ by using Eq. 2. The weights $\mathbf{w}$ are trained by using the minimum error rate training (MERT) Machery et al. (2008). We defined the error function as the differences between the reference word segmentations and the POS tags of the reference sequence y_{ref} and the system output $\arg\min_{y \in L(s)} \mathbf{w} \cdot \mathbf{f}(y)$.

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathbf{W}} \sum_{i=1}^{N} \mathrm{error}(y_{ref}, \arg\min_{y \in L(s)} \mathbf{w} \cdot \mathbf{f}(y)) \quad (2)$$

4 Experiments

4.1 Dataset and Estimated Transformation Table

We conducted experiments to confirm the effectiveness of the proposed method, in which we annotated corpora of a Japanese blog and Twitter. The Twitter corpus was split into three parts: the training, development, and test sets. The test data comprised 300 tweets, development data comprised 500 sentences and the training data comprised 4208 tweets. We randomly selected the test data which contained at least one non-standard token. The test data comprised 4635 words, 403 words of them are non-standard token and are orthographically transformed into normalized form and POS tags. The blog corpus comprised 8023 sentences and all of them were used as training data. Training data was used for extracting character transformation table and development data was used for estimating parameters of discriminative model. We used the IPA dictionary provided by MeCab to generate the word-level lattice and extracted the dictionary-based features. We itemized the estimated character transformation patterns in Table 3. There were 5228 transformation patterns that were learned from the training data and we used 3268 of them, which meets the predefined condition. The learned patterns cover most of the previously pro-

non-standard character c_t	standard character c_v	$\log p(q)$	non-standard character c_t	standard character c_v	$\log p(q)$
ー	null	-4.233	っす (*ssu*)	です (*desu*)	-5.999
まぁ(*maa*)	まあ (*maa*)	-5.059	どー (*doo*)	どう (*dou*)	-6.210
しょ(*syo*)	しょう (*syou*)	-5.211	ねー (*nee*)	ない (*nai*)	-6.232
だろ (*daro*)	だろう (*darou*)	-5.570	りゃ(*rya*)	れは (*reha*)	-6.492
っ(*ttsu*)	null	-5.648	てん (*ten*)	てる (*teru*)	-6.633
んと (*nto*)	んとう (*ntou*)	-5.769	ゆう (*yuu*)	いう (*iu*)	-6.660
わ(*wa*)	は (*wa*)	-5.924	なん (*nan*)	なの (*nano*)	-6.706

Table 3: Example of character-level transformation table

posed rules. In addition, our method can learn more of the variational patterns that are difficult to create manually.

4.2 Baseline and Evaluation Metrics

We compared the five methods listed in Table 4 in our experiments. Traditional means that which generates no normalization candidates and only uses the word cost and the cost for a pair of adjacent POS, so we can consider it as a traditional Japanese morphological analysis. We compared three baselines, Baseline1, Baseline2 and Baseline3. Baseline1 is the conventional rule-based method (considering insertion of long sound symbols and lowercases, and substitution with long sound symbols and lowercases), which was proposed by Sasano et al. (2013). In Baseline2, 3, and Proposed, we basically use the proposed discriminative model and features, but there are several differences. Baseline2 only generates character-level normalization candidates. Baseline3 uses our two-step normalization candidate generation algorithms, but the character transformation cost of all the normalization candidates that are generated by character normalization is the same. Proposed generates the character-level and Hiragana and Katakana normalization candidates and use all features we proposed.

We evaluated each method on the basis of precision and recall and the F-value for the overall system accuracy. Since Japanese morphological analysis simultaneously estimates the word segmentation and POS tagging, we have to check whether or not our system is negatively affected by anything other than the non-standard tokens. We also evaluated the recall with considering only normalized words. That value directly reflects the performance of our normalization method. We registered emoticons that occurred in the test data in the dictionary so that they would not negatively affect the systems' performance.

4.3 Results and Discussion

The results are classified in Table 4. As the table shows, the proposed methods performed statistically significantly better than the baselines and the traditional method in both precision and recall ($p < 0.01$), where the precision was greatly improved. This indicates that our method can not only correctly analyze the non-standard tokens, but can also reduce the number of wrong words generated. Baseline1 also improved the accuracy and recall compared to the traditional method, but the effect was limited. When we compare Proposed with Baseline2, we find the F-value is improved when we take the Hiragana and Katakana substitution into consideration. Baseline3 also improved the F-value but its performance is inferior to proposed method.This proves that even if we can generate sufficient normalization candidates, the results worsen if the weight parameter of each normalization candidate is not appropriately tuned. The column of "recall*" in Table 4 specifies the improvement rates of the non-standard tokens. The proposed methods improve about seven times when using Baseline1 while preventing degradation. These results prove that we have to generate appropriate and sufficient normalization candidates and appropriately tune the cost of each candidate to improve both the precision and recall.

We show examples of the system output in Table 5. In the table, slashes indicate the position of the estimated word segmentations and the words that were correctly analyzed are written in bold font. Examples (1) to (5) are examples improved by using the proposed method. Examples (6) to (7) are examples that were not improved and example (8) is an example that was degraded. Examples (1) to (3) include phonetic variations and example (4) is a Hiragana substitution. Example (5) is a combinational trans-

method	word segmentation			word segmentation and POS tag			
	precision	recall	F-value	precision	recall	F-value	recall*
Traditional	0.716	0.826	0.767	0.683	0.788	0.732	-
Rule based (BL1**)	0.753	0.833	0.791	0.717	0.794	0.754	0.092
Proposed	**0.856**	**0.883**	**0.869**	**0.822**	**0.849**	**0.835**	**0.667**
- without Hiragana and Katakana normalization (BL2)	0.834	0.875	0.854	0.798	0.838	0.818	0.509
- character transformation cost is fixed (BL3)	0.838	0.865	0.851	0.807	0.834	0.821	0.533

* considering only normalized words, ** BL:baseline

Table 4: Results of precision and recall of test data

input	traditional	proposed	gold standard
(1) あ ぢー(*adii*)	あ (*a*)/ぢ (*di*)/ー	**あつい** (*atsui*)	あつい (*atsui*, “hot”)
(2) す げー(*sugee*)	すげ (suge)/ー	**すごい** (*sugoi*)	すごい (*sugoi*, “great”)
(3) ご っめ ー ん (*gommeen*)	ご (*go*)/っ/め (*me*)/ー/ん (*n*)/	**ごめん** (*gomen*)	ごめん (*gomen*, “I’m sorry”)
(4)ひつよう(*hitsuyou*)	ひつ (*hitsu*)/よう (*you*)	**必要** (*hitsuyou*)	必要 (*hitsuyou*, “necessary”)
(5)だいちゅき(*daichuki*)	だ (*da*)/いち (*ichi*)/ゅ(*yu*)/き (*ki*)/	**大好き** (*daisuki*)	大好き (*daisuki*, “like very much”)
(6) お せえええ(*oseee*)	おせ (*ose*)/ええ (*ee*)/え (*e*)	おせ (*ose*)	おそい (*osoi*, “slow”)
(7) か ん わいい (*kanwaii*)	かん (*kan*)/わ (*wa*)/いい (*ii*)	官話 (*kanwa*)/いい (*ii*)	かわいい (*kawaii*, “cute”)
(8) いない (*inai*)	**い (*i*)/ない (*nai*)**	以内 (*inai*)	い/ない (*i*/*nai*, “absent”)

Table 5: System output examples

formation pattern of a phonetic variation and Hiragana substitution. We can see our system can analyze such variational non-standard tokens for all these examples. Two types of errors were identified. The first occurred as the result of a lack of a character transformation pattern and the second was search errors. Example (6) shows an example of a case in which our system couldn’t generate correct normalization candidate because there was not corresponding character transformation pattern, even though there was a similar phonetic transformation pattern. To ensure there will be no lack of transformation patterns, we should either increase the parallel corpus size to enable the learning of more patterns or derive new transformation patterns from the learned patterns. Example (7) shows an example of a case in which a normalized candidate was generated but a search failed to locate it. Example (8) shows an example of a case in which the result was degraded. Our system can control the degradation well, but there are several degradation caused by normalization. We will need to develop a more complicated model or introduce other features into the current model to reduce the number of search errors.

5 Conclusion and Future Work

We introduced a text normalization approach into joint Japanese morphological analysis and showed that our two-step lattice generation algorithm and formulation using discriminative methods outperforms the previous method. In future work, we plan to extend this approach by introducing an unsupervised or semi-supervised parallel corpus extraction for learning character alignments to generate more patterns at a reduced cost. We also plan to improve our model’s structure and features and implement it with a decoding method to reduce the number of search errors. In addition, we should consider adding other types of unknown words (such as named entities) to the morphological analysis system to improve its overall performance.

References

AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for sms text normalization. *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, pages 33–40.

Maximilian Bisani and Hermann Ney. 2008. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Commun.*, 50(5):434–451, May.

Paul Cook and Suzanne Stevenson. 2009. An unsupervised model for text message normalization. *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 71–78.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 368–378.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432.

Hany Hassan and Arul Menezes. 2013. Social text normalization using contextual graph random walks. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1577–1586, August.

Wenbin Jiang, Haitao Mi, and Qun Liu. 2008. Word lattice reranking for chinese word segmentation and part-of-speech tagging. *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, pages 385–392.

Nobuhiro Kaji and Masaru Kitsuregawa. 2013. Efficient word lattice generation for joint word segmentation and pos tagging in japanese. *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 153–161.

Keigo Kubo, Hiromichi Kawanami, Hiroshi Saruwatari, and Kiyohiro Shikano. 2011. Unconstrained many-to-many alignment for automatic pronunciation annotation. *In Proc. of APSIPA ASC*.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to japanese morphological analysis. *In Proc. of EMNLP*, pages 230–237.

T. Kudo. 2005. Mecab : Yet another part-of-speech and morphological analyzer. *http://mecab.sourceforge.net/*.

Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto, and Makoto Nagao. 1994. Improvements of japanese morphological analyzer juman. *In Proc. of The International Workshop on Sharable Natural Language Resources*, page 22—38.

Chen Li and Yang Liu. 2012. Improving text normalization using character-blocks based models and system combination. *Proceedings of COLING 2012*, pages 1587–1602.

Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. 2013. Paraphrasing 4 microblog normalization. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 73–84, October.

Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011. Insertion, deletion, or substitution? normalizing text messages without pre-categorization nor supervision. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 71–76, June.

Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1035–1044.

W Machery, F J Och, and I Uszkoreit J Thayer. 2008. Lattice-based minimum error rate training for statistical machine translation. *In Proc. of EMNLP*, 1:725–734.

Teruaki Oka, Mamoru Komachi, Toshinobu Ogiso, and Yuji Matsumoto. 2011. Handling orthographic variations in morphological analysis for near-modern japanese (in japanese). *In Proc. of The 27th Annual Conference of the Japanese Society for Articial Intelligence*.

Deana Pennell and Yang Liu. 2011. A character-level machine translation approach for normalization of sms abbreviations. *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 974–982, November.

Akira Sasaki, Junta Mizuno, Naoaki Okazaki, and Kentaro Inui. 2013. Normalization of text in microblogging based on machine learning(in japanese) (in japanese). *In Proc. of The 27th Annual Conference of the Japanese Society for Articial Intelligence*.

Ryohei Sasano, Sadao Kurohashi, and Manabu Okumura. 2013. A simple approach to unknown word processing in japanese morphological analysis. *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 162–170.

Jiampojamarn Sittichai, Kondrak Grzegorz, and Sherif Tarek. 2007. Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. *In Proc. of The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 372–379.

Adapting taggers to Twitter with not-so-distant supervision

Barbara Plank[1], **Dirk Hovy**[1], **Ryan McDonald**[2] **and Anders Søgaard**[1]
Center for Language Technology, University of Copenhagen[1]
Google Inc.[2]
{bplank,dirkh}@cst.dk,ryanmcd@google.com,soegaard@hum.ku.dk

Abstract

We experiment with using different sources of distant supervision to guide unsupervised and semi-supervised adaptation of part-of-speech (POS) and named entity taggers (NER) to Twitter. We show that a particularly good source of not-so-distant supervision is linked websites. Specifically, with this source of supervision we are able to improve over the state-of-the-art for Twitter POS tagging (89.76% accuracy, 8% error reduction) and NER (F1=79.4%, 10% error reduction).

1 Introduction

Twitter contains a vast amount of information, including first stories and breaking news (Petrovic et al., 2010), fingerprints of public opinions (Jiang et al., 2011) and recommendations of relevance to potentially very small target groups (Benson et al., 2011). In order to automatically extract this information, we need to be able to analyze tweets, e.g., determine the part-of-speech (POS) of words and recognize named entities. Tweets, however, are notoriously hard to analyze (Foster et al., 2011; Eisenstein, 2013; Baldwin et al., 2013). The challenges include dealing with variations in spelling, specific conventions for commenting and retweeting, frequent use of abbreviations and emoticons, non-standard syntax, fragmented or mixed language, etc.

Gimpel et al. (2011) showed that we can induce POS tagging models with high accuracy on in-sample Twitter data with relatively little annotation effort. Learning taggers for Twitter data from small amounts of labeled data has also been explored by others (Ritter et al., 2011; Owoputi et al., 2013; Derczynski et al., 2013). Hovy et al. (2014), on the other hand, showed that these models overfit their respective samples and suffer severe drops when evaluated on out-of-sample Twitter data, sometimes performing even worse than newswire models. This may be due to drift on Twitter (Eisenstein, 2013) or simply due to the heterogeneous nature of Twitter, which makes small samples biased. So while existing systems perform well on their own (in-sample) data sets, they over-fit the samples they were induced from, and suffer on other (out-of-sample) Twitter data sets. This bias can, at least in theory, be corrected by learning from additional unlabeled tweets. This is the hypothesis we explore in this paper.

We present a semi-supervised learning method that does not require additional labeled in-domain data to correct sample bias, but rather leverages pools of unlabeled Twitter data. However, since taggers trained on newswire perform poorly on Twitter data, we need additional guidance when utilizing the unlabeled data. This paper proposes distant supervision to help our models learn from unlabeled data. Distant supervision is a weakly supervised learning paradigm, where a knowledge resource is exploited to gather (possible noisy) training instances (Mintz et al., 2009). Our basic idea is to can use linguistic analysis of linked websites as a novel kind of distant supervision for learning how to analyze tweets. We explore standard sources of distant supervision, such as Wiktionary for POS tagging, but we also propose to use the linked websites of tweets with URLs as supervision. The intuition is that we can use websites to provide a richer linguistic context for our tagging decisions. We exploit the fact that tweets with URLs provide a one-to-one map between an unlabeled instance and the source of supervision, making this

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1783–1792, Dublin, Ireland, August 23-29 2014.

1: $X = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{N}$ labeled tweets
2: $U = \{\langle \mathbf{x}_i, w_i \rangle\}_{i=1}^{M}$ unlabeled tweet-website pairs
3: I iterations
4: $k = 1000$ pool size
5: **v=train**(X) base model
6: **for** $i \in I$ **do**
7: **for** $\langle \mathbf{x}, w \rangle \in \mathbf{pool}_k(U)$ **do**
8: $\hat{y}$=**predict**$(\langle \mathbf{x}, w \rangle; \mathbf{v})$
9: $X \leftarrow X \cup \{\langle \hat{y}, \mathbf{x} \rangle\}$
10: **end for**
11: **v=train**(X)
12: **end for**
13: **return** **v**

Figure 1: Semi-supervised learning with not-so-distant supervision, i.e. tweet-website pairs $\{\langle \mathbf{x}_i, w_i \rangle\}$. SELF-TRAINING, WEB, DICT, DICT$\prec$WEB and WEB$\prec$DICT differ only in how **predict()** (line 8) is implemented (cf. Section 2).

less distant supervision. Note that we use linked websites only for semi-supervised learning, but do *not* require them at test time.

Our semi-supervised learning method enables us to learn POS tagging and NER models that perform more robustly across different samples of tweets than existing approaches. We consider both the scenario where a small sample of labeled Twitter data is available, and the scenario where only newswire data is available. Training on a mixture of out-of-domain (WSJ) and in-domain (Twitter) data as well as unlabeled data, we get the best reported results in the literature for both POS tagging and NER on Twitter. Our tagging models are publicly available at `https://bitbucket.org/lowlands/ttagger-nsd`

2 Tagging with not-so-distant supervision

We assume that our labeled data is highly biased by domain differences (Jiang and Zhai, 2007), population drift (Hand, 2006), or by our sample size simply being too small. To correct this bias, we want to use unlabeled Twitter data. It is well-known that semi-supervised learning algorithms such as self-training sometimes effectively correct model biases (McClosky et al., 2006; Huang et al., 2009). This paper presents an augmented self-training algorithm that corrects model bias by exploiting unlabeled data and not-so-distant supervision. More specifically, the idea is to use hyperlinks to condition tagging decisions in tweets on a richer linguistic context than what is available in the tweets. This semi-supervised approach gives state-of-the-art performance across available Twitter POS and NER data sets.

The overall semi-supervised learning algorithm is presented in Figure 1. The aim is to correct model bias by predicting tag sequences on small pools of unlabeled tweets, and re-training the model across several iterations to gradually correct model bias. Since information from hyperlinks will be important, the unlabeled data U is a corpus of tweets containing URLs. We present a baseline and four system proposals that only differ in their treatment of the **predict()** function.

In the SELF-TRAINING baseline, **predict()** corresponds to standard Viterbi inference on the unlabeled Twitter data. This means, the current model $\mathbf{v}$ is applied to the tweets by disregarding the websites in the tweet-website pairs, i.e., tagging $\mathbf{x}$ without considering w. Then the automatically tagged tweets are added to the current pool of labeled data and the procedure is iterated (line 7-11 in Figure 1).

In the WEB method, we additionally use the information from the websites. The current model $\mathbf{v}$ is used to predict tags for the pooled tweets *and* the website they linked to. For all the words that occur both in a tweet and on the corresponding website, we then project the tag most frequently assigned to those words on the website to their occurrences in the tweet. This enables us to basically condition the tag decision for each such word on its accumulated context on the website. The assumption of course being that the word in the tweet has the part-of-speech it most often has on the website linked to.

Example Here is an example of a tweet that contains a URL:

(1) #Localization #job: Supplier / Project Manager - Localisation Vendor - NY, NY, United States http://bit.ly/16KigBg #nlppeople

The words in the tweet are all common words, but they occur without linguistic context that could help a tagging model to infer whether these words are nouns, verbs, named entities, etc. However, on the website that the tweet refers to, all of these words occur in context:

(2) The Supplier/Project Manager performs the selection and maintenance ...

For illustration, the Urbana-Champaign POS tagger[1] incorrectly tags *Supplier* in (1) as an adjective. In (2), however, it gets the same word right and tags it as a noun. The tagging of (2) could potentially help us infer that *Supplier* is also a noun in (1).

Obviously, the superimposition of tags in the WEB method may change the tag of a tweet word such that it results in an unlikely tag sequence, as we will discuss later. Therefore we also implemented type-constrained decoding (Täckström et al., 2013), i.e., prune the lattice such that the tweet words observed on the website have *one of* the tags they were labeled with on the website (soft constraints), or, alternatively, were forced during decoding to have the most frequent tags they were labeled with (hard constraint decoding), thereby focusing on licensed sequences. However, none of these approaches performed significantly better than the simple WEB approach on held-out data. This suggests that sequential dependencies are less important for tagging Twitter data, which is of rather fragmented nature. Also, the WEB approach allows us to override transitional probabilities that are biased by the observations we made about the distribution of tags in our out-of-domain data.

Furthermore, we combine the not-so-distant supervision from linked websites (WEB) with supervision from dictionaries (DICT). The idea here is to exploit the fact that many word types in a dictionary are actually unambiguous, i.e., contain only a single tag. In particular, 93% of the word types in Wiktionary[2] are unambiguous. Wiktionary is a crowdsourced tag dictionary that has previously been used for minimally supervised POS tagging (Li et al., 2012; Täckström et al., 2013). In the case of NER, we use a gazetteer that combines information on PER, LOC and ORG from the KnownLists of the Illinois tagger.[3] For this gazetteer, 79% of the word types contained only a single named entity tag.

We experiment with a model that uses the dictionary only (DICT) and two ways to combine the two sources. In the former setup, the current model is first applied to tag the tweets, then any token that appears in the dictionary and is unambiguous is projected back to the tweet. The next two methods are combinations of WEB and DICT: either first project the predicted tags from the website and then, in case of conflicts, overrule predictions by the dictionary (WEB$\prec$DICT), or the other way around (DICT$\prec$WEB).

The intuition behind the idea of using linked websites as not-so-distant supervision is that while tweets are hard to analyze (even for humans) because of the limited context available in 140 character messages, tweets relate to real-world events, and Twitter users often use hyperlinks to websites to indicate what real-world events their comments address. In fact, we observed that about 20% of tweets contain URLs. The websites they link to are often newswire sites that provide more context and are written in a more canonical language, and are therefore easier to process. Our analysis of the websites can then potentially inform our analysis of the tweets. The tweets with the improved analyses can then be used to bootstrap our tagging models using a self-training mechanism. Note that our method *does not* require tweets to contain URLs at test time, but rather uses unlabeled tweets with URLs during training to build better tagging models for tweets in general. At test time, these models can be applied to any tweet.

[1] http://cogcomp.cs.illinois.edu/demo/pos/

[2] http://en.wiktionary.org/ - We used the Wiktionary version derived by Li et al. (2012).

[3] http://cogcomp.cs.illinois.edu/page/software_view/NETagger

3 Experiments

3.1 Model

In our experiments we use a publicly available implementation of conditional random fields (CRF) (Lafferty et al., 2001).[4] We use the features proposed by Gimpel et al. (2011), in particular features for word tokens, a set of features that check for the presence of hyphens, digits, single quotes, upper/lowercase, 3 character prefix and suffix information. Moreover, we add Brown word cluster features that use 2^i for $i \in 1, ..., 4$ bitstring prefixes estimated from a large Twitter corpus (Owoputi et al., 2013), which is publicly available.[5] We use a pool size of 1000 tweets. We experimented with other pool sizes {500,2000} showing similar performance. The number of iterations i is set on the development data.

For NER on websites, we use the Stanford NER system (Finkel et al., 2005)[6] with POS tags from the LAPOS tagger (Tsuruoka et al., 2011).[7] For POS we found it to be superior to use the current POS model for re-tagging websites; for NER it was slightly better to use the Stanford NER tagger and thus off-line NER tagging rather than retagging the websites in every iteration.

3.2 Data

In our experiments, we consider two scenarios, sometimes referred to as unsupervised and semi-supervised domain adaptation (DA), respectively (Daumé et al., 2010; Plank, 2011). In unsupervised DA, we assume only (labeled) newswire data, in semi-supervised DA, we assume labeled data from both domains, besides unlabeled target data, but the amount of labeled target data is much smaller than the labeled source data. Most annotated corpora for English are newswire corpora. Some annotated Twitter data sets have been made available recently, described next.

	POS	**NER**
train	WSJ (700k)	REUTER-CONLL (Tjong Kim Sang and De Meulder, 2003) (200k)
	GIMPEL-TRAIN (Owoputi et al., 2013) (14k)	FININ-TRAIN (Finin et al., 2010) (170k)
dev	FOSTER-DEV (Foster et al., 2011) (3k)	n/a
	RITTER-DEV (Ritter et al., 2011) (2k)	n/a
test	FOSTER-TEST (Foster et al., 2011) (2.8k)	RITTER-TEST (Ritter et al., 2011) (46k)
	GIMPEL-TEST (Gimpel et al., 2011) (7k)	FININ-TEST (Finin et al., 2010) (51k)
	HOVY-TEST (Hovy et al., 2014)	FROMREIDE-TEST (Fromreide et al., 2014) (20k)

Table 1: Overview of data sets. Number in parenthesis: size in number of tokens.

Training data. An overview of the different data sets is given in Table 3.2. In our experiments, we use the SANCL shared task[8] splits of the OntoNotes 4.0 distribution of the WSJ newswire annotations as newswire training data for POS tagging.[9] For NER, we use the CoNLL 2003 data sets of annotated newswire from the Reuters corpus.[10] The in-domain training POS data comes from Gimpel et al. (2011), and the in-domain NER data comes from Finin et al. (2010) (FININ-TRAIN). These data sets are added to the newswire sets when doing semi-supervised DA. Note that for NER, we thus do not rely on expert-annotated Twitter data, but rely on crowdsourced annotations. We use MACE[11] (Hovy et al., 2013) to resolve inter-annotator conflicts between turkers (50 iterations, 10 restarts, no confidence threshold). We believe relying on crowdsourced annotations makes our set-up more robust across different samples of Twitter data.

Development and test data. We use several evaluation sets for both tasks to prevent overfitting to a specific sample. We use the (out-of-sample) development data sets from Ritter et al. (2011) and Foster

[4] http://www.chokkan.org/software/crfsuite/
[5] http://www.ark.cs.cmu.edu/TweetNLP/
[6] http://http://nlp.stanford.edu/software/CRF-NER.shtml
[7] http://www.logos.ic.i.u-tokyo.ac.jp/~tsuruoka/lapos/
[8] https://sites.google.com/site/sancl2012/home/shared-task
[9] LDC2011T03.
[10] http://www.clips.ua.ac.be/conll2003/ner/
[11] http://www.isi.edu/publications/licensed-sw/mace/

et al. (2011). For NER, we simply use the parameters from our POS tagging experiments and thus do not assume to have access to further development data. For both POS tagging and NER, we have three test sets. For POS tagging, the ones used in Foster et al. (2011) (FOSTER-TEST) and Ritter et al. (2011) (RITTER-TEST),[12] as well as the one presented in Hovy et al. (2014) (HOVY-TEST). For NER, we use the data set from Ritter et al. (2011) and the two data sets from Fromreide et al. (2014) as test sets. One is a manual correction of a held-out portion of FININ-TRAIN, named FININ-TEST; the other one is referred to as FROMREIDE-TEST. Since the different POS corpora use different tag sets, we map all of them corpora onto the universal POS tag set by Petrov et al. (2012). The data sets also differ in a few annotation conventions, e.g., some annotate URLs as NOUN, some as X. Moreover, our newswire tagger baselines tend to get Twitter-specific symbols such as URLs, hashtags and user accounts wrong. Instead of making annotations more consistent across data sets, we follow Ritter et al. (2011) in using a few post-processing rules to deterministically assign Twitter-specific symbols to their correct tags. The major difference between the NER data sets is whether Twitter user accounts are annotated as PER. We follow Finin et al. (2010) in doing so.

Unlabeled data We downloaded 200k tweet-website pairs from the Twitter search API over a period of one week in August 2013 by searching for tweets that contain the string *http* and downloading the content of the websites they linked to. We filter out duplicate tweets and restrict ourselves to websites that contain more than one sentence (after removing boilerplate text, scripts, HTML, etc).[13] We also require website and tweet to have at least one matching word that is not a stopword (as defined by the NLTK stopword list).[14] Finally we restrict ourselves to pairs where the website is a subsite, because website head pages tend to contain mixed content that is constantly updated. The resulting files are all tokenized using the Twokenize tool.[15] Tweets were treated as one sentence, similar to the approaches in Gimpel et al. (2011) and Owoputi et al. (2013); websites were processed by applying the Moses sentence splitter.[16]

The out-of-vocabulary (OOV) rates in Figure 2 show that in-domain training data reduces the number of unseen words considerably, especially in the NER data sets. They also suggest that some evaluation data sets share more vocabulary with our training data than others. In particular, we would expect better performance on FOSTER-TEST than on RITTER-TEST and HOVY-TEST in POS tagging, as well as better performance on FININ-TEST than on the other two NER test sets. In POS tagging, we actually do see better results with FOSTER-TEST across the board, but in NER, FININ-TEST actually turns out to be the hardest data set.

4 Results

4.1 POS results

Baselines We use three supervised CRF models as baselines (cf. the first part of Table 2). The first supervised model is trained only on WSJ. This model does very well on FOSTER-DEV and FOSTER-TEST, presumably because of the low OOV rates (Figure 2). The second supervised model is trained only on GIMPEL-TRAIN; the third on the concatenation of WSJ and GIMPEL-TRAIN. While the second baseline performs well on held-out data from its own sample (90.3% on GIMPEL-DEV), it performs poorly across our out-of-sample test and development sets. Thus, it seems to overfit the sample of tweets described in Gimpel et al. (2011). The third model trained on the concatenation of WSJ and GIMPEL-TRAIN achieves the overall best baseline performance (88.4% macro-average accuracy). We note that this is around one percentage point better than the best available off-the-shelf system for Twitter (Owoputi et al., 2013) with an average accuracy of 87.5%.

[12] Actually (Ritter et al., 2011) do cross-validation over this data, but we use the splits of Derczynski et al. (2013) for POS.

[13] Using `https://github.com/miso-belica/jusText`

[14] `ftp://ftp.cs.cornell.edu/pub/smart/english.stop`

[15] `https://github.com/brendano/ark-tweet-nlp`

[16] `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/ems/support/split-sentences.perl`

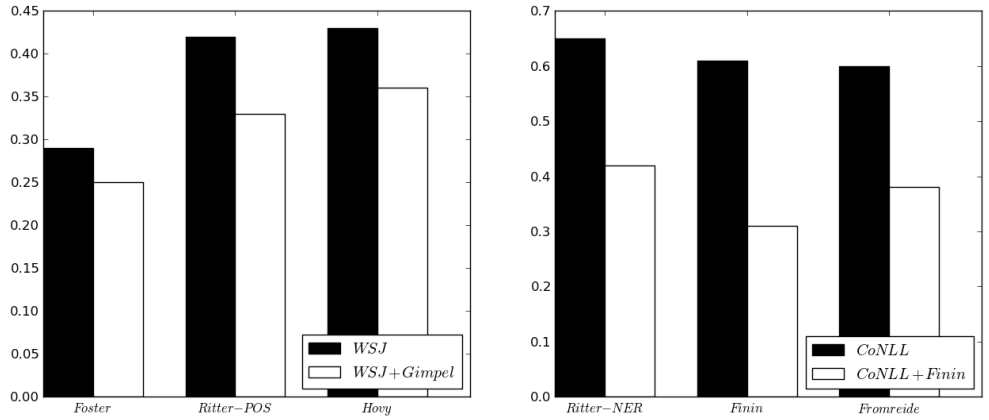


Figure 2: Test set (type-level) OOV rates for POS (left) and NER (right).

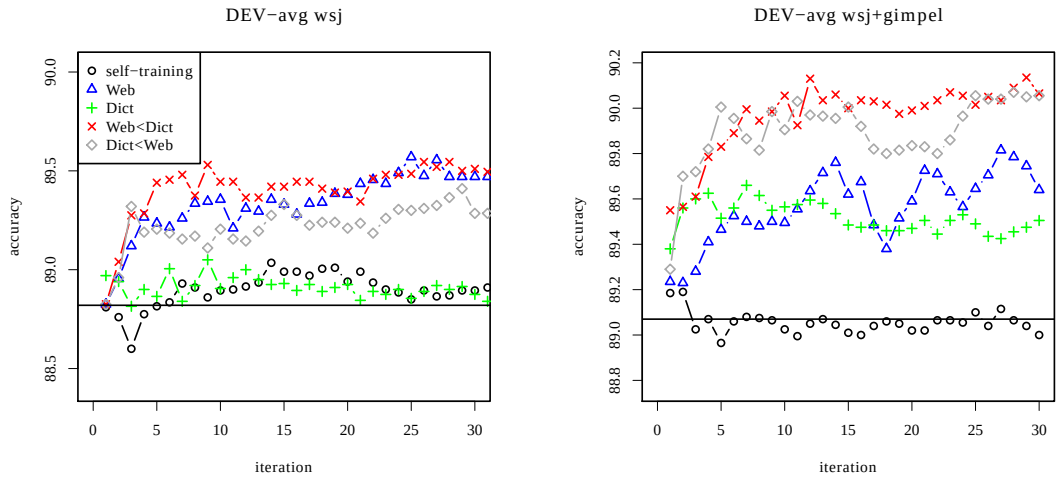


Figure 3: Learning curves on DEV-avg for systems trained on WSJ (left) and WSJ+GIMPEL (right) used to set the hyperparameter i.

Learning with URLs The results of our approaches are presented in Table 2. The hyperparameter i was set on the development data (cf. Figure 3). Note, again, that they do not require the test data to contain URLs. First of all, naive self-training does not work: accuracy declines or is just around baseline performance (Table 2 and Figure 3). In contrast, our augmented self-training methods with WEB or DICT reach large improvements. In case we assume no target training data (train on WSJ only, i.e. unsupervised DA), we obtain improvements of up to 9.1% error reduction. Overall the system improves from 88.42% to 89.07%. This also holds for the second scenario, i.e. training on WSJ+GIMPEL-TRAIN (semi-supervised DA, i.e., the case where we have some labeled target data, besides the pool of unlabeled tweets) where we reach error reductions of up to 10%. Our technique, in other words, improves the robustness of taggers, leading to much better performance on new samples of tweets.

4.2 NER results

For our NER results, cf. Table 3, we used the same feature models and parameter settings as those used for POS tagging, except conditioning also on POS information. It is conceivable that other parameter settings would have led to better results, but we did not want to assume the existence of in-domain development data for this task. Our baselines are again supervised systems, as well as off-the-shelf systems. Our in-

	DEV-avg	TEST			TEST-avg
		FOSTER	HOVY	RITTER	
Baselines trained on					
WSJ	88.82	91.87	87.01	86.38	88.42
GIMPEL-TRAIN	83.32	84.86	86.03	81.67	84.19
WSJ+GIMPEL-TRAIN	89.07	91.59	87.50	87.39	88.83
Systems trained on WSJ					
SELF-TRAINING $i = 25$	85.52	91.80	86.72	85.90	88.14
DICT $i = 25$	85.61	92.08	87.63	85.68	88.46
WEB $i = 25$	85.27	92.47	87.30	86.60	88.79
DICT$\prec$WEB $i = 25$	86.11	**92.61**	87.70	**86.69**	89.00
WEB$\prec$DICT $i = 25$	**86.15**	92.57	**88.12**	86.51	**89.07**
max err.red	4.7%	9.1%	8.6%	2.3%	4.2%
Systems trained on WSJ+GIMPEL-TRAIN					
SELF-TRAINING $i = 27$	89.12	91.83	86.88	87.43	88.71
DICT $i = 27$	89.43	92.22	**88.38**	87.69	89.43
WEB $i = 27$	89.82	**92.43**	87.43	88.21	89.36
DICT$\prec$WEB $i = 27$	**90.04**	**92.43**	**88.38**	**88.48**	**89.76**
WEB$\prec$DICT $i = 27$	**90.04**	92.40	87.99	88.39	89.59
max err.red	8.9%	10%	7.1%	8.6%	8.4%

Table 2: POS results.

house supervised baselines perform better than the available off-the-shelf systems, including the system provided by Ritter et al. (2011) (TEST-avg of 54.2%). We report micro-average F_1-scores over entity types, computed using the publicly available evaluation script.[17] Our approaches again lead to substantial error reductions of 8–13% across our NER evaluation data sets.

	TEST			TEST-avg
	RITTER	FROMREIDE	FININ	
Baseline trained on				
CONLL+FININ-TRAIN	77.44	82.13	74.02	77.86
Systems trained on CONLL+FININ-TRAIN				
SELF-TRAINING $i = 27$	**78.63**	82.88	74.89	78.80
DICT $i = 27$	65.24	69.1	65.45	66.60
WEB $i = 27$	78.29	83.82	**74.99**	79.03
DICT$\prec$WEB $i = 27$	78.53	**83.91**	75.83	**79.42**
WEB$\prec$DICT $i = 27$	65.97	69.92	65.86	67.25
err.red	9.1%	13.3%	8.0%	9.8%

Table 3: NER results.

5 Error analysis

The majority of cases where our taggers improve on the ARK tagger (Owoputi et al., 2013) seem to relate to richer linguistic context. The ARK tagger incorrectly tags the sequence *Man Utd* as PRT-NOUN, whereas our taggers correctly predict NOUN-NOUN. In a similar vein, our taggers correctly predict the tag sequence NOUN-NOUN for *Radio Edit*, while the ARK tagger predicts NOUN-VERB.

However, some differences seem arbitrary. For example, the ARK tagger tags the sequence *Nokia*

[17] http://www.cnts.ua.ac.be/conll2000/chunking/

D5000 in FOSTER-TEST as NOUN-NUM. Our systems correctly predict NOUN-NOUN, but it is not clear which analysis is better in linguistic terms. Our systems predict a sequence such as *Love his version* to be VERB-PRON-NOUN, whereas the ARK tagger predicts VERB-DET-NOUN. Both choices seem linguistically motivated.

Finally, some errors are made by all systems. For example, the word *please* in *please, do that*, for example, is tagged as VERB by all systems. In FOSTER-TEST, this is annotated as X (which in the PTB style was tagged as interjection UH). Obviously, *please* often acts as a verb, and while its part-of-speech in this case may be debatable, we see *please* annotated as a verb in similar contexts in the PTB, e.g.:

(3) Please/VERB make/VERB me/PRON ...

It is interesting to look at the tags that are projected from the websites to the tweets. Several of the observed projections support the intuition that coupling tweets and the websites they link to enables us to condition our tagging decisions on a richer linguistic context. Consider, for example *Salmon-Safe*, initially predicted to be a NOUN, but after projection correctly analyzed as an ADJ:

Word	Context	Initial tag	Projected tag
Salmon-Safe	... parks	NOUN	ADJ
Snohomish	... Bakery	ADJ	NOUN
toxic	ppl r ...	NOUN	ADJ

One of the most frequent projections is analyzing *you're*, correctly, as a VERB rather than an ADV (if the string is not split by tokenization).

One obvious limitation of the WEB-based models is that the projections apply to all occurrences of a word. In rare cases, some words occur with different parts of speech in a single tweet, e.g., *wish* in:

(4) If I gave you one **wish** that will become true . What's your **wish** ?... ? i **wish** i'll get <num> wishes from you :p <url>

In this case, our models enforce all occurrences of *wish* to, incorrectly, be verbs.

6 Related work

Previous work on tagging tweets has assumed labeled training data (Ritter et al., 2011; Gimpel et al., 2011; Owoputi et al., 2013; Derczynski et al., 2013). Strictly supervised approaches to analyzing Twitter has the weakness that labeled data quickly becomes unrepresentative of what people write on Twitter. This paper presents results using no in-domain labeled data that are significantly better than several off-the-shelf systems, as well as results leveraging a mixture of out-of-domain and in-domain labeled data to reach new highs across several data sets.

Type-constrained POS tagging using tag dictionaries has been explored in weakly supervised settings (Li et al., 2012), as well as for cross-language learning (Das and Petrov, 2011; Täckström et al., 2013). Our type constraints in POS tagging come from tag dictionaries, but also from linked websites. The idea of using linked websites as distant supervision is similar in spirit to the idea presented in Ganchev et al. (2012) for search query tagging.

Ganchev et al. (2012), considering the problem of POS tagging search queries, tag search queries and the associated snippets provided by the search engine, projecting tags from the snippets to the queries, guided by click-through data. They do not incorporate tag dictionaries, but consider a slightly more advanced matching of snippets and search queries, giving priority to n-gram matches with larger n. Search queries contain limited contexts, like tweets, but are generally much shorter and exhibit less spelling variation than tweets.

In NER, it is common to use gazetteers, but also dictionaries as distant supervision (Kazama and Torisawa, 2007; Cucerzan, 2007). Rüd et al. (2011) consider using search engines for distant supervision of NER of search queries. Their set-up is very similar to Ganchev et al. (2012), except they do not use click-through data. They use the search engine snippets to generate feature representations rather than projections. Want et al. (2013) also use distant supervision for NER, i.e., Wikipedia page view counts,

applying their model to Twitter data, but their results are considerably below the state of the art. Also, their source of supervision is not linked to the individual tweets in the way mentioned websites are.

In sum, our method is the first successful application of distant supervision to POS tagging and NER for Twitter. Moreover, it is, to the best of our knowledge, the first paper that addresses both problems using the same technique. Finally, our results are significantly better than state-of-the-art results in both POS tagging and NER.

7 Conclusion

We presented a semi-supervised approach to POS tagging and NER for Twitter data that uses dictionaries and linked websites as a source of not-so-distant (or linked) supervision to guide the bootstrapping. Our approach outperforms off-the-shelf taggers when evaluated across various data sets, achieving average error reductions across data sets of 5% on POS tagging and 10% on NER over state-of-the-art baselines.

References

Timothy Baldwin, Paul Cook, Marco Lui, Andrew MacKinlay, and Li Wang. 2013. How noisy social media text, how diffrnt social media sources? In *IJCNLP*.

Edward Benson, Aria Haghighi, and Regina Barzilay. 2011. Event discovery in social media feeds. In *ACL*.

Silvia Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *ACL*.

Hal Daumé, Abhishek Kumar, and Avishek Saha. 2010. Frustratingly easy semi-supervised domain adaptation. In *ACL Workshop on Domain Adaptation for NLP*.

Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. 2013. Twitter part-of-speech tagging for all: overcoming sparse and noisy data. In *RANLP*.

Jacob Eisenstein. 2013. What to do about bad language on the internet. In *NAACL*.

Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in Twitter data with crowdsourcing. In *NAACL-HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*.

Jenny Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *ACL*.

Jennifer Foster, Ozlem Cetinoglu, Joachim Wagner, Josef Le Roux, Joakim Nivre, Deirde Hogan, and Josef van Genabith. 2011. From news to comments: Resources and benchmarks for parsing the language of Web 2.0. In *IJCNLP*.

Hege Fromreide, Dirk Hovy, and Anders Søgaard. 2014. Crowdsourcing and annotating ner for twitter #drift. In *LREC*.

Kuzman Ganchev, Keith Hall, Ryan McDonald, and Slav Petrov. 2012. Using search-logs to improve query tagging. In *ACL*.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *ACL*.

David Hand. 2006. Classifier technology and illusion of progress. *Statistical Science*, 21(1):1–15.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *NAACL*.

Dirk Hovy, Barbara Plank, and Anders Søgaard. 2014. When pos datasets don't add up: Combatting sample bias. In *LREC*.

Zhongqiang Huang, Mary Harper, and Slav Petrov. 2009. Self-training with products of latent variable grammars. In *EMNLP*.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *ACL*.

Long Jiang, Mo Yo, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent Twitter sentiment classification. In *ACL*.

Jun'ichi Kazama and Kentaro Torisawa. 2007. Exploiting wikipedia as external knowledge for named entity recognition. In *EMNLP-CoNLL*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *ICML*.

Shen Li, João Graça, and Ben Taskar. 2012. Wiki-ly supervised part-of-speech tagging. In *EMNLP*.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *HLT-NAACL*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *NAACL*.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *LREC*.

Sasa Petrovic, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to Twitter. In *NAACL*.

Barbara Plank. 2011. *Domain Adaptation for Parsing*. Ph.D. thesis, University of Groningen.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *EMNLP*.

Stefan Rüd, Massimiliano Ciaramita, Jens Müller, and Hinrich Schütze. 2011. Piggyback: Using search engines for robust cross-domain named entity recognition. In *ACL*.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *TACL*, 1:1–12.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *In CoNLL*.

Yoshimasa Tsuruoka, Yusuke Miyao, and Jun'ichi Kazama. 2011. Learning with lookahead: can history-based models rival globally optimized models? In *CoNLL*.

Chun-Kai Wang, Bo-June Hsu, Ming-Wei Chang, and Emre Kiciman. 2013. Simple and knowledge-intensive generative model for named entity recognition. Technical report, Microsoft Research.

Interpolated Dirichlet Class Language Model for Speech Recognition Incorporating Long-distance N-grams

Md. Akmal Haidar and Douglas O'Shaughnessy
INRS-EMT, University of Quebec
6900-800 De la Gauchetier Ouest, H5A 1K6, Montreal (Quebec), Canada
haidar@emt.inrs.ca, dougo@emt.inrs.ca

Abstract

We propose a language modeling (LM) approach incorporating interpolated distanced n-grams in a Dirichlet class language model (DCLM) (Chien and Chueh, 2011) for speech recognition. The DCLM relaxes the bag-of-words assumption and documents topic extraction of latent Dirichlet allocation (LDA). The latent variable of DCLM reflects the class information of an n-gram event rather than the topic in LDA. The DCLM model uses default background n-grams where class information is extracted from the $(n$-$1)$ history words through Dirichlet distribution in calculating n-gram probabilities. The model does not capture the long-range information from outside of the n-gram window that can improve the language modeling performance. In this paper, we present an interpolated DCLM (IDCLM) by using different distanced n-grams. Here, the class information is exploited from $(n$-$1)$ history words through the Dirichlet distribution using interpolated distanced n-grams. A variational Bayesian procedure is introduced to estimate the IDCLM parameters. We carried out experiments on a continuous speech recognition (CSR) task using the Wall Street Journal (WSJ) corpus. The proposed approach shows significant perplexity and word error rate (WER) reductions over the other approach.

1 Introduction

Statistical n-gram LMs have been successfully used for speech recognition and many other applications. They suffer from insufficiencies of training data and long-distance information, which limit the model generalization (Chien, 2006). The data sparseness problem is usually solved by backoff smoothing using lower-order language models (Katz, 1987; Kneser and Ney, 1995). The class-based language model was investigated where the class n-grams were calculated by considering the generation of concatenated classes rather than words (Brown et al., 1992). By incorporating the multidimensional word classes and considering the classes from various positions of left and right contextual information (Bai et al., 1998), the class n-gram can be improved (Yamamoto et al., 2003). A neural network language model (NNLM) was trained by linearly projecting the history words of an n-gram event into a continuous space (Bengio et al., 2003; Schwenk, 2007). Later, a recurrent neural network-based LM was investigated that shows better results than NNLM (Mikolov et al., 2010; Mikolov et al., 2011). Unsupervised class-based language models such as Random Forest LM (Xu and Jelinek, 2007), Model M (Chen, 2008) have been investigated that outperform a word-based LM. However, the long-distance information is captured by using a cache-based LM that takes advantage of the fact that a word observed earlier in a document could occur again. This helps to increase the probability of the seen words when predicting the next word (Kuhn and Mori, 1990).

To compensate for the weakness of the n-gram models, latent topic analysis has been used broadly. Several techniques such as Latent Semantic Analysis (LSA) (Deerwester et al., 1990; Bellegarda, 2000), probabilistic LSA (PLSA) (Hofmann, 1999; Gildea and Hofmann, 1999), and Latent Dirichlet Allocation (LDA) (Blei et al., 2003) have been studied to extract the latent semantic information from a training

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1793–1802, Dublin, Ireland, August 23-29 2014.

corpus. The LSA, PLSA and LDA models have been used successfully in recent research work for LM adaptation (Bellegarda, 2000; Gildea and Hofmann, 1999; Mrva and Woodland, 2004; Tam and Schultz, 2005; Tam and Schultz, 2006; Haidar and O'Shaughnessy, 2011; Haidar and O'Shaughnessy, 2012b; Haidar and O'Shaughnessy, 2012a). Even so, the extracted topic information is not directly useful for speech recognition, where the latent topic of n-gram events should be of concern. In (chien and Chueh, 2008), a latent Dirichlet language model (LDLM) was proposed where the latent topic information was exploited from (n-1) history words through the Dirichlet distribution in calculating the n-gram probabilities. A topic cache language model was proposed where the topic information was obtained from long-distance history through multinomial distributions (Chueh and Chien, 2010). Topic-dependent-class-based n-gram LM was proposed where the LSA method was used to reveal latent topic information from noun-noun relations (Naptali et al., 2012). In (Bassiou and Kotropoulos, 2010), a PLSA technique enhanced with long-distance bigrams was used to incorporate the long-term word dependencies in determining word clusters. This technique was used in (Haidar and O'Shaughnessy, 2013b) and (Haidar and O'Shaughnessy, 2013a) for the PLSA and LDLM models respectively where the long-distance information was captured by using interpolated distanced n-grams and their parameters were estimated by using an expectation maximization (EM) procedure (Dempster et al., 1977). In (Chien and Chueh, 2011), the DCLM model was proposed to tackle the data sparseness and to extract the large-span information for the n-gram model. In this model, the topic structure in LDA is assumed to derive the hidden classes of histories in calculating the language model. A Bayesian class-based language model was presented where a variational Bayes-EM procedure was used to compute the model parameters. Also, a cache DCLM model was proposed to capture the long-distance information beyond the n-gram window. However, in the DCLM model (Chien and Chueh, 2011), the class information of the history words was obtained from the n-gram events of the corpus. Here, the long-range information outside the n-gram window is not captured. In this paper, we present an IDCLM model to capture the long-range information in the DCLM using the interpolated distanced n-grams. The n-gram probabilities of the proposed IDCLM model are computed by mixing the component distanced word probabilities for classes and the interpolated class information for histories. Similar to the DCLM model, the parameters of the IDCLM model are computed by using the variational Bayesian-EM procedure.

The rest of this paper is organized as follows. Section 2 is used for reviewing the DCLM model. The proposed IDCLM model is described in section 3. The comparison of the IDCLM and the DCLM models is described in section 4. The experimental details are described in section 5. Finally, the conclusions and future work are described in section 6.

2 DCLM

LDA is used to compute the document probability by using the topic structure at the document level, which is inconsistent with the language model for speech recognition where the n-gram regularities are characterized (Chien and Chueh, 2011). The DCLM was developed to model the n-gram events of the corpus for speech recognition. In the DCLM, the class structure is described by Dirichlet densities and estimated from n-gram events. The graphical model of the DCLM for a text corpus that comprises n-gram events $\{w_{i-n+1}^{i-1}, w_i\}$ is described in Figure 1. Here, H and N_h represent the number of history events w_{i-n+1}^{i-1} and the number of collected words that occur following the history w_{i-n+1}^{i-1}, respectively. The (n-1) history words w_{i-n+1}^{i-1} are represented by a (n-1)V $\times$ 1 vector $\mathbf{h}$, consisting of n-1 block subvectors, with the entries of the seen words assigned to ones and those of unseen words assigned to zeros (Chien and Chueh, 2011). Here, V represents the size of the vocabulary. The vector $\mathbf{h}$ is then projected into a C-dimensional continuous class space using a class-dependent linear discriminant function:

$$g_c(\mathbf{h}) = \mathbf{a}_c^T \mathbf{h} \tag{1}$$

where $\mathbf{a}_c^T$ is the c^{th} row vector of matrix $\mathbf{A} = [\mathbf{a}_1, \cdots, \mathbf{a}_C]$ (Chien and Chueh, 2011). The function $g_c(\mathbf{h})$ describes the class posterior probability $p(c|\mathbf{h})$, which is used in predicting the class information for an unseen history (Chien and Chueh, 2011). The model can be described as:

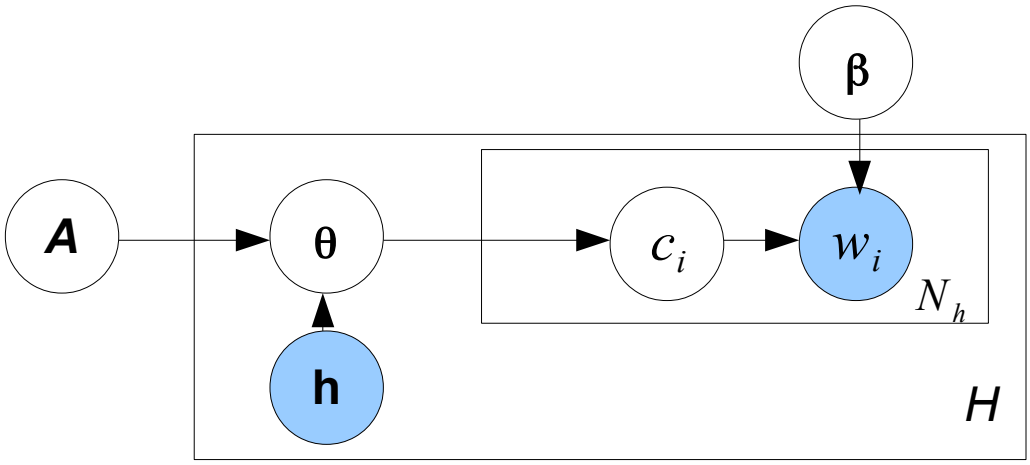


Figure 1: The graphical model of the DCLM. Shaded circles represent observed variables.

- For each history vector $\mathbf{h}$, the class information c is drawn from a history-dependent Dirichlet prior $\boldsymbol{\theta}$, which is related to a global projection matrix $\mathbf{A}$:

$$p(\boldsymbol{\theta}|\mathbf{h},\mathbf{A}) \propto \prod_{c=1}^{C} \theta_c^{g_c(\mathbf{h})-1}, \tag{2}$$

- For each predicted word w_i of the n-gram events from a multinomial distribution with parameter $\boldsymbol{\beta}$, the associated class c_i is chosen by using a multinomial distribution with parameter $\boldsymbol{\theta}$. The joint probability of the variable $\boldsymbol{\theta}$, c_i, and w_i conditioned on $\mathbf{h}$ can be computed as:

$$p(\boldsymbol{\theta}, c_i, w_i|\mathbf{h},\mathbf{A},\boldsymbol{\beta}) = p(\boldsymbol{\theta}|\mathbf{h},\mathbf{A})p(c_i|\boldsymbol{\theta})p(w_i|c_i,\boldsymbol{\beta}) \tag{3}$$

- The conditional probability in the n-gram language model can thus be obtained as:

$$p(w_i|\mathbf{h},\mathbf{A},\boldsymbol{\beta}) = \int p(\boldsymbol{\theta}|\mathbf{h},\mathbf{A}) \sum_{c_i=1}^{C} p(c_i|\boldsymbol{\theta})p(w_i|c_i,\boldsymbol{\beta})d\boldsymbol{\theta}, \tag{4}$$

  where the integral is computed as:

$$p(c_i|\mathbf{h},\mathbf{A}) = \int p(\boldsymbol{\theta}|\mathbf{h},\mathbf{A})p(c_i|\boldsymbol{\theta})d\boldsymbol{\theta} = \frac{g_{c_i}(\mathbf{h})}{\sum_{j=1}^{C} g_j(\mathbf{h})}. \tag{5}$$

  which is an expectation of a Dirichlet distribution of latent class c_i (Chien and Chueh, 2011).

Therefore, the probability of an n-gram event using the DCLM (Equation 4 and 5) can be written as (Chien and Chueh, 2011):

$$p(w_i|\mathbf{h},\mathbf{A},\boldsymbol{\beta}) = \sum_{c=1}^{C} p(w_i|c,\boldsymbol{\beta}) \frac{g_c(\mathbf{h})}{\sum_{j=1}^{C} g_j(\mathbf{h})} \tag{6}$$

The parameters ($\mathbf{A}$,$\boldsymbol{\beta}$) of the model are computed by using the variational bayesian EM (VB-EM) procedure (Chien and Chueh, 2011).

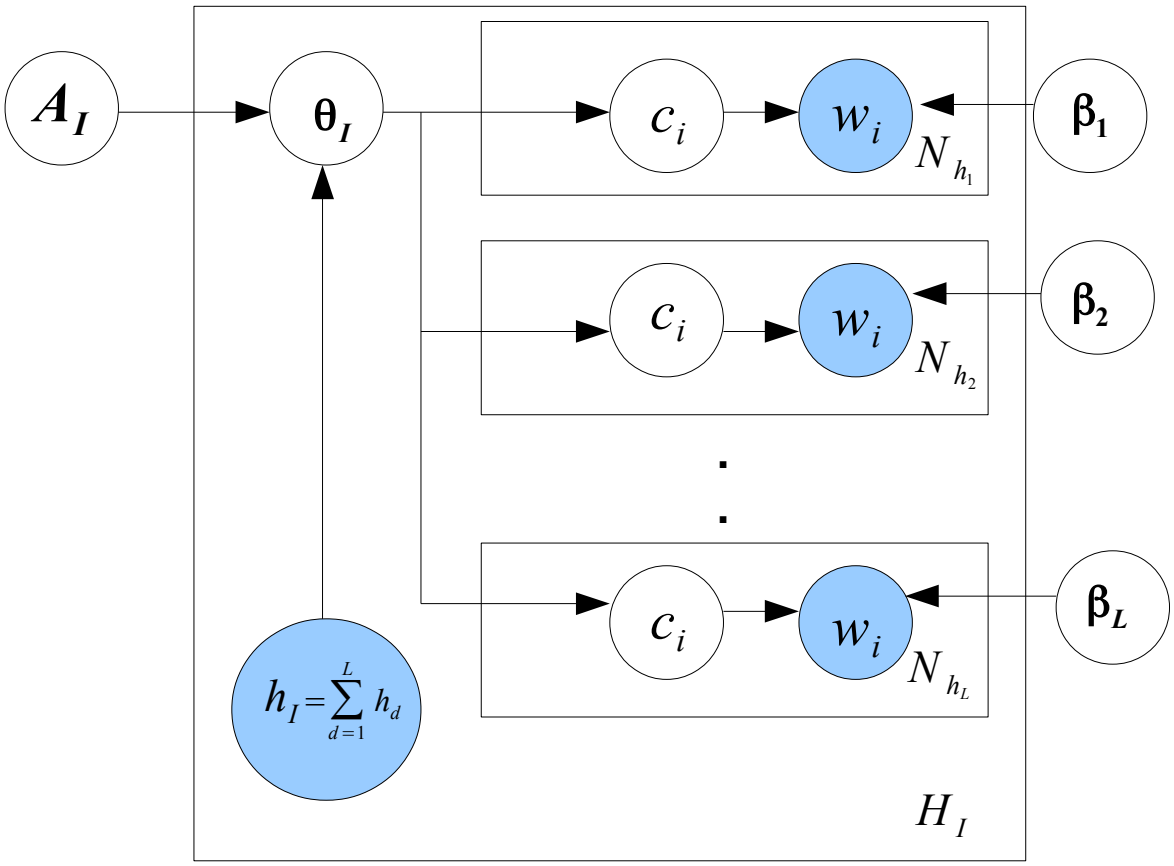


Figure 2: The graphical model of the IDCLM. Shaded circles represent observed variables.

3 Proposed IDCLM

The DCLM does not capture the long-range information from outside of the n-gram window (Chien and Chueh, 2011). To incorporate the long-range information into the DCLM, we propose an IDCLM where the class information is extracted from interpolated distance n-gram histories through a Dirichlet distribution in calculating the language model probability. In this model, we interpolate the distanced n-gram events into the original n-gram events of the DCLM. The graphical model of the IDCLM is described in Figure 2. In Figure 2, H_I contains the histories of all the distanced d n-grams, d represents the distance between words in the n-gram events, and L describes the maximum length of distance d. When d = 1, the n-grams are the default background n-grams. For example, the distanced tri-grams of the phrase "Interpolated Dirichlet Class Language Model for Speech Recognition" are described in Table 1 for the distance $d = 1, 2, 3$. Here, the (n-1)V dimensional discrete history vector $\mathbf{h}_I$ is projected

d	Trigrams
1	*Interpolated Dirichlet Class, Dirichlet Class Language, Class Language Model, Language Model for, Model for Speech, for Speech Recognition*
2	*Interpolated Class Model, Dirichlet Language for, Class Model Speech, Language for Recognition*
3	*Interpolated Language Speech, Dirichlet Model Recognition*

Table 1: *Distanced tri-grams for the phrase "Interpolated Dirichlet Class Language Model for Speech Recognition"*

into a C-dimensional continuous class space using a class-dependent linear discriminant function:

$$g_c(\mathbf{h}_I) = \mathbf{a}_{c,I}^T \mathbf{h}_I \tag{7}$$

where $\mathbf{h}_I$ is the combined histories of all the distanced histories $\mathbf{h}_d$ and is defined as $\mathbf{h}_I = \sum_{d=1}^{L} \mathbf{h}_d$. Here, $\sum$ represents the logical OR operator. $\mathbf{a}_{c,I}^T$ is the c^{th} row vector of the matrix $\mathbf{A}_I$ and $g_c(\mathbf{h}_I)$ describes the class posterior probability $p(c|\mathbf{h}_I)$.

The n-gram probability of the IDCLM model is computed as:

$$\begin{aligned} p_I(w_i|\mathbf{h}_I, \mathbf{A}_I, \boldsymbol{\beta}_d) &= \sum_{c_i=1}^{C} \left\{ \Big[\sum_d \lambda_d p_d(w_i|c_i, \boldsymbol{\beta}_d) \Big] \times \int p(\boldsymbol{\theta}_I|\mathbf{h}_I, \mathbf{A}_I) p(c_i|\boldsymbol{\theta}_I) d\boldsymbol{\theta}_I \right\} \\ &= \sum_{c=1}^{C} \Big[\sum_d \lambda_d \beta_{d,ic} \Big] \frac{g_c(\mathbf{h}_I)}{\sum_{j=1}^{C} g_j(\mathbf{h}_I)} \end{aligned} \tag{8}$$

where λ_d are the weights for each component probability estimated on the held-out data using the EM algorithm (Bassiou and Kotropoulos, 2010; Dempster et al., 1977).

The parameters of the IDCLM model are computed using the variational Bayes EM (VB-EM) procedure by maximizing the marginal distribution of the training data that contains a set of n-gram events $D = \{w_{i-n+1}^{i-1}, w_i\}$:

$$\begin{aligned} \log p(D|\mathbf{A}_I, \boldsymbol{\beta}_d) &= \sum_{(w_i, \mathbf{h}_I) \in D} \log p_I(w_i|\mathbf{h}_I, \mathbf{A}_I, \boldsymbol{\beta}_d) \\ &= \sum_{\mathbf{h}_I} \log \left\{ \int p(\boldsymbol{\theta}_I|\mathbf{h}_I, \mathbf{A}_I) \times \left[\sum_d \prod_{j=1}^{N_{h_d}} \sum_{c_j=1}^{C} \lambda_d p_d(w_j|c_j, \boldsymbol{\beta}_d) p(c_j|\boldsymbol{\theta}_I) \right] d\boldsymbol{\theta}_I \right\} \end{aligned} \tag{9}$$

where D contains all the distanced n-gram events, N_{h_d} represents the number of collected words that occur following the history $\mathbf{h}_d$ in d-distanced n-grams. In Equation 9, the summation is over all possible histories in training samples D. However, directly optimizing the Equation 9 is intractable (Chien and Chueh, 2011). A variational IDCLM is introduced where the marginal likelihood is approximated by maximizing the lower bound of Equation 9. The VB-EM procedure is required since the parameter estimation involves the latent variables of $\{\boldsymbol{\theta}_I, \mathbf{c}_{h_d} = \{c_i\}_{i=1}^{N_{h_d}}\}$.

The lower bound $L(\mathbf{A}_I, \boldsymbol{\beta}_d; \hat{\boldsymbol{\gamma}}_I, \hat{\boldsymbol{\phi}}_d)$ is given by:

$$\begin{aligned} &\sum_{\mathbf{h}_I} \left\{ \log \Gamma\Big(\sum_{c=1}^{C} g_c(\mathbf{h}_I) \Big) - \sum_{c=1}^{C} \log \Gamma(g_c(\mathbf{h}_I)) + \sum_{c=1}^{C} (g_c(\mathbf{h}_I) - 1) \times \Big(\Psi(\gamma_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C} \gamma_{h_I,j} \Big) \Big) \right\} \\ &\qquad + \sum_d \sum_{\mathbf{h}_d} \sum_{i=1}^{N_{h_d}} \sum_{c=1}^{C} \lambda_d \phi_{h_d,ic} \Big(\Psi(\gamma_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C} \gamma_{h_I,j} \Big) \Big) \\ &+ \sum_d \sum_{\mathbf{h}_d} \sum_{i=1}^{N_{h_d}} \sum_{c=1}^{C} \sum_{v=1}^{V} \lambda_d \phi_{h_d,ic} \delta(w_v, w_i) \log \beta_{d,vc} - \sum_{\mathbf{h}_I} \left\{ \log \Gamma\Big(\sum_{c=1}^{C} \gamma_{h_I,c} \Big) - \sum_{c=1}^{C} \log \Gamma(\gamma_{h_I,c}) \right. \\ &\qquad \left. + \sum_{c=1}^{C} (\gamma_{h_I,c} - 1) \Big(\Psi(\gamma_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C} \gamma_{h_I,j} \Big) \Big) \right\} - \sum_d \sum_{\mathbf{h}_d} \sum_{i=1}^{N_{h_d}} \sum_{c=1}^{C} \lambda_d \phi_{h_d,ic} \log \phi_{h_d,ic} \end{aligned}$$

where $\Psi(.)$ is the derivative of the log gamma function, and is known as a digamma function (Chien and Chueh, 2011). The history-dependent variational parameters $\{\hat{\boldsymbol{\gamma}}_{h_I} = \hat{\gamma}_{h_I,c}, \hat{\boldsymbol{\phi}}_{h_d} = \hat{\phi}_{h_d,vc}\}$, corresponding to the latent variables $\boldsymbol{\theta}_I, \mathbf{c}_{h,d}$, are then estimated in the VB-E step by setting the differentials $(\partial L(\boldsymbol{\gamma}))/(\partial \gamma_{h_I,c})$ and $(\partial L(\boldsymbol{\phi}))/(\partial \phi_{h_d,ic})$ to zero respectively (Chien and Chueh, 2011):

$$\hat{\gamma}_{h_I,c} = g_c(\mathbf{h}_I) + \sum_d \sum_{i=1}^{N_{h_d}} \lambda_d \phi_{h_d,ic} \tag{10}$$

$$\hat{\phi}_{h_d,ic} = \frac{\beta_{d,ic} \exp\left[\Psi(\gamma_{h_I,c}) - \Psi(\sum_{j=1}^{C} \gamma_{h_I,j})\right]}{\sum_{l=1}^{C} \beta_{d,il} \exp\left[\Psi(\gamma_{h_I,l}) - \Psi(\sum_{j=1}^{C} \gamma_{h_I,j})\right]} \tag{11}$$

In computing $\hat{\phi}_{h_d,ic}$ the corresponding $\gamma_{h_d,c}$ is used in Equation 11. With the updated $\hat{\boldsymbol{\gamma}}_{h_I}, \hat{\boldsymbol{\phi}}_{h_d}$ in the VB-E step, the IDCLM parameters $\{\mathbf{A}_I,\boldsymbol{\beta}_d\}$ are estimated in the VB-M step as (Chien and Chueh, 2011):

$$\hat{\beta}_{d,vc} = \frac{\sum_{\mathbf{h}_d} \sum_{i=1}^{N_{h_d}} \lambda_d \hat{\phi}_{h_d,ic} \delta(w_v, w_i)}{\sum_{m=1}^{V} \sum_{\mathbf{h}_d} \sum_{i=1}^{N_{h_d}} \lambda_d \hat{\phi}_{h_d,ic} \delta(w_m, w_i)} \tag{12}$$

where $\sum_{v=1}^{V} \beta_{d,vc}$=1 and $\delta(w_v, w_i)$ is the Kronecker delta function that equals one when vocabulary word w_v is identical to the predicted word w_i and equals zero otherwise. The gradient ascent algorithm is used to calculate the parameters $\hat{\mathbf{A}}_I = [\hat{\mathbf{a}}_{1,I}, \cdots, \hat{\mathbf{a}}_{C,I}]$ by updating the gradient $\bigtriangledown_{\mathbf{a}_{c,I}}$as (Chien and Chueh, 2011):

$$\bigtriangledown_{\mathbf{a}_{c,I}} \leftarrow \bigtriangledown_{\mathbf{a}_{c,I}} + \sum_{\mathbf{h}_I} \left[\Psi\Big(\sum_{j=1}^{C} g_j(\mathbf{h}_I) \Big) - \Psi(g_c(\mathbf{h}_I)) + \Psi(\hat{\gamma}_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C} \hat{\gamma}_{h_I,j} \Big) \right] . \mathbf{h}_I \tag{13}$$

The n-gram probabilities $p_t(w_i, \mathbf{h}_t, \mathbf{A}_I, \boldsymbol{\beta}_d)$ of the test document t are then computed using Equation 8. To capture the local lexical regularities, the model $p_t(w_i|\mathbf{h}_t, \mathbf{A}_I, \boldsymbol{\beta}_d)$ is then interpolated with the background trigram model as:

$$p_{Interpolated}(w_i|\mathbf{h}) = \mu p_{Background}(w_i|\mathbf{h}) + (1-\mu) p_t(w_i|\mathbf{h}_t, \mathbf{A}_I, \boldsymbol{\beta}_d) \tag{14}$$

4 Comparison of DCLM and IDCLM Models

In the DCLM model, the class information for the $(n-1)$ history words is obtained by using the n-gram counts in the corpus. The current word is predicted from the history-dependent Dirichlet parameter, which is controlled by a matrix $\mathbf{A}$ and corpus-based histories $\mathbf{h}$ (Chien and Chueh, 2011). In contrast, the IDCLM model captures long-range information by incorporating distanced n-grams. Here, the class information is exploited for the interpolated $(n-1)$ history words $\mathbf{h_I}$ that are obtained from all the distanced n-gram events. Both the DCLM and IDCLM exploit the word distribution given the history words. They perform the history clustering of the corpus. For the DCLM model, the number of parameters $\{\mathbf{A},\boldsymbol{\beta}\}$ increases linearly with the number of history words and is given by $(n-1)CV + CV$. For the IDCLM model, the number of parameters $\{\mathbf{A}_I,\boldsymbol{\beta}_d\}$ increases linearly with the number of history words and distance d and is given by $((n-1)CV + CVd)$. The time complexity of DCLM and IDCLM are $O(HVC)$ and $O(H_IVCd)$ with H corpus-based histories, H_I corpus-based interpolated histories, V vocabulary words, d distances and C classes.

5 Experiments

5.1 Data and experimental setup

The LM approaches are evaluated using the Wall Street Journal (WSJ) corpus (Paul and Baker, 1992). The SRILM toolkit (Stolcke, 2002) and the HTK toolkit (Young et al., 2013) are used for generating the LMs and computing the WER respectively. The '87-89 WSJ corpus is used to train language models. The background trigrams are trained using the back-off version of the Witten-Bell smoothing; the 5K non-verbalized punctuation closed vocabulary. We train the trigram IDCLM model using $L = 2$ and $L = 3$. Ten EM iterations in the VB-EM procedure were used. The initial values of the entries in the matrix $\boldsymbol{\beta}$,$\boldsymbol{\beta}_d$ were set to be $1/V$ and those in $\mathbf{A}, \mathbf{A}_I$ were randomly set in the range [0,1]. To update the variational parameters in the VB-E step, one iteration was used. The VB-M step was executed to update the parameters $\mathbf{A}, \mathbf{A}_I$ by three iterations (Chien and Chueh, 2011). To capture the local lexical regularity, trigrams of various methods are interpolated with the background trigrams. The acoustic model from (Vertanen, 2013) is used in our experiments. The acoustic model is trained by using all WSJ and TIMIT (Garofolo et al., 1993) training data, the 40-phone set of the CMU dictionary (-, 2013),

approximately 10000 tied-states, 32 Gaussians per state and 64 Gaussians per silence state. The acoustic waveforms are parameterized into a 39-dimensional feature vector consisting of 12 cepstral coefficients plus the 0^{th} cepstral, delta and delta delta coefficients, normalized using cepstral mean subtraction $(MFCC_{0-D-A-Z})$. We evaluated the cross-word models. The values of the word insertion penalty, beam width, and the language model scale factor are -4.0, 350.0, and 15.0 respectively (Vertanen, 2013). The interpolation weights λ_d and μ are computed by optimizing on the held-out data according to the metric of perplexity. The experiments are evaluated on the evaluation test, which is a total of 330 test utterances from the November 1992 ARPA CSR benchmark test data for vocabularies of 5K words (Paul and Baker, 1992; Woodland et al., 1994).

5.2 Experimental Results

Due to the higher memory and training time requirements for the IDCLM model, we trained the DCLM and IDCLM models for class sizes of 10 and 20. The perplexity and WER results are described in Table 2 and Figure 3 respectively.

Language Model	10 Classes	20 Classes
Background (B)	109.41	109.41
B+Class	106.65	106.97
B+DCLM	100.20	100.45
B+IDCLM (L=2)	98.01	97.94
B+IDCLM (L=3)	95.63	95.43

Table 2: Perplexity results of the models

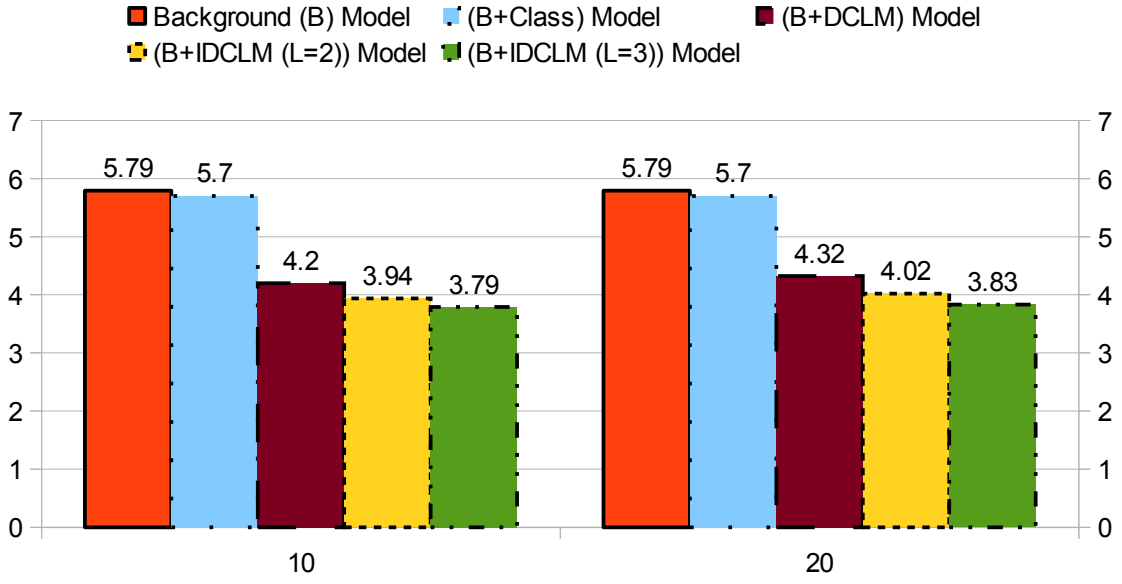


Figure 3: WER results for different class sizes

From Table 2, we can note the proposed IDCLM model outperforms the other models for all class sizes. The performance of IDCLM improves with more distances ($L = 3$).

We evaluated the WER experiments using lattice rescoring. In the first pass decoding, we used the background trigram for lattice generation. In the second pass, we applied the interpolated model for lattice rescoring. The WER results are described in Figure 3. From Figure 3, we can note that the proposed IDCLM ($L = 3$) model yields a WER reduction of about 34.54% (5.79% to 3.79%), 33.5% (5.7% to 3.79%), and 9.76% (4.2% to 3.79%) for 10 classes and about 33.85% (5.79% to 3.83%), 32.8%

(5.7% to 3.83%), and 11.34% (4.32% to 3.83%) over the background trigram, class trigram (Brown et al., 1992), and the DCLM (Chien and Chueh, 2011) approaches respectively. The significance improvement in WER is done by using a match-pair-test where the misrecognized words in each test utterance are counted. The p-values are described in Table 3. From Table 3, we can note that the IDCLM ($L = 2$)

Language Model	10 Classes	20 Classes
B+Class & B+IDCLM (L=2)	3.8E-10	4.3E-10
B+Class & B+IDCLM (L=3)	4.7E-12	4.7E-12
B+DCLM & B+IDCLM (L=2)	0.04	0.01
B+DCLM & B+IDCLM (L=3)	0.004	0.006

Table 3: p-values obtained from the match-pair test on the WER results

is statistically significant to the class-based LM (Brown et al., 1992) and DCLM (Chien and Chueh, 2011) at a significance level of 0.01 and 0.05 respectively. However, the IDCLM ($L = 3$) model is statistically significant to the above models at a significance level of 0.01. We have also seen that the cache DCLM model also gives the same results as DCLM (Chien and Chueh, 2011) for smaller number of classes (Chien and Chueh, 2011).

6 Conclusions and Future Work

In this paper, we proposed an integration of distanced n-grams into the original DCLM model (Chien and Chueh, 2011). The DCLM model (Chien and Chueh, 2011) extracted the class information from the $(n$-1) history words through a Dirichlet distribution in calculating the n-gram probabilities. However, it does not capture the long-range semantic information from outside of the n-gram events. The proposed IDCLM overcomes the shortcomings of DCLM by incorporating the interpolated long-distance n-grams that capture the long-term word dependencies. Using the IDCLM, the class information for the histories is trained using the interpolated distanced n-grams. The IDCLM yields better results with including more distances ($L = 3$). The model probabilities are computed by weighting the component word probabilities for classes and the interpolated class information for histories. A variational Bayesian EM (VB-EM) procedure is presented to estimate the model parameters.

For future work, we will evaluate the proposed approach with neural network-based language models and exponential class-based language models. Furthermore, we will find out a way to perform the experiments for higher numbers of classes.

References

-. 2013. The Carnegie Mellon University (CMU) Pronounciation Dictionary. `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`.

A.P. Dempster, N.M. Laird, and D.B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, Series B 39(1):1 – 38.

Andreas Stolcke. 2002. SRILM-an Extensible Language Modeling Toolkit. In *Proceedings of ICSLP*, pages 901–904.

Chuang-H. Chueh and Jen-T. Chien. 2010. Topic Cache Language Model for Speech Recognition. In *Proc. of ICASSP*, pages 5194–5197.

Daniel Gildea and Thomas Hofmann. 1999. Topic-based Language Models using EM. In *Proceedings of EUROSPEECH*, pages 2167–2170.

David Mrva and Philip C. Woodland. 2004. A PLSA-based Language Model for Conversational Telephone Speech. In *Proc. of ICSLP*, pages 2257–2260.

David M. Blei, Andrew Y. Ng., and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Dougls B. Paul and Janet M. Baker. 1992. The Design for the Wall Street Journal-based CSR Corpus. In *Proc. of ICSLP*, pages 899–902.

Hirofumi Yamamoto, Shuntaro Isogai, and Yoshinori Sagisaka. 2003. Multi-class Composite n-gram Language Model. *Speech Communication*, 41:369 – 379.

Holger Schwenk. 2007. Continuous Space Language Models. *Computer Speech and Language*, 21:492 – 518.

Jen-T. Chien and Chuang-H. Chueh. 2008. Latent Dirichlet Language Model for Speech Recognition. In *Proc. of IEEE SLT Workshop*, pages 201–204.

Jen-T. Chien and Chuang-H. Chueh. 2011. Dirichlet Class Language Models for Speech Recognition. *IEEE Trans. on Audio, Speech and Language Processing*, 19(3):482 – 495.

Jen-T. Chien. 2006. Association Pattern Language Modeling. *IEEE Trans. on Audio, Speech and Language Processing*, 14(5):1719 – 1728.

Jerome R. Bellegarda. 2000. Exploiting Latent Semantic Information in Statistical Language modeling. *IEEE Transactions on Speech and Audio Processing*, 88 (8):1279–1296.

John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallett, Nancy L. Dahlgren, and Victor Zue. 1993. TIMIT Acoustic-phonetic Continuous Speech Corpus. *Linguistic Data Consortium.*

Keith Vertanen. 2013. HTK Wall Street Journal Training Recipe. `http://www.keithv.com/software/htk/us/`.

Md. A. Haidar and Douglas O'Shaughnessy. 2011. Unsupervised Language Model Adaptation using N-gram Weighting. In *Proceedings of CCECE*, pages 857–860.

Md. A. Haidar and Douglas O'Shaughnessy. 2012a. LDA-based LM Adaptation using Latent Semantic Marginals and Minimum Discrimination Information. In *Proceedings of EUSIPCO*, pages 2040–2044.

Md. A. Haidar and Douglas O'Shaughnessy. 2012b. Topic N-gram Count Language Model for Speech Recognition. In *Proceedings of IEEE Spoken Language Technology (SLT) Workshop*, pages 165–169.

Md. A. Haidar and Douglas O'Shaughnessy. 2013a. Fitting Long-range Information using Interpolated Distanced n-grams and Cache Models into a Latent Dirichlet Language Model for Speech Recognition. In *Proc. of INTERSPEECH*, pages 2678–2682.

Md. A. Haidar and Douglas O'Shaughnessy. 2013b. PLSA Enhance with a Long-distance Bigram Language Model for Speech Recognition. In *Proc. of EUSIPCO*.

Nikoletta Bassiou and Constantine Kotropoulos. 2010. Word Clustering PLSA Enhanced with Long Distance Bigrams. In *Proc. of International Conferance on Pattern Recognition*, pages 4226–4229.

P.C. Woodland, J. J. Odell, V. Valtchev, and S. J. Young. 1994. Large Vocabulary Continuous Speech Recognition using HTK. In *Proceedings of ICASSP*, pages 125–128.

Peng Xu and Frederick Jelinek. 2007. Random Forests and the Data Sparseness Problem in Language Modeling. *Computer Speech and Language*, 21 (1):105 – 152.

Peter F. Brown, Vincent Della Pietra, Peter De Souza, Jenifer Lai, and Robert L. Mercer. 1992. Classbased n-gram Models of Natural Language. *Computational Linguist.*, 18 (4):467 – 479.

Reinhard Kneser and Hermann Ney. 1995. Improved Backing-off for m-gram Language Modeling. In *Proc. IEEE Int Conf. Acoust., Speech, Signal Process.*, pages 181–184.

Roland Kuhn and Renato D. Mori. 1990. A Cache-based Natural Language Model for Speech Recognition. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 12 (6):570–583.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391 – 407.

Shuanghu Bai, Haizhou Li, Zhiwei Lin, and Baosheng Yuan. 1998. Building Class-based Language Models with Contextual Statistics. In *Proc. IEEE Int Conf. Acoust., Speech, Signal Process*, pages 173–176.

Slava M. Katz. 1987. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *EEE Trans. Acoust., Speech, Signal Process.*, 35(3):400 – 401.

Stanley Chen, 2008. *Performance Prediction for Exponential Language Models*. Tech. Rep. RC 24671, IBM Research, Tech. Rep.

Steve Young, Phil Woodland, Gunnar Evermann, and Mark Gales. 2013. The HTK Toolkit 3.4.1. `http://htk.eng.cam.ac.uk/`.

Thomas Hofmann. 1999. Probabilistic Latent Semantic Analysis. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 289–296, San Francisco, CA. Morgan Kaufmann.

Tomas Mikolov, Martin Karafiat, Lukas Burget, Jan H. Cernocky, and Sanjeev Khudanpur. 2010. Recurrent Neural Network Based Language Model. In *Proc. of INTERSPEECH*, pages 1045–1048.

Tomas Mikolov, Stefan Kombrink, Lukas Burget, Jan H. Cernocky, and Sanjeev Khudanpur. 2011. Extensions Recurrent Neural Network Language Model. In *Proc. of ICASSP*, pages 5528–5531.

Welly Naptali, Masatoshi Tsuchiya, and Seiichi Nakagawa. 2012. Topic Dependent Class-based n-gram Language Model. *IEEE Trans. on Audio, Speech and Language Processing*, 20:1513 – 1525.

Yik-Cheung Tam and Tanja Schultz. 2005. Dynamic Language Model Adaptation using Variational Bayes Inference. In *Proceedings of INTERSPEECH*, pages 5–8.

Yik-Cheung Tam and Tanja Schultz. 2006. Unsupervised Language Model Adaptation using Latent Semantic Marginals. In *Proceedings of INTERSPEECH*, pages 2206–2209.

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137 – 1155.

Situated Incremental Natural Language Understanding using a Multimodal, Linguistically-driven Update Model

Casey Kennington
CITEC, Bielefeld University
ckennington[1]

Spyros Kousidis
Bielefeld University
spyros.kousidis[2]

David Schlangen
Bielefeld University
david.schlangen[2]

[1]@cit-ec.uni-bielefeld.de
[2]@uni-bielefeld.de

Abstract

A common site of language use is interactive dialogue between two people situated together in shared time and space. In this paper, we present a statistical model for understanding natural human language that works incrementally (i.e., does not wait until the end of an utterance to begin processing), and is grounded by linking semantic entities with objects in a shared space. We describe our model, show how a semantic meaning representation is grounded with properties of real-world objects, and further show that it can ground with embodied, interactive cues such as pointing gestures or eye gaze.

1 Introduction

Dialogue between co-located participants is possibly the most common form of language use (Clark, 1996). It is highly interactive (time is shared between two participants), interlocutors can refer to objects in their visual field (space is also shared), and visual cues such as gaze or pointing gestures often play a role (shared time *and* space). Most computational dialogue research focuses only one of these constraints.

In this paper, we present a model that processes incrementally (i.e., can potentially work interactively), can make use of the visual world by symbolically representing objects in a scene, and incorporate gaze and gestures. The model can learn from conversational data and can potentially be used in an application for a situated dialogue system, such as an autonomous robot.

In the following section we will provide background and present related work. That will be followed by a description of the task and the model. In Section 4 we will show how our model performs in two experiments, the first uses speech and a visual scene, the second incorporates visual cues.

2 Background and Related Work

2.1 Background: Incremental Dialogue Processing

Dialogue systems that process incrementally produce behavior that is perceived by human users to be more natural than systems that use a turn-based approach (Aist et al., 2006; Skantze and Schlangen, 2009; Skantze and Hjalmarsson, 2010). Incremental dialogue has seen improvements in speech recognition (Baumann et al., 2009), speech synthesis (Buschmeier et al., 2012), and dialogue management (Buß et al., 2010; Selfridge et al., 2012). Futhermore, architectures for incremental dialogue systems have been proposed (Schlangen and Skantze, 2009; Schlangen and Skantze, 2011) and incremental toolkits are also available (Baumann and Schlangen, 2012).

In this paper, we approach *natural language understanding* (NLU), which aims to map an utterance to an *intention*, as a component in the incremental model of dialogue processing as described in (Schlangen and Skantze, 2011; Schlangen and Skantze, 2009), where incremental systems consist of a network of processing *modules*. Each module has a *left buffer* and a *right buffer*, where a typical module takes input

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1803–1812, Dublin, Ireland, August 23-29 2014.

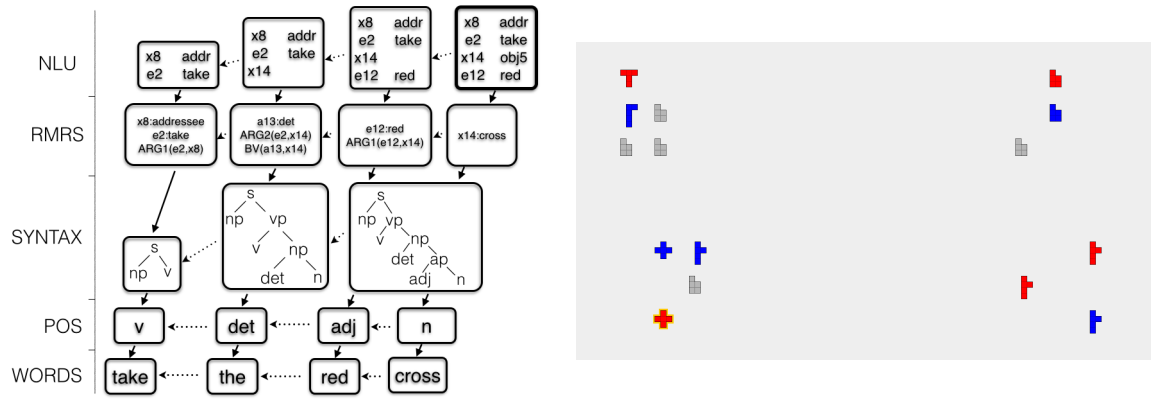


Figure 1: Example of an IU network composed of words, parts of speech (POS), a semtic representation (*Robust Minimal Recursion Semantics*; RMRS), and NLU modules. Solid arrows represent GRIN links and the dotted lines represent SLLs. The utterance *take the red cross* is represented as word IUs, which are GRIN by the part of speech tags, phrase-structure parse, semantic representation, and the intention. Note that *red* and *cross* are GRIN by the same syntactic IU, which in turn is GRIN by two semantic IUs. Succeeding levels of IUs are shifted slightly to the right, representing a processing delay. The X14 slot in the bolded NLU frame refers to the cross-shaped object in the game board on the right.

from its left buffer, performs some kind of processing on that data, and places the processed result onto its right buffer. The data are packaged as the payload of *incremental units* (IU) which are passed between modules. The IUs themselves are also interconnected via so-called *same level links* (SLL) and *grounded-in links* (GRIN), the former allowing the linking of IUs as a growing sequence, the latter allowing that sequence to convey what IUs directly affect them. See Figure 1 for an example; each layer represents a module in the IU-module network and each node is an IU in the IU network. The focus of this paper is the top layer (module), but how it is produced depends on the layers below it.

2.2 Related Work

The work presented in this paper connects and extends recent work in *grounded semantics* (Roy, 2005; Hsiao et al., 2008; Liu et al., 2012; Chai et al., 2014), which aims to connect language with the world, but typically does not work incrementally; *semantic parsing / statistical natural language understanding* via logical forms (Zettlemoyer and Collins, 2007; Zettlemoyer and Collins, 2009), *dependency-based compositional semantics* (Liang et al., 2011), *neural networks* (Huang and Er, 2010), *Markov Logic Networks* (Meurs et al., 2008; Meza-Ruiz et al., 2008), and *Dynamic Bayesian Networks* (Meurs et al., 2009); see also overviews of NLU in (De Mori et al., 2008; Tur and De Mori, 2011), but typically neither provide situated interpretations nor incremental specifications of the representations; *incremental* NLU (DeVault et al., 2009; DeVault et al., 2011; Aist et al., 2007; Schlangen and Skantze, 2009), which focuses on incrementality, but not on situational grounding; as well as integration of *gaze* into language understanding (Prasov and Chai, 2010).

We move beyond this work in that we present a model that is incremental, uses a form of grounded semantics, can easily incorporate multi-modal information sources, and which inference can be performed quickly, satisfying the demands of real-time dialogue.

3 Task and Model

3.1 Task

The task for our model is as follows: to compute at any moment a distribution over possible intentions which the speaker wanted to convey in the utterance, expressed as semantic *frames*, given the unfolding utterance and information about the state of the world in which the utterance is happening. The *slots* of these frames are to be filled with semantic constants, that is, they are uniquely resolved, if appropriate, to objects in the shared environment. This is illustrated in Figure 1 where the words of the utterance give

rise to the part-of-speech tags, the incrementally growing syntax, semantic representation, and, finally, the intention. Note how x14 in the bolded NLU frame resolves to an object identifier for a real object in the shared scene (red cross in the bottom-left of the game board shown on the right in the figure).

3.2 Model

Kennington et al., (2013) presented a simple, incremental model of NLU, which is an update model (i.e., increments build on previous ones) and which can potentially work in real time and in situated environments. The goal of the model is to recover I, the intention of the speaker behind the utterance, word by word. We observe U, the current word (or in this paper, a semantic meaning representation, see below) and an unobserved mediating variable R which represents visual or abstract properties of the object of the intention. Formally, we are interested in $P(I|U)$, the probability of a certain intention I underlying utterance U. We assume a latent variable R (p*R*operties of entities in the world), and build a generative model (that is, model the joint $P(I, R, U)$). Going from $P(I, R|U)$ and making certain independence assumptions, we arrive at

$$P(I|U) = \frac{P(I)}{P(U)} \sum_{r \in R} P(U|R = r)P(R = r|I) \tag{1}$$

That is, we assume that R is only conditional on I, and U is only conditional on R, and we can move $P(I)$ and $P(U)$ out of the summation, as they do not depend on R. This is an update model in the usual sense that the posterior ($P(I|U)$) at one step becomes the prior ($P(I)$) at the next. $P(R|I)$ provides the link between the intentions and the properties.

Another variant of the model which we will use in this paper is as follows: we rewrite $P(U|R)$ using Bayes' rule, which cancels $P(U)$ and introduces $P(R)$ into the summation, but $P(R)$ can be dropped since (in this work) it can be approximated with a uniform distribution, yielding:

$$P(I|U) = P(I) \sum_{r \in R} P(R = r|U)P(R = r|I) \tag{2}$$

There are, however, three important differences between the realisation of our model and the one presented in Kennington et al., (2013), all of which are a direct result of replacing, as we do here, the n-gram model represented by $P(U|R)$ with output from a parser that produces a *Robust Minimal Recursion Semantics* (RMRS) semantic representation (Copestake, 2007). Such a representation provides our model with a structured way to abstract over the surface forms. We will first give a brief explanation of the RMRS framework, then describe each of the three differences between our model and that of Kennington et al., (2013), namely (1) how the language grounds with the world, (2) how the frame is built, and (3) when to consider evidence for the slots in the frame.

RMRS RMRS is a framework for representing semantics that factors a logical form into *elementary predicates* (EP). For example in Table 1, the first row represents the first word of an utterance, *take*, and the corresponding RMRS representation; the EPs *take* and *addressee* are produced. The EPs in this example have *anchor* variables and in most cases, an EP has an argument *entity*. Relations between EPs can be expressed via *argument relations*, e.g., for *take* in the table, there is an ARG1 relation, denoting *addressee* as the first argument of the predicate *take*. Other relations include ARG2 and BV (relating determiners to the words they modify). A full example of an utterance and corresponding RMRS representation can be found in Table 1, where each row in the word column makes up the words of the example utterance.

In this paper we are interested in processing utterances incrementally. As argued in Peldzsus et al., (2012), RMRS is amenable to incremental processing by allowing for *underspecification* in how relations are represented (RMRS can also underspecify scope, but we don't consider that here). Table 1 has an example of an underspecified relation: when the second word *the* is uttered, the RMRS segment predicts that the entity represented by x14 will be the ARG2 relation of the EP for *take*, but the actual word that

word	RMRS segment
take	$a7 : addressee(x8), a1 : take(e2), ARG1(a1, x8)$
the	$a13 : def(), ARG2(a1, x14), BV(a13, x14)$
red	$a33 : red(e34), ARG1(a33, x14)$
cross	$a19 : cross(x14)$
next to	$a49 : next(e50), ARG1(a49, x14), ARG2(a49, x53)$
the	$a52 : def(), BV(a52, x53)$
blue	$a72 : blue(e73), ARG1(a72, x53)$
piece	$a58 : piece(x53)$

Table 1: Example RMRS representation for the utterance *take the red cross next to the blue piece*. Each row represents an increment of the utterance.

produces the EP that has `x14` as an argument has not yet been uttered. Each row in the table represents what we would want an RMRS parser to produce for our model at each word increment.

A more detailed explanation of RMRS can be found in Copestake (2007). We will now discuss the three key differences of our model with that of previous work.

(1) Grounding Semantics with the Visual World In Kennington et al., (2013), the utterance was represented via n-grams, which was used to ground with the world. Here, we ground RMRS structures with the world. For example, Figure 1 shows which words produced which RMRS increments; our model learns the co-occurances between those increments and properties of objects (real properties such as colors, shapes, and spatial placements, or abstract properties; e.g., `take` is a property of the action *take*).

(2) Building the Frame In this paper, intentions are represented as frames. However, unlike Kennington et al., (2013), we don't assume beforehand that we know the slots of the frame. To determine the slots, we turn again to RMRS and build a slot for each *entity* that is produced (more on this below). This kind of frame, coupled with the RMRS representation, shows not just a meaning representation, but also *interpretation* of the representation in the current model (the real situation / visual domain of discourse), outputted incrementally making our model *fully* incremental in the sense of Heintze et al., (2010). The final, bolded NLU frame in Figure 1 shows the addressee (in this case, the dialogue system) as the recipient of the request, the request itself is a `take` request, where the object to be taken is `obj5`, as indexed by the real world, and that object happens to be red (i.e., `e12` represents the notion of *redness*).

(3) Driven by Sematics Another important difference is *when* to consider the semantic evidence and when to ignore it, in terms of when to apply the model for interpretation of the slots. In Kennington et al., (2013), each slot in the frame was processed at each increment in the entire utterance, regardless of whether n-grams in that segment contributed to the interpretation of that slot. In our approach, again, we turn to RMRS. At each word increment, RMRS produces a corresponding, underspecified semantic meaning represenation which is added to at the next increment. Our model takes the new information and only attempts to process the interpretation for those "active" entities. For example, by the time *red* is uttered in Figure 1, the processing for entities `x8`, `e2`, and `e12` is complete, but the processing for `x14` is under way, and active as long as `x14` is referenced as an entity in the RMRS increment.

With these important extensions, our model of NLU is highly driven by the semantic meaning representation that is being built incrementally for the utterance. We will now show through two experiments how our approach improves upon previous work.

4 Experiments

Similar to Kennington et al., (2013), we use the model represented formally in Equation 2, where $P(R|U)$ is realised using a maximum entropy classifier (ME) that predicts properties from RMRS evidence.[1] We use the German RMRS parser described in Peldszus et al (2012), Peldszus and Schlangen (2012) which is a top-down PCFG parser that builds RMRS structure incrementally with the parse.

We train an individual model for each RMRS entity type (e.g., `e` and `x`), where the features are the entity type, relations, and predicates of an RMRS increment and the class label are the visual properties.

[1] http://opennlp.apache.org/

The RMRS representations are not checked for accuracy (i.e., they do not represent ground truth); we use the top-predicted output of the RMRS parser explained in Peldszus et al (2012).

4.1 Pento Puzzle with Speech

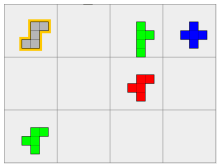


Figure 2: Example Pentomino Board

ACTION	rotate
OBJECT	obj4
RESULT	clockwise

Figure 3: Pento gold frame example

X8	addr
E2	rotate
X14	obj4
E21	clockwise

Figure 4: Pento frame example from our model

Data and Task The *Pentomino* domain (Fernández et al., 2007) contains task-oriented conversational data which has been used in several situated dialogue studies (Heintze et al., 2010; Peldszus et al., 2012; Kennington and Schlangen, 2012; Kennington et al., 2013). This corpus was collected in a Wizard-of-Oz study, where the user goal was to instruct the computer to pick up, delete, rotate or mirror puzzle tiles on a rectangular board (as in Figure 2), and place them onto another board. For each utterance, the corpus records the state of the game board before the utterance, the immediately preceding system action, and the intended interpretation of the utterance (as understood by the Wizard) in the form of a semantic frame specifying action-type and arguments, where those arguments are objects occurring in the description of the state of the board. The language of the corpus is German. See Figure 2 for a sample source board, and Figure 3 for an annotated frame.

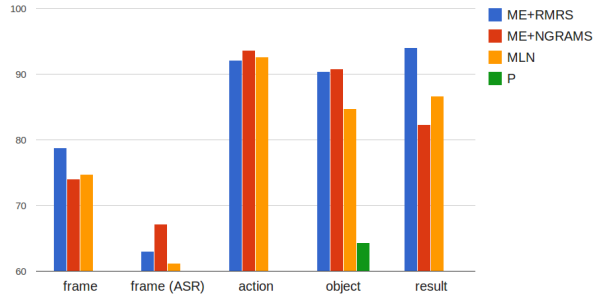
The task that we want our model to perform is as follows: given information about the state of the world (i.e., game board), previous system action, and the ongoing utterance, incrementally build the frame by providing the interpretation of each RMRS entity, represented as a distribution over all possible interpretations for that entity (i.e., domain of discourse).

Procedure To make our work comparable to previous work, results were obtained by averaging the results of a 10-fold validation on 1489 Pento boards (i.e., utterances+context, as in (Kennington and Schlangen, 2012)). We used a separate set of 168 boards for small-scale, held-out experiments. For incremental processing, we used INPROTK.[2] We calculate accuracies by comparing against a gold frame, with assumptions. We check to see if the slot values (3 slots in total) exist in the frame our model produces. If a gold slot value exists in any slot produced by our model, it is counted as correct (it is difficult to tell which slot from our model's frame maps to which slot in the gold frame, we leave this for future work). A fully correct frame would contain all three values. For example, each of the values for the gold slots in Figure 3 exist in the example frame our model would produce in Figure 4, marking each gold slot as correct, and the entire frame as correct since all three were correct together. To directly compare with previous work, we will use the gold slot names `action`, `object`, and `result` in the Results section. We perform training and evaluation on hand-transcribed data and on automatically transcribed data, using the incremental speech recogniser (Sphinx4) in InproTK. We report results on sentence-level and incremental evaluations.

On the incremental level, we followed previously used metrics for evaluation:
first correct: how deep into the utterance do we make the first correct guess?
first final: how deep into the utterance do we make the correct guess, without subsequent changes?
edit overhead: what is the ratio of unnecessary edits / sentence length, where the only *necessary* edit is the first prediction for an entity?

Results Figure 5 shows the results of our evaluation in graph and table form. As expected, our model dramatically improved the `result` value, which generally is verbally represented towards the end of

[2] `https://bitbucket.org/inpro/inprotk`



	ME+RMRS	ME+NGRAMS	MLN	P
frame	**78.75**	74.08	74.76	
	(63.0)	**(67.2)**	(61.2)	
action	92.11	**93.62**	92.62	
object	90.44	**90.79**	84.71	64.3
result	**94.0**	82.34	86.65	

Figure 5: Comparison of accuracies in Pento using the model presented here **ME+RMRS**, (Kennington et al., 2013) **ME+NGRAMS**, (Kennington and Schlangen, 2012) **MLN**, (Peldszus et al., 2012) **P**; parentheses denote results from automatically transcribed speech. Bolded values represent the highest values for that row. Note that the column chart begins at 60%. The chart and table show the same information.

an utterance. This resulted in a dramatic increase in frame accuracy (a somewhat strict metric). Our model fares better than previous work using speech (in parentheses in the figure), but is outperformed by the n-gram approach. These results are encouraging, however we leave improvements on automatically transcribed speech to future work.

Incremental Table 2 shows the incremental results of Kennington et al.,(2013), and Table 3 shows our results. Utterances are binned into short, normal, and long utterance lengths (1-6, 7-8, 9-17 words, respectively; 7-8 word utterances were the most represented). Previous work processed all three slots throughout the ongoing utterance, whereas the model presented here only processed entities (that could give rise to these slots) as dictated by the RMRS. This causes a later overall *first correct*, but an overall earlier *first final*, with a much narrower window between them. This represents an ideal system that waits for processing a slot until it needs to, but comes to a final decision quickly, without changing its mind later. This is further evidenced by the *edit overhead* which is lower here than previous work. This has implications in real-time systems that need to define *operating points*; i.e., a dialogue system would need to wait for specific information before making a decision.

action	**1-6**	**7-8**	**9-14**
first correct (% into utt.)	5.78	2.56	3.64
first final (% into utt.)	38.26	36.10	30.84
edit overhead	2.37		
object	**1-6**	**7-8**	**9-14**
first correct (% into utt.)	7.39	7.5	10.11
first final (% into utt.)	44.7	44.18	35.55
edit overhead	4.6		
result	**1-6**	**7-8**	**9-14**
first correct (% into utt.)	15.16	23.23	20.88
first final (% into utt.)	42.55	40.57	35.21
edit overhead	10.19		

Table 2: Incremental Results for Pento slots with varying sentence lengths, Kennington et al.,(2013), Edit overhead represents all lengths of utterances.

action	**1-6**	**7-8**	**9-14**
first correct (% into utt.)	12.03	7.8	12.59
first final (% into utt.)	37.84	26.02	24.11
edit overhead	1.57		
object	**1-6**	**7-8**	**9-14**
first correct (% into utt.)	30.64	17.66	14.46
first final (% into utt.)	32.27	19.20	15.79
edit overhead	3.1		
result	**1-6**	**7-8**	**9-14**
first correct (% into utt.)	59.72	54.50	48.94
first final (% into utt.)	62.80	64.13	60.72
edit overhead	7.71		

Table 3: Incremental Results for Pento slots with varying sentence lengths, current work. Edit overhead represents all lengths of utterances.

4.2 Pento Puzzle with Speech, Gaze, and Deixis

Data and Task The second experiment uses data also from the Pentomino domain, as described in (Kousidis et al., 2013; Kennington et al., 2013), also a Wizard-of-Oz study consisting of 7 participants, example in Figure 1. The user was to select a puzzle tile (out of a possible 15) on a game board shown on a large monitor, and then describe this piece to the "system" (wizard). Speech, eye gaze (tracked by *Seeingmachines FaceLab*) and pointing gestures (tracked by *Microsoft Kinect*) were recorded. After the participant uttered a confirmation, the wizard began a new episode, generating a new random board and

the process repeated.

The task for the NLU in this experiment was reference resolution. The information available to our model for these data included the utterance (hand-transcribed) the visual context (game board), gaze information, and deixis (pointing) information, where a rule-based classifier predicted from the motion capture data the quadrant of the screen at which the participant was pointing. These data were very noisy (and hence, realistic) despite the constrained conditions of the task; the participants were not required to say things a certain way (as long as it was understood by the wizard), their hand movements potentially covered their faces which interfered with the eye tracker, and each participant had a different way of pointing (e.g., different gesture space, handedness, distance of hand from body when pointing, alignment of hand with face, etc.).

Procedure Removing the utterances which were flagged by the wizard (i.e., when the wizard misunderstood the participant) and the utterances of one of the participants (who had misunderstood the task) left a total of 1051 utterances. We used 951 for development and training the model, and 100 for evaluation. We give results as resolution accuracy. All models were trained on hand-transcribed data, but two evaluations were performed: one with hand-transcribed data, and one with speech automatically transcribed by the Google Web Speech API.[3] Gaze and deixis are incorporated by incrementally computing properties to be provided to our NLU model; i.e., a tile has a property in R of being `gazed_at` if it is gazed at for some interval of time, or tiles in a quadrant of the screen have the property of being `pointed_at`. Figure 6 shows an example utterance, gaze, and gesture activity over time and how they are reflected in the model. Our baseline model is the NLU without using gaze or deixis information; random accuracy is 7%. We will compare our model with that of an NGRAM (up to trigram) model in the evaluations, for each of the conditions (baseline, deixis, gaze, deixis and gaze).

We also include the percentage of the time the gold tile is in the top 2 and top 4 rankings (out of 15); situations in which a dialogue system could at least provide alternatives in a clarification request (if it could detect that it should have low confidence in the best prediction; which we didn't investigate here). For gaze, we also make the naive assumption that over the utterance the participant (who in this case is the speaker) will gaze at his chosen intended tile most of the time.

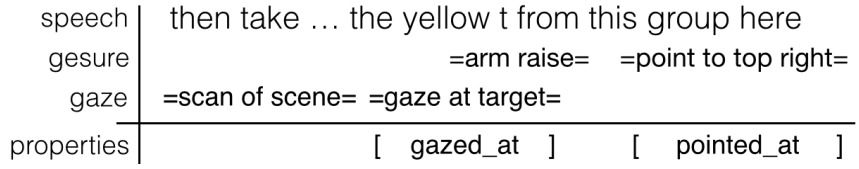


Figure 6: Human activity (top) aligned with how modalities are reflected in the model for Gaze and Point (bottom) over time for example utterance: *take the yellow* t *from this group here*. The intervals of the properties are denoted by square brackets.

Results Table 4 shows the results of our evaluation. Overall, the model that uses RMRS outperforms the model that uses NGRAMS under all conditions using hand-transcribed data. The results for speech tell a different story; speech with NGRAMS is generally better – an effect of the model here relying on parser output. Overall, both model types increase performance when using hand-transcribed or automatically-transcribed speech when incorporating other modalities, particularly pointing. Furthermore, the Top 2 and Top 4 columns show that this model has an overall good distribution, especially in the case of RMRS and pointing, where the target object is in the top four ranks 90% of the time. This would allow a real-time system to ask a specific clarification request to the human, with a high confidence that the object is among the top four ranking objects.

Incremental For further incremental results, Figure 7 shows the rank of each object on an example board using our baseline model for the utterance *nimm das rote untere kreuz* (take the red below cross /

[3]The Web Speech API Specificiation: `https://dvcs.w3.org/hg/speech-api/raw-file/tip/speechapi.html`

NLU	Acc	Top 2	Top 4
NGRAMS	68%	83%	87%
(speech) NGRAMS	44%	57%	69%
RMRS	73%	82%	88%
(speech) RMRS	36%	54%	66%
NLU + Pointing	**Acc**	**Top 2**	**Top 4**
NGRAMS	70%	83%	88%
(speech) NGRAMS	46%	60%	72%
RMRS	**78%**	**85%**	**90%**
(speech) RMRS	40%	56%	73%

NLU + Gaze	Acc	Top 2	Top 4
NGRAMS	68%	84%	88%
(speech) NGRAMS	43%	59%	71%
RMRS	74%	81%	88%
(speech) RMRS	39%	54%	67%
NLU + Gaze + Point	**Acc**	**Top**	**Top**
NGRAMS	70%	84%	87%
(speech) NGRAMS	45%	61%	65%
RMRS	77%	**85%**	89%
(speech) RMRS	41%	56%	74%

Table 4: Results for Experiment 2. The highest scores for each column are in bold. Four evaluations are compared under four different settings; **Acc** denotes accuracy (referent in top position), **Top 2** and **Top 4** respectively show the percentage of time the referent was between those ranks and the top.

take the red cross below). Once *das* (the) is uttered, RMRS makes an X entity and the model begins to interpret. The initial distribution appears to be quite random as *das* does not have high co-occurence with any particular object property. Once *rote* (red) is uttered, all non-red objects fall to the lowest ranks in the distribution. Once *untere* (under / below) is uttered, all of the red pieces in the bottom two quadrants increase overall in rank. Finally, as *kreuz* (cross) is uttered, the two crosses receive the highest ranks, the bottom one being the highest rank and intended object. Note the rank of the cross in the top left quadrant over time; it began with a fairly high rank, which moved lower once *untere* was uttered, then moved into second rank once *kreuz* was uttered. As the utterance progresses the rank of the intended object decreases, showing that our model predicted the correct piece at the appropriate word.

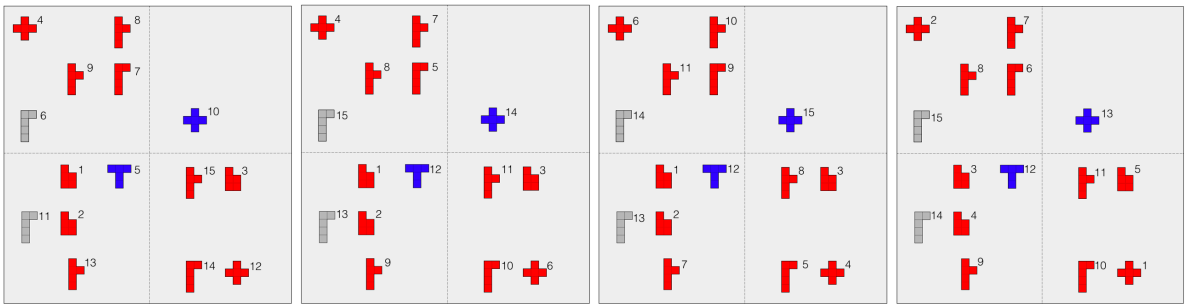


Figure 7: Example of reference resolution for the utterance: *nimm das rote untere kreuz / take the red below cross*; objects are annotated with their rank in the distribution as outputed by the NLU model at each increment. The board size has been adjusted for formatting purposes.

5 Discussion and Conclusions

We have presented a model of NLU that uses a semantic representation to recover the intention of a speaker utterance. Our model is general in that it doesn't fit a template or ontology like other NLU approaches (though we would need to determine how a dialogue manager would make use of such a frame), and grounds the semantic representation with a symbolic representation of the visual world. It works incrementally and can incorporate other modalities incrementally. It improves overall upon previous work that used a similar model, but relied on n-grams. Our model implicitely handles complex utterances that use spatial language. However, we leave important aspects, such as negation in an utterance, to future work (they were not very common in our data).

The experiments in this paper were done off-line, but we have a real-time system currently working. Our model incorporates in real-time the gesture and gaze information as it is picked up by the sensors, as well as the speech of the user. We leave a full evaluation using this interactive setup with human participants for future work.

Acknowledgements Thanks to the anonymous reviewers for their useful comments.

References

Gregory Aist, James Allen, Ellen Campana, Lucian Galescu, Carlos A Gomez Gallo, Scott Stoness, Mary Swift, and Michael Tanenhaus. 2006. Software architectures for incremental understanding of human speech. In *Proceedings of InterspeechICSLP*.

Gregory Aist, James Allen, Ellen Campana, Carlos Gomez Gallo, Scott Stoness, Mary Swift, and Michael K Tanenhaus. 2007. Incremental understanding in human-computer dialogue and experimental evidence for advantages over nonincremental methods. In *Proceedings of Decalog (Semdial 2007)*, Trento, Italy.

Timo Baumann and David Schlangen. 2012. The InproTK 2012 Release. In *NAACL*.

Timo Baumann, Michaela Atterer, and David Schlangen. 2009. Assessing and Improving the Performance of Speech Recognition for Incremental Systems. In *Proceedings of NAACL-HLT 2009*, Boulder, USA, June.

Hendrik Buschmeier, Timo Baumann, Benjamin Dosch, Stefan Kopp, and David Schlangen. 2012. Combining Incremental Language Generation and Incremental Speech Synthesis for Adaptive Information Presentation. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 295–303, Seoul, South Korea, July. Association for Computational Linguistics.

Okko Buß Timo Baumann, and David Schlangen. 2010. Collaborating on Utterances with a Spoken Dialogue System Using an ISU-based Approach to Incremental Dialogue Management. In *Proceedings of the SIGdial 2010 Conference*, pages 233–236, Tokyo, Japan, September.

Joyce Y Chai, Lanbo She, Rui Fang, Spencer Ottarson, Cody Littley, Changsong Liu, and Kenneth Hanson. 2014. Collaborative Effort towards Common Ground in Situated Human-Robot Dialogue. In *HRI'14*, pages 33–40, Bielefeld, Germany.

Herbert H Clark. 1996. *Using Language*. Cambridge University Press.

Ann Copestake. 2007. Semantic composition with (robust) minimal recursion semantics. In *Proceedings of the Workshop on Deep Linguistic Processing - DeepLP '07*, page 73, Morristown, NJ, USA. Association for Computational Linguistics.

Renato De Mori, Frederic Béchet, Dilek Hakkani-tür, Michael Mctear, Giuseppe Riccardi, and Gokhan Tur. 2008. Spoken Language Understanding. *IEEE Signal Processing Magazine*, (May):50–58, May.

David DeVault, Kenji Sagae, and David Traum. 2009. Can I finish?: learning when to respond to incremental interpretation results in interactive dialogue. In *Proceedings of the 10th SIGdial*, number September, pages 11–20. Association for Computational Linguistics.

David DeVault, Kenji Sagae, and David Traum. 2011. Incremental Interpretation and Prediction of Utterance Meaning for Interactive Dialogue. *Dialogue & Discourse*, 2(1):143–170.

Raquel Fernández, Tatjana Lucht, and David Schlangen. 2007. Referring under restricted interactivity conditions. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*, pages 136–139.

Silvan Heintze, Timo Baumann, and David Schlangen. 2010. Comparing local and sequential models for statistical incremental natural language understanding. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 9–16. Association for Computational Linguistics.

Kai-yuh Hsiao, Soroush Vosoughi, Stefanie Tellex, Rony Kubat, and Deb Roy. 2008. Object schemas for grounding language in a responsive robot. *Connection Science2*, 20(4):253–276.

Guangpu Huang and Meng Joo Er. 2010. A Hybrid Computational Model for Spoken Language Understanding. In *11th International Conference on Control, Automation, Robotics, and Vision*, number December, pages 7–10, Singapore. IEEE.

Casey Kennington and David Schlangen. 2012. Markov Logic Networks for Situated Incremental Natural Language Understanding. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 314–322, Seoul, South Korea. Association for Computational Linguistics.

Casey Kennington, Spyros Kousidis, and David Schlangen. 2013. Interpreting Situated Dialogue Utterances: an Update Model that Uses Speech, Gaze, and Gesture Information. In *SIGdial 2013*.

Spyros Kousidis, Casey Kennington, and David Schlangen. 2013. Investigating speaker gaze and pointing behaviour in human-computer interaction with the mint.tools collection. In *SIGdial 2013*.

Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning Dependency-Based Compositional Semantics. In *Proceedings of the 49th ACLHLT*, pages 590–599, Portland, Oregon. Association for Computational Linguistics.

Changsong Liu, Rui Fang, and Joyce Chai. 2012. Towards Mediating Shared Perceptual Basis in Situated Dialogue. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 140–149, Seoul, South Korea, July. Association for Computational Linguistics.

Marie-Jean Meurs, Frederic Duvert, Fabrice Lefevre, and Renato De Mori. 2008. Markov Logic Networks for Spoken Language Interpretation. *Information Systems Journal*, (1978):535–544.

Marie-Jean Meurs, Fabrice Lefèvre, and Renato De Mori. 2009. Spoken Language Interpretation: On the Use of Dynamic Bayesian Networks for Semantic Composition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 4773–4776.

Ivan Meza-Ruiz, Sebastian Riedel, and Oliver Lemon. 2008. Accurate Statistical Spoken Language Understanding from Limited Development Resources. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 5021–5024. IEEE.

Andreas Peldszus and David Schlangen. 2012. Incremental Construction of Robust but Deep Semantic Representations for Use in Responsive Dialogue Systems. In *Proceedings of the Workshop on Advances in Discourse Analysis and its Computational Aspects*, pages 59–76, Mumbai, India, December. The COLING 2012 Organizing Committee.

Andreas Peldszus, Okko Buß, Timo Baumann, and David Schlangen. 2012. Joint Satisfaction of Syntactic and Pragmatic Constraints Improves Incremental Spoken Language Understanding. In *Proceedings of the 13th EACL*, pages 514–523, Avignon, France, April. Association for Computational Linguistics.

Zahar Prasov and Joyce Y Chai. 2010. Fusing Eye Gaze with Speech Recognition Hypotheses to Resolve Exophoric References in Situated Dialogue. In *EMNLP 2010*, number October, pages 471–481.

Deb Roy. 2005. Grounding words in perception and action: computational insights. *Trends in Cognitive Sciences*, 9(8):389–396, August.

David Schlangen and Gabriel Skantze. 2009. A General, Abstract Model of Incremental Dialogue Processing. In *Proceedings of the 10th EACL*, number April, pages 710–718, Athens, Greece. Association for Computational Linguistics.

David Schlangen and Gabriel Skantze. 2011. A General, Abstract Model of Incremental Dialogue Processing. *Dialoge & Discourse*, 2(1):83–111.

Ethan O Selfridge, Iker Arizmendi, Peter A Heeman, and Jason D Williams. 2012. Integrating Incremental Speech Recognition and POMDP-Based Dialogue Systems. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 275–279, Seoul, South Korea, July. Association for Computational Linguistics.

Gabriel Skantze and Anna Hjalmarsson. 2010. Towards Incremental Speech Generation in Dialogue Systems. In *Proceedings of SigDial 2010*, pages 1–8, Tokyo, Japan, September.

Gabriel Skantze and David Schlangen. 2009. Incremental dialogue processing in a micro-domain. *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics on EACL 09*, (April):745–753.

Gokhan Tur and Renato De Mori. 2011. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. Wiley.

Luke S Zettlemoyer and Michael Collins. 2007. Online Learning of Relaxed CCG Grammars for Parsing to Logical Form. *Computational Linguistics*, (June):678–687.

Luke S Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. *Proceedings of the Joint Conference of the 47th ACL and the 4th AFNLP: Volume 2 - ACL-IJCNLP '09*, 2:976.

Quality Estimation for Automatic Speech Recognition

Matteo Negri[(1)] **Marco Turchi**[(1)] **José G. C. de Souza**[(1,2)] **Daniele Falavigna**[(1)]
[(1)] FBK - Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
[(2)] University of Trento, Italy
{negri,turchi,desouza,falavi}@fbk.eu

Abstract

We address the problem of estimating the quality of Automatic Speech Recognition (ASR) output at utterance level, without recourse to manual reference transcriptions and when information about system's confidence is not accessible. Given a source signal and its automatic transcription, we approach this problem as a regression task where the word error rate of the transcribed utterance has to be predicted. To this aim, we explore the contribution of different feature sets and the potential of different algorithms in testing conditions of increasing complexity. Results show that our automatic quality estimates closely approximate the word error rate scores calculated over reference transcripts, outperforming a strong baseline in all the testing conditions.

1 Introduction

In recent years, the increasing usage of large vocabulary continuous speech recognition (LVCSR) systems to transcribe audio recordings from different sources (*e.g.* Youtube videos, TV programs, DVD movies, meetings, etc) has sparked the need of accurate, fast and cost-effective methods to estimate the quality of ASR output. This need contrasts with the fact that, after decades of progress in ASR research, the established evaluation protocol is based on computing word error rate scores (WER)[1] over large test sets of hand-crafted reference transcriptions. Indeed, despite its reliability, reference-based performance assessment has an evident drawback represented by the cost of acquiring manual transcripts. Besides increasing the cost-effectiveness of ASR evaluation routines, bypassing this bottleneck has several other motivations. From an application perspective, for instance, reference-free quality estimation methods could be used to: *i)* decide at run-time whether a given input signal has been properly recognized (*e.g.* if a user spoken utterance needs to be repeated in a dialogue application), *ii)* decide if an automatic transcription is acceptable as is (*e.g.* if manual revision is needed in an automatic subtitling application), or *iii)* select the best transcription among options from multiple ASR systems.

When information about the inner workings of the system used to produce the transcriptions is accessible, current reference-free *confidence estimation* methods can supply ASR applications with reliable indicators about output reliability. This condition, however, does not always hold in the aforementioned scenarios. A clear motivating example is provided by the exponential growth of captioned TED Talks and Youtube videos,[2] for which no information is available about how transcriptions have been produced. In this case, neither reference-based methods, nor standard confidence measures can be applied to obtain useful quality estimates. Nevertheless, in this scenario, supplying reliable indicators of transcription quality has a huge market potential (*e.g.* to reduce the costs of manual revision/translation) which motivates our research.

Focusing on these compelling needs, **this paper investigates the automatic prediction of ASR output quality when: *i)* manual reference transcripts are not available and *ii)* information about the**

[1] The word error rate is the minimum edit distance between an hypothesis and the reference transcription. Edit distance is calculated as the number of edits (word insertions, deletions, substitutions) divided by the number of words in the reference.

[2] Since 2009, Youtube videos in English can be automatically captioned. In 2012, for the 72 hours of video uploaded per minute, such functionality was already available for 10 languages. Currently, more than 200 million Youtube videos have either automatic or human-created captions (source: http://goo.gl/9swYSS).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1813–1823, Dublin, Ireland, August 23-29 2014.

inner workings of the ASR system is not accessible. Casting the problem as a supervised regression task, we experiment in a range of testing conditions on a well-known LVCSR setting (*i.e.* the automatic transcription of TED talks). In this framework, we analyse the performance of various models (*i.e.* their capability to predict utterance-level WER scores) as a function of the different learning algorithms used, the proposed features, and the amount of training data available.

Our features are categorized according to the type of information they aim to capture. Since the nature of the proposed features is a relevant aspect for the applicability of our approach, an important distinction is made between "glass-box" and "black-box" features, which are respectively informed and agnostic about systems' internal decoding strategies. The former can play an important role when all the intermediate processing steps are accessible (*e.g.* in the selection of the best possible transcription hypothesis). In contrast, black-box features have a wider applicability to situations where such information is not available (*e.g.* to estimate the quality of online video subtitles).

Another important aspect relevant to our study is the relation between the accuracy of utterance-level quality predictions and the degree of homogeneity of training and test data. Indeed, as in any supervised learning framework, the similarity between training and test data has a direct impact on (classification and regression) results. In order to fully understand the potential of our approach, we hence measure performance variations under different levels of similarity between the data used to train the regressor and the data used for evaluation. To this aim, our experiments account for a range of possible conditions. These vary from the situation in which training and test are fully homogeneous (*i.e.* same dataset, with training instances produced by the same ASR system) to the more challenging situation where training and test are not homogeneous (*i.e.* different datasets, with training instances produced by different ASR systems). Our results, obtained with two different state-of-the-art algorithms for regression, demonstrate that in all such variable conditions our ASR quality estimation models lead to accurate predictions (*i.e.* close the word error rate scores calculated over reference transcripts).

To the best of our knowledge, this paper represents the first extensive investigation on *reference-free* and *system-agnostic* automatic estimation of ASR output quality. Along this direction, our main contributions can be summarized as follows:

1. We propose a supervised, application-oriented approach to ASR quality estimation that bypasses the need of manual reference transcriptions and is system-independent.

2. We evaluate our method with different learning algorithms and in different conditions, showing that its estimates closely approximate the WER scores calculated over reference transcripts.

3. We perform feature analysis, isolating the contribution of each feature set in all the testing conditions.

4. We analyse the learning curves of our best models, investigating the relation between performance results and the amount of data needed for training.

Overall, these contributions provide useful insights about the feasibility of automatic ASR quality estimation, opening interesting research avenues relevant for system development and for ASR applications.

2 Related Work

As a *reference-free* automatic evaluation method, our work introduces a valid application-oriented alternative to the standard evaluation protocols used within current ASR evaluation campaigns such as IWSLT (Federico et al., 2011; Federico et al., 2012; Cettolo et al., 2013).[3] Besides that, our approach to ASR *quality* estimation (QE) also differs from the well-established *confidence* estimation (CE) techniques proposed in previous ASR literature (Sukkar and Lee, 1996; Evermann and Woodland, 2000; Wessel et al., 2001; Sanchis et al., 2012; Seigel, 2013, *inter alia*). Such difference firstly relies in the fact that, while in CE is the system itself that provides an indicator of the reliability of its output transcriptions, QE aims to provide an external and more objective measure of goodness through WER predictions. A

[3] See http://www.iwslt2013.org/ for details about the last edition of the IWSLT Workshop held in 2013.

second (related) difference is that, in contrast with previous CE methods that heavily rely on information about the internal behaviour of the ASR system, our technique does not necessarily depend on the access to such information. This extends its applicability to scenarios (out of the scope of CE research) where the quality of transcriptions produced by (possibly unknown) ASR systems has to be evaluated/compared solely based on information about the input audio signals and the output transcriptions.

An interesting approach exploiting ASR word accuracy estimates to automatically score the proficiency of non-native English speakers has been proposed by Yoon et al. (2010). To our knowledge this work is the most similar to the one presented here, although it differs in the application domain and several other aspects. First of all, similar to CE methods, it makes some use of glass-box features derived from knowledge about the ASR internal workings (*e.g.* word confidence and acoustic/language model probabilities). Secondly, the domain addressed is constrained to responses to prompted utterances, while in this paper we address a large unconstrained domain, namely the automatic transcription of lectures (TED talks) covering different topics. Finally, (Yoon et al., 2010) is based on a rather simple model whose performance is not carefully analysed from the learning point of view (*e.g.* by comparing the contribution different state-of-the-art algorithms) as we do here.

The problem of automating system evaluation without a gold standard has been addressed also in other NLP areas. For instance, (Louis and Nenkova, 2013) recently addressed the assessment of machine-generated summaries without model summaries. The strongest parallelism with our work, however, can be found in the Machine Translation (MT) evaluation field, where the goal of bypassing the need of manually-created reference translations has motivated a large body of research.[4] Quality estimation for MT and ASR have a number of commonalities. First, they both deal with a "source" (respectively a sentence in a language *L* and an acoustic utterance) and an "hypothesis" whose quality has to be estimated without references (respectively a translation in a language *L1* and an automatic transcription of the audio signal). Second, they can be addressed at various granularities. Indeed, ASR output quality estimation is similar to its MT counterpart where research focused on quality predictions at word level (Ueffing and Ney, 2007; Bach et al., 2011), sentence level (Specia et al., 2009; Mehdad et al., 2012) and document level (Soricut and Echihabi, 2010). Third, both tasks are suitable for supervised machine learning methods, either for classification (Blatz et al., 2003; Quirk, 2004) or for regression (Specia et al., 2010; Specia, 2011). Finally, both tasks motivate efforts in designing features capable to capture the difficulty to process the source, the plausibility of the output hypothesis and (but not necessarily) the confidence of the decoding process (Felice, 2012; Rubino et al., 2013b).

3 Approach

We approach the automatic estimation of ASR output quality as a supervised regression problem. Given a training set of (*signal*, *transcription*, *WER*) instances, the task is to predict the WER of each instance in a test set of unseen (*signal*, *transcription*) pairs.

Features. As shown in Table 1, the features used in our experiments (68 in total) can be categorized in four main groups. The first group (*ASR features*) includes several glass-box features proposed in previous literature on ASR confidence estimation (Litman et al., 2000; Gabsdil and Lemon, 2004; Goldwater et al., 2010; Higgins et al., 2011). These features are suitable only for the ideal situation in which information about systems' internal decoding strategies is available (as in the experiments discussed in §4.1). We use them as a term of comparison to evaluate the usefulness of the other three groups (*signal*, *hybrid* and *textual*), which belong to the black-box type. These features, which are totally uninformed about the decoding process, have wider applicability to the system-independent ASR quality estimation tasks that represent our target scenario (see Sections 4.2 and 4.3). More in detail:

- **ASR features** aim to capture the confidence of the speech recognizer and the reliability of the whole decoding process. In our experiments, as we do not have access to decoders of other systems, they are computed only for the ASR system developed in our labs (Falavigna et al., 2013). These features

[4] For a complete overview of the current approaches to MT quality estimation we refer the reader to the WMT12 and WMT13 shared task reports (Callison-Burch et al., 2012; Bojar et al., 2013).

are extracted both from word graphs (WGs) and *n*-best lists (n=100). In Table 1 "*Total probability*" is the weighted sum of log Language Model (LM) and log Acoustic Model (AM) probabilities. LM probability is computed with a *4*-gram backoff LM, trained over about 5 billion words using the IRSTLM toolkit (Federico et al., 2008) and the modified shift-beta smoothing method. AM probability is computed using a set of tied-state triphone Hidden Markov Models having, as output state density, a mixture of Gaussian probability densities with diagonal covariance matrices. "*Mean probability*" is obtained dividing the total probability by the number of hypothesized ASR output items (words + silences). Confidence scores are computed averaging time posterior word probabilities (Evermann and Woodland, 2000). "*Proportion of low confidence words*" is the fraction of words having confidence values ≤ 0.5. The remaining ASR features are directly extracted from word graphs and *n*-best lists scores.

- **Signal features** aim to capture the difficulty to transcribe a given input looking at the signal as a whole. They are computed from raw vectors extracted through frame analysis (we employ *20ms* analysis window and *10ms* analysis step). For each analysed window, 12 Mel Frequency Cepstral Coefficients (MFCCs) are evaluated plus log energy. Then, for each given segment, minimum, maximum and mean values of raw energy, as well as the mean MFCCs values and total segment duration, are computed to form the signal feature vector.

- **Hybrid features** provide a more fine-grained way to capture the difficulty of transcribing the signal. This is done by considering information about word and silence/noise regions, as well as their respective duration. These features are computed after having performed forced alignment between the input audio signal and the corresponding automatic hypotheses. Forced alignment is carried out with our ASR system (Falavigna et al., 2013), in order to detect audio segments related to words, hesitations and silences in the hypothesis. Pitch features have been computed with the Praat software tool (Boersma and Weenink, 2005).

- **Textual features** aim to capture the plausibility (*i.e.* the fluency) of an output transcription. To this aim, we consider surface information (such as the number of words and the percentage of numbers/content-words/nouns/verbs in the hypothesis) as well as information about LM perplexity and probability of the hypothesis (both at the level of words and parts of speech)[5].

Feature selection is performed throughout all our experiments to maximize results and, at the same time, analyse the contribution of the proposed features. To this aim, we use Randomized Lasso, or stability selection (Meinshausen and Bühlmann, 2010), which re-samples the training data several times and fits a Lasso regression model on each sample. Features that appear in a given number of samples are considered more informative for the task at hand, and hence retained (those marked in bold in Table 1 are the most informative ones based on the experiments described in Sections 4.2 and 4.3).

Learning algorithms. To build our regression models we experimented with two non-parametric learning approaches: Support Vector Machines (SVMs) (Shawe-Taylor and Cristianini, 2004) and Extremely Randomized Trees (XT) (Geurts et al., 2006). **SVMs** are non-parametric deterministic algorithms that have been widely used in several fields, in particular in NLP where they are the state-of-the-art for various tasks. **Extra-Trees** are a tree-based ensemble method for supervised classification and regression that were also successfully used for MT quality estimation (de Souza et al., 2013; de Souza et al., 2014a). In XTs each tree can be parametrized differently. When a tree is built, the node splitting step is done at random by picking the best split among a random subset of the input features. The results of the individual trees are combined by averaging their predictions. Hyper-parameter optimization of the SVM (with Radial basis function kernel – RBF) and XT models was performed using randomized search (Bergstra and Bengio, 2012). We used both learning methods as implemented in the Scikit-learn package (Pedregosa et al., 2011).

[5]The PoS LM has been obtained by processing with the TreeTagger (Schmid, 1995) the same data used for the word LM.

[6]Hesitations, such as *"uhm"*, *"eh"* and *"ah"* are found through matches with a predefined list. Consecutive repeated words in the same utterance are also considered as hesitations.

ASR (16)	Total probability of ASR output ($w \cdot logP_{LM} + logP_{AM}$), mean probability, total acoustic probability, mean acoustic probability, mean confidence score, Std of confidence scores, confidence scores per second, proportion of low-confidence words, WG node density, WG transition density, Mean/Std/Min n-best probability, Mean/Std/Min n-best acoustic probability.
Signal (16)	Total segment duration (sec), Mean/Min/**Max raw energy (dB)**, mean MFCC[**1**, **2**, 3, 4, **5**, 6, 7, **8**, 9, 10, 11,**12**].
Hybrid (26)	SNR (dB), mean noise energy (dB), Mean/Min/Max word energy (dB), Min/Max noise energy (dB), (max word - min noise) energy (dB), **# silences**, **ratio of silences and words**, # words per second, **# silences per second**, **total duration of words (sec)**, total duration of silences (sec), mean duration of words (sec), mean duration of silences (sec), ratio of (tot duration silences) and (tot duration words), Std of word duration (sec), Std of silence duration (sec), **(tot duration words) - (tot duration silences)**, Mean/**Std**/Min./**Max. pitch (Hz)**, **# hesitations**,[6] frequency of hesitations.
Textual (10)	Number of words, **LM log probability of the hypothesis**, LM log probability of POS of the hypothesis, **LM log perplexity of POS of the hypothesis**, Perplexity of the hypothesis, % of numbers in the hypothesis, % of tokens in the hypothesis which do not contain only a-z, **% of content words in the hypothesis**, % of nouns in the hypothesis, % of verbs in the hypothesis

Table 1: Full list of the 68 features used in our experiments, divided into four groups. The most predictive black-box features (resulting from feature selection in the §4.3 experiments) are marked in bold.

4 Experiments

To evaluate our approach we carried out three sets of experiments. In each set our feature groups are analysed: *i)* with the two learning algorithms, *ii)* in combination/isolation, *iii)* with/without feature selection. The three sets differ in terms of the difficulty of the quality estimation task from the learning point of view. To experiment with situations of increasing complexity, we alternate conditions in which all the features (glass-box and black-box) can be used, training and test sets are non-/homogeneous, the quality estimator is trained on transcriptions generated by the same/different ASR systems.

Data. The data used in the experiments consists of the audio recordings delivered for the IWSLT 2013 evaluation campaign (Cettolo et al., 2013). One of the tasks of IWSLT 2013 is the automatic transcription of English TED talks, a global set of conferences whose audio/video recordings are publicly available. The main challenges for ASR in these talks include: the large variability of topics (hence a large, unconstrained vocabulary), the presence of non-native speakers and a rather informal speaking style. Each IWSLT participant submitted one primary ASR output run for each of the talks included in the test set plus some optional contrastive ASR outputs. In addition, participants sent submissions for the ASR tracks delivered for the 2012 evaluation campaign. Our experiments have been carried out on the primary submissions, sent by 8 participants, related to the 2012 (consisting in 11 different talks) and 2013 (28 different talks) test sets. The 2012 test set has a total duration of around *1h45sec*, it contains *1,118* reference sentences and *18,613* running words. On such dataset, participants' primary submissions achieved a mean utterance WER ranging from 10.5% to 18.4% (in this work a WER score is computed for each reference sentence, and mean utterance WER represents the average of sentence WERs). The 2013 test set has a total duration of around *3h55sec*, it contains *2,238* reference sentences and *41,545* running words. On this dataset, primary participants' submissions achieve a mean utterance WER ranging from 15.9% to 30.8%.

In our experiments, we always use *1,118* utterances for training the regressor and *1,120* for testing. To this aim, the IWSLT 2013 data is randomly sampled three times in training and test sets of such dimensions. While for the 2012 test set manual utterance segmentation has been provided by the organizers, for the 2013 data the participants had to employ their own automatic segmentation systems before decoding the audio tracks (thus resulting in a different number of ASR sentence hypotheses for each team). Hence,

to ensure that each participant has the same number of ASR sentence hypotheses, an alignment with the reference manual segmentation has been performed in our experiments.

Evaluation. Our evaluation is carried out in terms of Mean Absolute Error (MAE), a standard metric for regression problems. The MAE is the average of the absolute errors $e_i = |f_i - y_i|$, where f_i is the prediction of the model and y_i is the actual WER for the i^{th} test instance. WER is calculated with the NIST SCLITE Scoring Package.[7] As it is a measure of error, lower MAE scores indicate that our predictions are closer to the real WER calculated for each test instance against the reference transcripts. For each experiment, we report the mean and the standard deviation of the MAE achieved by the best performing QE models on the IWSLT 2013 test sets.

Baseline. Besides measuring performance in terms of global MAE, each model is compared against a common baseline for regression tasks. This baseline, which is particularly relevant in settings featuring different data distributions between training and test sets, is calculated by labelling each test instance with the mean WER score calculated on the training set. Previous works, also in MT quality estimation, demonstrated that its results can be particularly hard to beat (Rubino et al., 2013a).

4.1 Experiment 1

In the first set of experiments we consider the easiest situation from the learning perspective. In this setting we predict the WER of transcriptions produced by our ASR system (denoted by *X*), whose inner workings are known (thus enabling the use of glass-box features). To investigate the relation between prediction accuracy and the degree of homogeneity of training and test data, we experiment both with similar datasets (disjoint training and test sampled from IWSLT13) and different datasets (IWSLT12 for training and samples from IWSLT13 for test). Results are reported in Table 2, where the notation "*LetterYear - LetterYear*" indicates the systems and the datasets used for training and test (respectively our system *X*, and data from IWSLT*12* and/or IWSLT*13*).

Train - Test	ALL (glass-box + BB_COMB)		ASR (glass-box)		BB_COMB (Signal+Hybrid+Textual)		Baseline
X13 - X13	**11.56±0.29**	SVR	12.11±0.29	XT	15.17±0.06	XT	19.84±0.06
X12 - X13	**12.61±0.13**	XT	13.78±0.16	XT	16.78±0.18	XT	19.06±0.12

Train - Test	Signal		Hybrid		Textual		Baseline
X13 - X13	16.42±0.1	XT	17.61±0.12	XT	17.42±0.15	SVR	19.84±0.06
X12 - X13	18.85±0.09†	XT	18.39±0.22	XT	17.58±0.15	XT	19.06±0.12

Table 2: MAE results using the same system on different datasets, with and without glass-box features.

As can be seen from the table, the two models using ALL the features achieve the largest improvements over the strong baseline used for comparison (up to *8.2* MAE points in the *X13 - X13* setting). This is not surprising if we consider the high predictive power of ASR (glass-box) features that, when used in isolation, lead to a considerably lower MAE with respect to the other three groups. However, it's worth observing that also the combination of only the black-box features (BB_COMB) allows the QE predictors to significantly outperform the baseline (up to *4.67* MAE points in *X13 - X13*). Such improvements come from the joint contribution of each of the three groups, which achieve good results also in isolation. Indeed, except in one case where the gain over the baseline is not significant[8] (*X12 - X13* with the Signal features), their MAE reduction ranges between *0.67* (*X12 - X13* Hybrid) and *3.42* MAE points (*X13 - X13* Signal). The good prediction capability of the black-box features is also shown by the fact that, when combined with the glass-box features, they lead to improvements between *0.55* and *1.17* MAE points over the ASR features alone. Considering the privileged condition of the (system-informed) glass-box features, this is a remarkable result that suggests some complementarity between the two groups.

In general, our supervised approach is sensitive to the similarity between training and test. This is evidenced by higher MAE results when non-homogeneous datasets (*i.e. X12 - X13*) are processed. In

[7] http://www1.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm

[8] Statistical significance is measured by considering the overlap of confidence intervals defined by the standard deviation range around the mean. In our tables, the results marked with the "†" symbol are not significantly better than the baseline.

terms of algorithms, XT generally performs better than SVR, in particular when the QE model is trained and tested on non-homogeneous data. This can be explained by their higher generalization capability due to variance reductions as explained in (Hastie et al., 2009, Chapter 15).

4.2 Experiment 2

In this set of experiments we consider a situation of intermediate difficulty from the learning perspective. Our objective is to evaluate, on homogeneous datasets (sampled from IWSLT13), the output of ASR systems whose inner workings are not known (hence only black-box features can be used). To make our analysis more complete, we also evaluate the performance of models trained on a given ASR system to predict the WER of hypotheses produced by a different one. This situation is closer to application scenarios in which the evaluated ASR system is unknown and different from the one used to train the quality estimator. Two systems with very different performance are considered for this purpose: the best and the worst according to the official IWSLT 2013 ranking (respectively denoted by *A* and *Z*).

Train - Test	BB_COMB		Signal		Hybrid		Textual		Baseline
A13 - A13	**11.18±0.22**	SVR	11.91±0.23	SVR	12.76±0.18	SVR	12.57±0.13	SVR	14.35±0.1
Z13 - A13	**16.01±0.23**	SVR	18.04±0.22	SVR	17.24±0.22	SVR	18.01±0.2	XT	21.58±0.15
Z13 - Z13	**15.52±0.6**	XT	16.94±0.41	XT	17.04±0.56	SVR	17.84±0.4	XT	19.65±0.43
A13 - Z13	**17.36±0.43**	XT	18.7±0.53	XT	18.21±0.45	XT	19.38±0.45	XT	21.03±0.51

Table 3: MAE results using different systems on the same dataset, without glass-box features.

The results reported in Table 3 confirm that: *i)* the combination of black-box features (BB_COMB) always leads to the best QE models, which significantly outperform the baseline, *ii)* the same holds also when each single group is used in isolation, *iii)* with less homogeneous training and test data, XT performs generally better than SVR.

In addition, it's worth noting that when a QE model is trained and tested on data transcribed by the same ASR system the results are significantly better (the MAE is always about *1.0 - 6.0* points lower). Indeed, as also shown by the same behaviour of our baseline, this condition is simpler and more suitable for supervised learning methods. This depends on the fact that each ASR system has its own coherent behaviour, which results in transcriptions with similar characteristics that supervised models are able to learn (*e.g.* recurring errors, similar WER distributions). In contrast, when training and test data are produced by different ASR systems, supervised learning becomes more difficult and the output predictions less reliable. Each feature group is affected by this situation, but it is interesting to note that the Hybrid features are more robust than the other two groups to less homogeneous datasets. This can be explained by the fact that they are extracted after applying forced alignment by means of a third system, which is likely to normalise and reduce the difference between training and test data. Overall, also in this more complex scenario where the glass-box features cannot be used, our results demonstrate a good prediction capability of the QE models, which are still able to beat a strong baseline.

4.3 Experiment 3

In the third set of experiments we consider the hardest case from the learning point of view. In this setting the evaluated ASR systems are unknown and training/test data are non homogeneous (*i.e.* training from IWSLT12, test from samples of IWSLT13). Results are reported in Table 4.

Train - Test	BB_COMB		Signal		Hybrid		Textual		Baseline
A12 - A13	**12.81±0.08**	XT	13.57±0.13†	XT	12.85±0.1	XT	13.25±0.23†	XT	13.65±0.17
Z12 - A13	14.78±0.1	SVR	15.66±0.09†	XT	**13.56±0.09**	SVR	13.63±0.24	SVR	15.51±0.35
Z12 - Z13	**17.16±0.4**	XT	19.34±0.32†	XT	17.68±0.3	XT	19.59±0.11†	XT	19.98±0.29
A12 - Z13	**19.83±0.23**	XT	21.85±0.2	XT	20.68±0.13	XT	22.62±0.08	XT	23.04±0.18

Table 4: MAE results using different systems on different dataset, without glass-box features.

Also in the most challenging scenario our results substantially confirm the previous findings. Indeed, except in one case (*Z12 - A13*), the following observations still hold: *i)* when used in combination, the

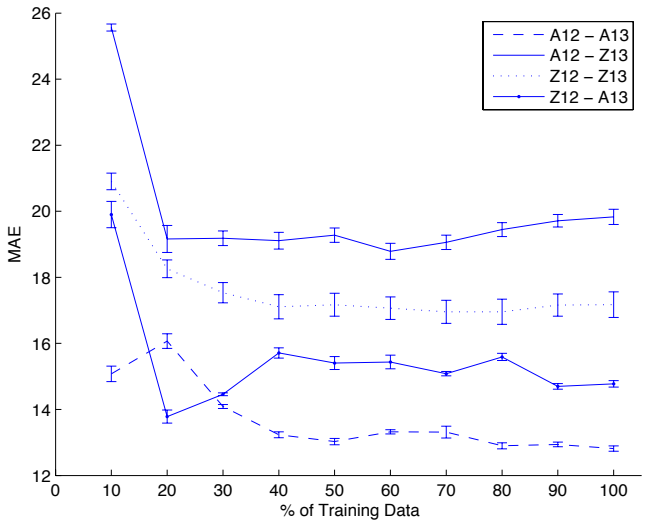


Figure 1: Learning curves for the best systems of "Experiment 3" (using BB_COMB features).

black-box features (BB_COMB) lead to the best QE models, which significantly outperform the baseline, *ii)* this holds also when each single group is used in isolation (although not significantly in 5 out of 12 settings), *iii)* with less homogeneous training and test data, XT performs generally better than SVR.

Unsurprisingly, as also observed in the previous set of experiments, the low homogeneity of training and test data has an impact on the accuracy of the predictions. The effect of training and testing on less homogeneous data produced by different systems is now clearly visible. Except for the more robust Hybrid features, which in the *Z12 - A13* setting produce the best model, the results obtained with the two other groups decreased to the point that their improvement over the baseline is often not significant. Nevertheless, even under the challenging conditions posed by this realistic and application-oriented scenario, reference-free and system-agnostic ASR evaluation remains a feasible task.

5 Feature Analysis and Learning Curves

In order to gain additional insights about the effectiveness of our method, we performed a further analysis of the "Experiment 3" results. In such challenging scenario, the most interesting from the application perspective, we first identified the most predictive features among those in the BB_COMB set. To this aim, we collected the features that are always chosen by the feature selection algorithm proposed in §3. The resulting list contains features from all the three black-box groups (marked in bold in Table 1). This confirms their complementarity in predicting the quality of a transcribed utterance.

In the same setting, we also investigated the relation between the amount of data used to train our models and the accuracy of their predictions. To this aim, we measured performance variations when the same models (*i.e.* those obtained with the BB_COMB set) are trained on different amounts of data. For each training set, nine subsets were created (with 10%, 20%,..., 90% of the data) by sub-sampling sentences from a uniform distribution. The process was iterated 5 times. Each subset was used to build the relative QE regressor, which was then evaluated on our test sets. Figure 1 shows the resulting learning curves (each point is the average result of the 5 runs on each test set; the error bars show $\pm$1std). As can be seen from all the curves, after an initial fluctuation of the MAE, performance results with 40% of the training data are comparable with those obtained using the whole training set. Moreover, it's worth remarking that in three out of four cases the models trained with such amount of data already outperform the baseline (for *Z12 - A13* the MAE is only 0.01 point higher). This suggests that reference-free, system-independent models for ASR quality estimation are able to provide informative predictions even with a limited amount ($\sim$400 manual transcripts) of training instances.

6 Conclusion

We investigated the problem of automatically predicting the word error rate of an automatically-transcribed utterance in a large vocabulary continuous speech recognition setting. In such scenario, we proposed a supervised regression approach that bypasses the need of manual reference transcriptions and does not necessarily depend on information about system's confidence (*first contribution* of the paper). Then, by evaluating models obtained with different state-of-the-art learning algorithms, we showed that our automatic predictions outperform a strong baseline and closely approximate the WER scores calculated over reference transcripts (*second contribution*). Different feature groups have been proposed and their contribution has been analysed in a range of testing conditions of increasing difficulty (*third contribution*). This made possible to isolate informative features that significantly contribute to the performance of our quality estimation models, and to get useful insights about the potential of our approach when different sources of information (glass-box, black box features) are available. Finally, analysing the relation between prediction performance and the size of the training set, we showed that the results obtained with 40% of the data are already comparable to our best MAE (*fourth contribution*).

Our analysis revealed a dependency between the performance of the quality estimation models and the degree of homogeneity between training and test data. This aspect is particularly relevant from the application perspective since in real working conditions the availability of *large amounts* of *representative* training instances is far from being guaranteed. In quality estimation for machine translation (a task featuring strong similarities with ours), these issues have recently motivated studies on domain adaptation and online learning techniques (de Souza et al., 2014b; Turchi et al., 2014). This suggests, as a first direction for future work, the investigation of approaches capable to better exploit the available training data and mitigate the impact of large differences between training and test instances.

Acknowledgements

This work has been partially funded by the European project EU-BRIDGE (FP7-287658) and by the Autonomous Province of Trento, Italy, under the project Wikivoice (L.P. 6/1999).

References

Nguyen Bach, Fei Huang, and Yaser Al-Onaizan. 2011. Goodness: a Method for Measuring Machine Translation Confidence. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 211–219. The Association for Computer Linguistics.

James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, 13:281–305.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2003. Confidence Estimation for Machine Translation. Summer workshop final report, JHU/CLSP.

Paul Boersma and David Weenink. 2005. Praat: Doing Phonetics by Computer (Version 4.3.01). *Retrieved from http://www.praat.org/.*

Ondrej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Eighth Workshop on Statistical Machine Translation*, WMT-2013, pages 1–44, Sofia, Bulgaria.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation (WMT'12)*, pages 10–51, Montréal, Canada.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2013. Report on the 10th IWSLT Evaluation Campaign. *Proceedings of the International Workshop on Spoken Language Translation, Heidelberg, Germany*.

José G. C. de Souza, Christian Buck, Marco Turchi, and Matteo Negri. 2013. FBK-UEdin participation to the WMT13 quality estimation shared task. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 352–358, Sofia, Bulgaria, August. Association for Computational Linguistics.

José G. C. de Souza, Jesús González-Rubio, Christian Buck, Marco Turchi, and Matteo Negri. 2014a. FBK-UPV-UEdin participation in the WMT14 Quality Estimation shared-task. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, Baltimore, MD, USA, June.

José G. C. de Souza, Marco Turchi, and Matteo Negri. 2014b. Predicting Machine Translation Quality Estimation Across Domains. In *Proceedings of the 25th International Conference on Computational Linguistics*, COLING'14, Dublin, Ireland.

Gunnar Evermann and Philip C. Woodland. 2000. Large Vocabulary Decoding and Confidence Estimation Using Word Posterior Probabilities. In *Proc. of ICASSP*, pages 2366–2369, Istanbul, Turkey, June.

Daniele Falavigna, Roberto Gretter, Fabio Brugnara, Diego Giuliani, and Romain Serizel. 2013. FBK@IWSLT 2013 - ASR Tracks. In *Proceedings of the IWSLT 2013 workshop*, Heidelberg, Germany.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. IRSTLM: an Open Source Toolkit for Handling Large Scale Language Models. pages 1618–1621, Brisbane, Australia, September.

Marcello Federico, Luisa Bentivogli, Michael Paul, and Sebastian Stüker. 2011. Overview of the IWSLT 2011 Evaluation Campaign. In *International Workshop on Spoken Language Translation*, pages 11–27.

Marcello Federico, Mauro Cettolo, Luisa Bentivogli, Michael Paul, and Sebastian Stüker. 2012. Overview of the IWSLT 2012 Evaluation Campaign. In *Proc. of the International Workshop on Spoken Language Translation*, Hong Kong, HK, December.

Mariano Felice. 2012. Linguistic Indicators for Quality Estimation of Machine Translations. Master's thesis, University of Wolverhampton, UK.

Malte Gabsdil and Oliver Lemon. 2004. Combining acoustic and pragmatic features to predict recognition performance in spoken dialogue systems. pages 344–351.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Mach. Learn.*, 63(1):3–42, April.

Sharon Goldwater, Dan Jurafsky, and Christopher Manning. 2010. Which words are hard to recognize? Prosodic, lexical, and disfluency factors that increase speech recognition error rates. 52(3):181–200.

Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. 2009. *The elements of statistical learning*, volume 2. Springer.

Derrick Higgins, Xiaoming Xi, Klaus Zechner, and David Williamson. 2011. A three-stage approach for automated scoring of spontaneous responses. (25):282–306.

Diane J. Litman, Julia B. Hirschberg, and Marc Swerts. 2000. Predicting Automatic Speech Recognition Performance Using Prosodic Cues. In *Proceedings of NAACL*, pages 218–225.

Annie Louis and Ani Nenkova. 2013. Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, 39(2):267–300.

Yashar Mehdad, Matteo Negri, and Marcello Federico. 2012. Match without a Referee: Evaluating MT Adequacy without Reference Translations. In *Proceedings of the Machine Translation Workshop (WMT2012)*.

Nicolai Meinshausen and Peter Bühlmann. 2010. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Christopher B. Quirk. 2004. Training a Sentence-Level Machine Translation Confidence Measure. In *Proceedings of LREC 2004*.

Raphael Rubino, José GC de Souza, Jennifer Foster, and Lucia Specia. 2013a. Topic Models for Translation Quality Estimation for Gisting Purposes. In *Proceedings of the Machine Translation Summit XIV*.

Raphael Rubino, Joachim Wagner, Jennifer Foster, Johann Roturier, Rasoul Samad Zadeh Kaljahi, and Fred Hollowood. 2013b. DCU-Symantec at the WMT 2013 Quality Estimation Shared Task. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 392–397.

Alberto Sanchis, Alfons Juan, and Enrique Vidal. 2012. A Word-Based Naive Bayes Classifier for Confidence Estimation in Speech Recognition. 20(12):565–574.

Helmut Schmid. 1995. Improvements in Part-of-Speech Tagging with an Application to German. In *Proceedings of the ACL SIGDAT-Workshop*, pages 47–50, Dublin, Ireland.

Matthew Stephen Seigel. 2013. *Condence Estimation for Automatic Speech Recognition Hypotheses*. University of Cambridge. PhD Thesis.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel methods for pattern analysis*. Cambridge university press.

Radu Soricut and Abdessamad Echihabi. 2010. TrustRank: Inducing Trust in Automatic Translations via Ranking. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 612–621, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lucia Specia, Nicola Cancedda, Marc Dymetman, Marco Turchi, and Nello Cristianini. 2009. Estimating the Sentence-Level Quality of Machine Translation Systems. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT'09)*, pages 28–35, Barcelona, Spain.

Lucia Specia, Dhwaj Raj, and Marco Turchi. 2010. Machine Translation Evaluation versus Quality Estimation. *Machine translation*, 24(1):39–50.

Lucia Specia. 2011. Exploiting Objective Annotations for Measuring Translation Post-editing Effort. *Proceedings of the 15th Conference of the European Association for Machine Translation*, pages 73–80.

Rafic Antoon Sukkar and Chin-Hui Lee. 1996. Vocabulary Independent Discriminative Utterance Verification for Nonkeyword Rejection in Subword Based Speech Recognition. 6(6):420–429.

Marco Turchi, Antonios Anastasopoulos, José G. C. de Souza, and Matteo Negri. 2014. Adaptive Quality Estimation for Machine Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL'14, Baltimore, MD, USA. Association for Computational Linguistics.

Nicola Ueffing and Hermann Ney. 2007. Word-Level Confidence Estimation for Machine Translation. *Comput. Linguist.*, 33(1):9–40, March.

Frank Wessel, Ralf Schlüter, Klaus Macherey, and Hermann Ney. 2001. Confidence Measures for Large Vocabulary Continuous Speech Recognition. 9(3):288–298.

Su-Youn Yoon, Lei Chen, and Klaus Zechner. 2010. Predicting word accuracy for the automatic speech recognition of non-native speech. In *Proc. of INTERSPEECH*, pages 773–776, Makuhari,Chiba, Japan.

A Generic Anaphora Resolution Engine for Indian Languages

Sobha Lalitha Devi
AU-KBC Research Centre
MIT Campus of Anna University, Chennai, India
sobha@au-kbc.org

Vijay Sundar Ram
AU-KBC Research Centre
MIT Campus of Anna University, Chennai, India
sundar@au-kbc.org

Pattabhi RK Rao
AU-KBC Research Centre
MIT Campus of Anna University, Chennai, India
pattabhi@au-kbc.org

Abstract

In this paper, we present a generic anaphora engine for Indian languages, which are mostly resource-poor languages. We have analysed the similarities and variations between pronouns and their agreement with antecedents in Indian languages. The generic algorithm developed uses the morphological richness of Indian languages. The machine learning approach uses the features which can handle major Indian languages. We have tested the system with Indo-Aryan and Dravidian languages namely Bengali, Hindi and Tamil. The results are encouraging.

1 Introduction

Natural language has different types of anaphoric expressions and these expressions bring elegance and make the natural language text interesting to read. Anaphoric expression in a discourse refers to another item in a discourse. The task of resolving anaphors with its referent, antecedent is called as anaphora resolution. Anaphora resolution is required in most of the NLP applications to achieve required performance. The importance of anaphora resolution in various tasks is demonstrated by researchers by integrating anaphora resolution with answer extraction system, automatic summarization system, relation extraction system, document similarity identifier etc.

Most of the anaphora resolution systems are developed for particular languages. The researchers have analyzed anaphors across languages at various levels such as syntactic, semantic, discourse, structured, and unstructured features. But there are very few attempts for language independent approaches. In this paper, we present a generic anaphora resolution engine for Indian languages. We have come up with a language independent engine, which takes shallow parsed text as input. The morphological richness of Indian languages is tapped to come up with a language independent anaphora resolution engine.

Early works in anaphora resolution by Hobbs (1978), Carbonell and Brown (1988), Rich and LuperFoy (1988) etc. were mentioned as knowledge intensive approach, where syntactic, semantic information, world knowledge and case frames were used. Centering theory, a discourse based approach for anaphora resolution was presented by Grosz (1977), Joshi and Kuhn (1979), Joshi and Weinstein (1981), Strube and Hahn (1999). Salience feature based approaches were presented by Lappin and Leass (1994), Kennedy Boguraev (1996) and Sobha et al., (2000). Indicator based resolution methods were presented by Mitkov (1997, 1998). One of the early works using machine learning technique was Dagan Itai's (1990) unsupervised approach based on co-occurrence words. With the use of machine learning techniques researchers work on anaphora resolution and noun phrase anaphora resolution simultaneously. The other machine learning approaches for anaphora resolution were the following. Aone and Bennett (1995), McCarty and Lahnert (1995), Soon et al., (2001), Ng and Cardia (2002) had used decision tree based classifier. Daelman and Van de Bosh (2005), Hendrickx et al., (2008), Recasen (2009) had used TiMBL, a memory based learning approach. Anaphora resolution using CRFs was presented by McCallum and Wellner (2003) for English, Li et al., (2008) for Chinese and Sobha

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1824–1833, Dublin, Ireland, August 23-29 2014.

et al., (2011, 2013) for English and Tamil. Expectation Maximization (EM) was used for anaphora resolution by Charniak and Elsner (2009). Wich et al., (2012) demonstrated a coreference resolution in a large scale using discriminative hierarchical model.

In Indian languages anaphora resolution engines are demonstrated only in few languages such as Hindi, Bengali, Tamil, and Malayalam. Most of the Indian languages do not have parser and other sophisticated pre-processing tools. The earliest work in Indian language, Vasisth was a rule based multilingual anaphora resolution platform by Sobha and Patnaik (2000, 2002), where the authors had exploited the morphological richness of Malayalam and Hindi. Prasad and Strube (2000), Uppalapu et al., (2009) and Dekwale et al., (2013) had presented different approaches using Centering theory for Hindi. Dutta et al (2008) had presented a Hindi anaphora resolution system using Hobbs' algorithm. Murthy et al., (2007) had presented a comparison on Tamil anaphora resolution using multi-linear regression and salience factor based approach. Sobha et al., (2007) presented a salience factor based with limited shallow parsing of text. Akilandeswari et al., (2013) used CRFs for resolution of third person pronoun and Akilandeswari et al., (2012) presented a work on resolution of 'atu', third person neuter pronoun in Tamil. Balaji et al., (2012) presented resolution using two stage bootstrapping approach. The author had used UNL representation. Ram et al., (2013) used Tree CRFs for anaphora resolution for Tamil with features from dependency parsed text.

One of the earliest multilingual anaphora resolution systems was presented by Aone and Mckee (1993), where the authors had used Global discourse world which contained syntactic, semantic, rhetorical and other information. They demonstrated the system for English, Spanish and Japanese. Mitkov (1998) extended the indicators based approach for other languages and presented it for English, Polish and Arabic. As mentioned earlier, Vasisth was the only multilingual attempt for Indian language anaphora. SemEval-2010 Task 1: Coreference resolution in Multiple Language, a tool contest accelerated the research in multilingual anaphora resolution. The contest had six languages, namely, English, German, Italian, Dutch, Spanish and Catalan. There were six participants, among them only two participants presented results for all six languages. The systems presented in the contest were RelaxCor, Corry, SUCRE, BART, TANL-1 and UBIU (Recasens et al., 2010). The multilingual anaphora resolution in Indian languages was re-initiated by Anaphora Resolution in Indian languages, the tool contest conducted as a part of ICON 2011 (Sobha et al., 2011). The contest had three languages, namely, Hindi, Bengali and Tamil. There were four participants and all the four submitted results for Bengali. Tamil and Hindi had two participants. This task boosted the anaphora resolution work in Bengali. Senapati et al., (2013) presented a work on Bengali anaphora resolution by customizing GuiTAR and BART tool was customized for Bengali by Sikdar et al., (2013). In all the above published multilingual systems, there was a language dependent module plugged in. In the present work, we have tried to come up with an approach without using language specific modules. We have developed a generic anaphora engine as the system is designed to work for different languages. We have overcome the agreement problem with the PNG information obtained from in-depth morphological analysis and PNG agreement heuristic rules. These rules are capable of filtering the possible candidate antecedents for an anaphor (pronouns) using the PNG information across languages.

The rest of the paper is organised as follows: In the following section we have described nature of Indian languages and variation in antecedent-anaphor agreement in Indian languages. Section 3, we have explained our approach towards generic engine by overcoming the variation in antecedent-anaphor agreements. In section 4, we have presented our experiment and results. And the last is the conclusion section.

2 Characteristics of Indian Language Anaphora

Indian languages are morphologically rich and verb-final languages. These languages have relatively free word order and clausal structures are more fixed order. Indian languages fall under the following broader families of languages, Indo-Aryan, Dravidian and Tibeto-Burman. Indo-Aryan family includes languages such as Hindi, Bengali, Marathi, Punjabi etc, Dravidian includes languages such as Telugu, Kannada, Malayalam, Tamil, Tibeto-Burman includes languages such as Bodo, Manipuri etc. Dravidian languages are highly agglutinated and have rich productive suffixation than the Indo-Aryan languages. Plural marker and case markers get affixed to the nouns and tense markers and Person, Number, Gender (PNG) markers affix with verbs. In certain Indo-Aryan languages such as Hindi, case

markers occur as postpositions following the nouns. These postpositions are handled in the preprocessing stage to occur in the noun morphological analysis. Indian languages vary largely in the distinction of Number (singular/plural) and Gender in pronouns. Few of the Indian languages and their details of Number and Gender Distinction in those languages are presented in table 1.

Language	**Number Distinction (singular/plural)**	**Gender Distinction**
Hindi	Yes	No
Sanskrit	Yes	Yes
Punjabi	Yes	No
Gujarati	Yes	No
Assamese	Yes	No
Bengali	Yes	No distinction for Masculine and Feminine. But there is animate- inanimate distinction.
Oriya	Yes	No
Telugu	Yes	Masculine and others
Kannada	Yes	Yes
Malayalam	Yes	Yes
Tamil	Yes	Yes

Table 1: Variation of Pronouns with respect to Number and Gender

The similarities and variations between languages in the number-gender characteristics of the pronouns is presented in Table 1. In the number characteristics the languages are similar whereas in the gender characteristics there are variations. The information in the table 1 brings forth the challenges in capturing the anaphor-antecedent PNG agreement for a generic anaphora resolution engine.

With example 1, 2 and 3, we have demonstrated the variation in Gender, Number distinction in pronouns in Tamil (Ta), Bengali (Bn) and Hindi (Hi).

Example 1

```
Ta:a) raamum        giithavum      cakotharan-cakothari.
      Ram(N)+inc    Gita(N)+inc    brother   -sister.
      (Ram and Gita are brothers and sisters.)

   b) avan     elzhaam     vakuppu      padikkiraan.
      He(PN)   seventh(N)  standard(N)  study(V)+present+3sm
      (He studies in seventh standard.)

   c) aval     paththaam vakuppu      padikkiraal.
      she(PN)  tenth(N)  standard(N)  study(V)+present+3sf
      (She studies in tenth standard.)
```

Example 2

```
Bn:a) raam o giita bhai-bon.
   b) se shapton shreni te pore.
   c) se doshom shreni te pore.
```

Example 3

```
Hi:a) raam aur giitaa bhaii-bahan hai.
   b) vaha satavIM kakshaa meM paTataa hai.
   c) vaha aaTaviM kakshaa meM paTatii hai.
```

Example 1 has three Tamil sentences. The second and third sentence has pronouns 'avan' *he* and 'aval' *she*, the third person masculine and the third person feminine pronouns respectively. Masculine pronoun in second sentence refers to the masculine noun 'raam' and the feminine pronoun in the third sentence refers to the feminine noun 'Gita' in the first sentence. Here the masculine and feminine pro-

nouns have a clear distinction. The three Tamil sentences in example 1 are translated to Bengali and Hindi. The Bengali translation is presented in example 2 and Hindi translation is presented in example 3. In example 2, the second and third sentence has 'se', the third person pronoun. In the second sentence, the pronoun 'se' refers to the masculine noun 'raam' and 'se' in the third sentence refers to feminine noun 'giitaa' in the first sentence. Here the third person pronoun does not have masculine/feminine distinction. Similarly in example 3, which has the Hindi translation, 'vaha', the third person pronoun does not have gender distinction. In sentence 2 of example 3, 'vaha' refers to the masculine noun and in sentence 3, 'vaha' refers to the feminine noun.

These variations in Number and Gender distinction in pronouns pose challenges in coming up with a generic anaphoric engine. The pronoun and its agreement with its antecedents vary between the languages and to handle the agreement we require a language dependent mapping.

3 Generic Anaphora Resolution Engine

Most of the Indian languages are resource poor languages. The morphological richness of these languages, help in building various high end NLP applications such as machine translation, anaphora resolution etc., with limited shallow parsed information without using sophisticated parsing tools. In this work we have tried to build a generic anaphora resolution engine using shallow parsed text. Similarities between Indian languages, described in the previous section, are tapped to come up with a generic approach for anaphora resolution in Indian languages. The variation in the antecedent-anaphor agreement mentioned in the section above is handled by an in-depth morphological analysis of the text. We have used CRFs, a linear graphical machine learning algorithm to resolve the antecedents.

3.1 Preprocessing of Data

We perform limited shallow parsing on the training and testing data. Both the data are pre-processed with morphological analyzer, Part-of-Speech (POS) tagger, Chunker, Clause boundary identifier and Named Entity Recognizer. Here morphological analysis, Part-of-Speech tagging, Chunking are obligatory. Clause boundary identification and Named Entity Recognition are optional pre-processing tasks. These two tasks add information, which can be used as constraint features in the machine learning approach. In this work, we perform a detailed morphological analysis for a given word. This is explained in the following section. The preprocessing tools available in Indian Language –Indian Language Machine Translation (IL-ILMT) consortium are used.

3.2 Detailed Morphological Analysis

We perform an in-depth morphological analysis for a given word. In the in-depth morphological analysis we analyse both inflectional and derivational morphology. The in-depth morphological analysis gives the suffix (case markers with the nouns, tense-aspect-model with the verbs) and PNG characteristics of the words. These suffix information is used in the syntactic feature and verb suffix feature for the machine learning technique which are described further in section 3.4. The post-position occurring with the nouns, its syntactic association with the noun is identified in morphological processing stage and information is used as syntactic feature.

The morphological analyser identifies the root word, its lexical category, gender, number, person, case (direct/oblige), case markers if the word is a noun and tense markers (vibhakthi as called in Indian traditional grammar) if the word is a verb and the suffixes. Gender information holds information such as 'm' – masculine, 'f' – feminine, 'n' – neuter, 'mf' – can be a masculine or feminine, 'fn' – feminine or neuter as in Telugu and 'any' – can be any gender. Number information can be singular, plural, dual or any. Person information can be 1st person, 2nd person, 3rd person or any. We have explained it further with following example words and its analysed output in table 2.

S.No	Language	Word	Analysis of the Word
1	Ta	jaanukku 'John(N)+dative'	<fs af='jaan,n,m,sg,3,d,ukku,ukku'>
2	Ta	viittil 'house(N)+locative'	<fs af='viitu,n,any,sg,3,d,il,il'>
3	Ta	avanaal 'he(pn)+INS'	<fs af='avan,pn,m,sg,3,d,aal,aal'>
4	Ta	avalukku 'he(pn)+dative'	<fs af='aval,pn,f,sg,3,d,ukku,ukku'>
5	Hi	adhikaarii 'officer (N)'	<fs af='adhikaarii,n,m,sg,3,d,,'>

6	Hi	siitaa ‘sita (N)’	<fs af='siitaa,n,f,sg,3,d,,'>
7	Hi	uskaa ‘he/she/it (pn)’	<fs af='vaha,pn,any,sg,3,d,kaa,kaa'>
8	Hi	ve ‘they (pn)’	<fs af='vaha,pn,any,pl,3,d,,'>
9	Bn	chele ‘boy (N)’	<fs af=’chele,n,m,sg,3,d,,'>
10	Bn	meyze ‘girl (N)’	<fs af=’meyze,n,m,sg,3,d,,'>
11	Bn	se ‘he/she (PN)’	<fs af='se,pn,mf,sg,3,d,,'>

Table 2: Words and in-depth analysis

In the table 2, we have presented nouns and pronouns from Hindi (Hi), Bengali (Bn) and Tamil (Ta) and their in-depth analysis. The first word 'jaanukku' is a masculine singular noun. So the analysis has 'm,sg,3'. The second word 'viittil' is a neuter singular noun and its analysis has 'n,sg,3'. The third word is third person masculine and the fourth word is a feminine pronoun, so the analysis are ‘m,sg,3’ and ‘f,sg,3’ respectively. The words in the Fifth and sixth example are Hindi nouns with masculine and feminine gender respectively. The seventh example is third person singular and eighth example is third person plural word from Hindi. Hindi pronouns do not have gender distinction. The gender, number, person for the two pronouns are 'any,sg,3' and 'any,pl,3' respectively. 'any' in the gender slot shows the pronoun can refer to a noun phrase with any gender including neuter gender. The ninth and tenth example words are Bengali nouns ‘boy’ and ‘girl’, with masculine and feminine gender respectively. The eleventh word is a Bengali third person masculine and feminine pronoun. The gender, number, person in the morphological analysis has 'mf,sg,3'. This pronoun can refer to both masculine and feminine noun in Bengali.

3.3 Data Format

After pre-processing, the data is presented in a column format. The following are the columns information. First column has sentence id, followed by word id, POS tag, chunk tag, in-depth morphological analysis, clause information and Named Entity information. The training data has an additional column having the antecedent-anaphor agreement information.

3.4 Architecture of the Engine

The engine works independent of language. We have used heuristic rule based algorithm to select the candidate noun phrases for a given pronoun and machine learning techniques based approach to filter the exact antecedent noun phrase. As every supervised machine learning approach, this approach also has training and testing phase. The architecture of our approach for training and testing is given in figure1 and 2 respectively.

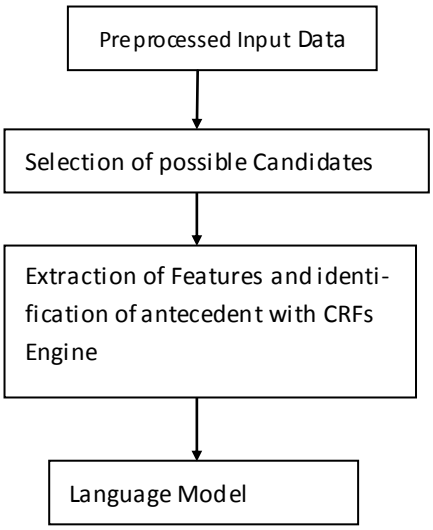


Figure 1: Training phase

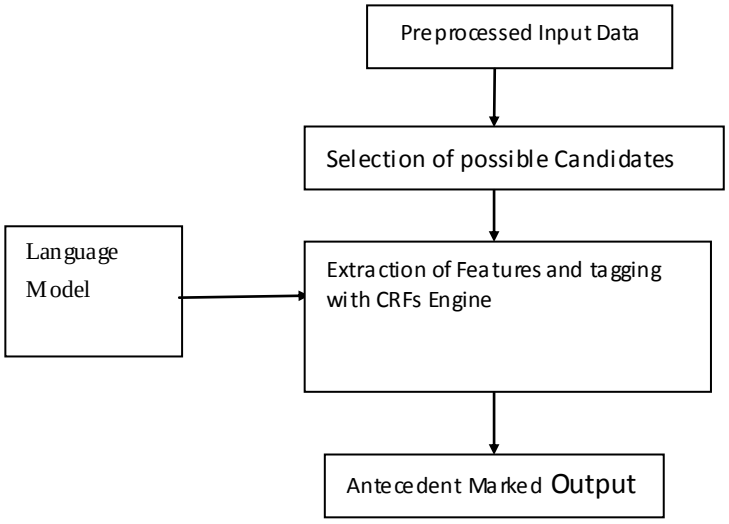


Figure 2: Testing phase

Selection of Candidate Noun Phrases for Antecedent

Both the training and testing phase has selection of the possible candidates. The noun phrases which agree with the pronoun in PNG should be selected as possible candidates for its antecedent. In training phase, the noun phrases which match with PNG of the pronoun and occur in between the anaphor and the antecedent are collected for each pronoun and given for training using the machine learning algorithm. The exact anaphor and antecedent pair forms positive pair and other noun phrases and anaphor form negative pairs for learning. In the testing phase all the noun phrases that match in PNG with the pronouns are collected from the current sentence and four prior sentences. The gender distinction and anaphor-antecedent agreement varies widely among Indian languages. In order to have a language independent engine, these variations have to be dynamically captured and the rules for checking PNG agreement have to be generated. We have used the gender information from the morphological analysis extracted with a set of heuristic rules to capture the variation in PNG agreement. The heuristic rules describe the possible genders that can match with the gender of the pronoun that varies between languages. The heuristic rules are presented below.

1. If the gender of the pronoun is 'm', then the nouns having masculine gender are chosen as candidate antecedents.
2. If the gender of the pronoun is 'f', then the nouns with feminine gender are chosen as candidate antecedents.
3. If the gender of the pronoun is 'n', then the nouns having neuter gender are chosen as candidate antecedents.
4. If the gender of the pronoun is 'mf', then the nouns with gender 'mf', 'm' and 'f' are chosen as candidate antecedents and the nouns with gender 'mf' is given importance.
5. If the gender of the pronoun is 'fn' is the gender of the pronoun, then nouns with 'fn' are chosen as candidate antecedents.
6. If the gender of the pronoun is 'any', then all the nouns are considered for candidate antecedent set and the nouns with gender 'any' is given higher priority.

Once the possible candidates for antecedents for the anaphors are selected, they are given for training/testing using the machine learning technique. Here we have used CRFs, a linear graphical technique to learn and identify the antecedents.

Anaphora Resolver

The core anaphora engine uses CRFs, a machine learning technique. In the training phase the system is provided with annotated data and the features for learning. After the system learns, a model file is generated as output. In the testing phase any unseen text is given for the automatic anaphora resolution. In our approach we have modeled this as a binary classification task. The machine has to classify whether the given candidate antecedent is the real antecedent or not based on the features of the candidate antecedents and the pronoun. The features for learning are extracted from the shallow parsed data. The feature extraction module extracts these features for all possible candidate antecedent and pronoun pairs from the shallow parsed data. The features used for learning are described below. Here we have used the freely available open source CRFs (Kudo, 2005).

Features for Learning

The features required for this task are identified from shallow parsed input sentences. The features for all possible candidate antecedent and pronoun pairs are obtained by preprocessing the input sentences with in-depth morphological analyser, POS tagger, and chunker, clause boundary identifier and Named Entity recognizer, where the last two preprocessing tasks are optional. The features identified can be classified as positional features, syntactic features and constraint features.

a) *Positional Features:* The occurrence of the candidate antecedent is noted. Is it in the same sentence where the pronoun occurs or in the prior sentences? Prior four sentences from the current sentence are considered.

b) *Syntactic Features:*

Syntactic Role: The syntactic role of the candidate noun phrases in the sentence is a key feature. The syntactic role of the noun phrases such as subject, object, indirect object, are obtained from the

case suffix affixed with the noun phrase. We consider Nominative and Dative cases for subject and other cases for object, the position and the other cases for indirect object.

Linguistic Characteristics: POS tag and chunk information of Candidate NP, suffixes affixed with the noun.

c) *Verb Suffixes:* The suffixes which show the gender which gets attached to the verb.

d) *Nature of NP:* Whether the candidate NP (probable antecedent) is Possessive or Existential.

e) *Constraint Features:* The constraint features are obtained from clause boundary and named entities recognized.

The position of the candidate NP with respect to clause boundary such as is candidate NP in current clause or immediate clause or non-immediate clause.

The Named Entity tags associated with the candidate NPs help the learning algorithm to learn constraints that types of NEs that can be its possible antecedents.

f) Combination of the above said features.

The features for learning have been identified based on the characteristics of the pronouns. For example constraint feature (current-clause) and syntactic feature (subject) helps in identifying the antecedent of the reflexives. For relative anaphors the constraint feature (immediate-clause) and syntactic feature (subject) help in identifying the antecedent.

4 Experiment, Results and Discussion

We have tested our approach using the dataset provided in "Anaphora Resolution for Indian Language", a tool contest conducted as a part of ICON 2011. The tool contest had three languages namely Tamil, Hindi and Bengali. The dataset is presented in column format with the following information viz. line index, word index, word, its POS and chunking information, followed by Named Entity information. We have enriched the dataset with in-depth morphological analysis. Table 3 presents the statistics of the ICON 2011 tool contest dataset.

Language	Training Data			Testing Data		
	Bengali	Hindi	Tamil	Bengali	Hindi	Tamil
Total Number of Pronouns	814	835	925	494	507	609
Number of Anaphoric Pronouns	476	557	580	283	344	348
Number of Non-Anaphoric Pronouns	338	278	345	211	163	261

Table 3: Statistics of ICON 2011 Dataset

MUC, B^3, BLANC, CEAF are the common scorers available for coreference resolution, where the coreference chains are being evaluated. Anaphora resolution is generally evaluated with performance measures such as Precision, Recall and F-measure. In this work, we have measured the performance with Precision, Recall and F-Measure and it is presented in table 4.

Example 4

Ta:	**aalluNar**	**rosaiyya**	villavil	kalanthukkoNtaar.	**avar**	athil	uraiyaRRinaal.
	Governor	Rosiah	function+loc	join+past+3SH .	He(gender neutral)	this	gave_lecture

(Governor Rosiah joined the function. He gave the lecture there.)

In above example 4, 'avar' (third honorific singular pronoun) in the second sentence, refers to a masculine noun phrase 'aalluNar rosaiyya' in the previous sentence. The honorific pronoun can also refer to a feminine pronoun. This possibility of referring to both the gender introduces more errors.

In Hindi, most of the pronouns such as "vaha" (he/she/it), "usa" (he/she/it), "unhone" (he/she honorofic) and "khuda" (himself) etc., do not have gender distinction and can be used to refer to antecedents of both feminine and masculine. PNG agreement adds more challenges in anaphora resolution, due to which the system gives more false positives. In our algorithm we were able to reduce the number of false positives and obtained better precision by having positional features and the verb suffixes as learning features. For languages such as Hindi, we observe that there is necessity of having verb

analysis in the text processing component. If we have this information as pre-processed data to the resolution engine, it can reduce ambiguity and improve the anaphora resolution.
As it is seen from the results table we have obtained lesser scores for Bengali. In Bengali third person pronouns such as “ami” (I), “túmi/tui/apni” (you), “se/tini” (he/she), “amra” (we), “tara/tnara” (they), do not have masculine, feminine distinction, but there is animacy distinction. And also the verb has no gender agreement. This adds more challenge to anaphora resolution engine and hence lesser scores than other languages. We identify the animacy feature in the morphological analysis stage for all the languages, but for Bengali language it is not robust and it affects in the anaphora resolution. For such languages the order of NPs and syntactic roles play major role in anaphora resolution. The use of features such as Named Entities helps in improving the resolution of pronouns referring to person and location. The addition of clause boundary information improves resolution by adding structural constraints.

5 Conclusion

We have presented a generic anaphora resolution engine, which can be used for all Indian languages. The three languages, Hindi, Bengali, and Tamil, we have chosen are the most spoken languages belonging to two major language families of India, namely belonging to Indo-Aryan and Dravidian families respectively. Though the Indian languages have similarities, they vary in Person, Number, Gender distinction, which pose a challenge in building language independent engine. The engine is language independent as it uses the information from the in-depth morphological analysis. It is specifically designed in such a way that it is scalable and allows plug-n-play architecture. The core anaphora resolution engine uses CRFs a machine learning technique. This uses feature based learning and we have provided syntactic and positional based features obtained from in-depth morphological analysis. We have obtained encouraging evaluation results. The major contributions of this work are the following:

a) Our attempt is first of its kind in Indian languages to develop a single generic engine, using machine learning.

b) It is a known fact that most of the Indian languages are resource poor, hence we have used very minimal resources, only shallow parsing has been used.

c) The results obtained are comparable to other reported works.

Reference

Akilandeswari A., and Sobha Lalitha Devi. (2013). Conditional Random Fields Based Pronominal Resolution in Tamil. *International Journal on Computer Science and Engineering*, Vol. 5 Issue 6 pp 601–610.

Akilandeswari A, Bakiyavathi T and Sobha Lalitha Devi, (2012), "atu Difficult Pronominal in Tamil", *In Proceedings of Lrec 2012*, Istanbul

Aone C., and McKee D. (1993).A Language-Independent Anaphora Resolution System for Understanding Multilingual Texts. *In proceeding of ACL 1993*, pp 156-163.

Aone C., and Bennett S. (1995). Evaluating automated and manual acquisition of anaphora resolution strategies. *In: 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 122-129.

Balaji J., Geetha T.V., Ranjani Parthasarathi R., Karky M. (2012).Two-Stage Bootstrapping for Anaphora Resolution *In: Proceedings of COLING 2012*, pp 507–516.

Carbonell J. G., and Brown R. D. (1988). Anaphora resolution: A multi-strategy approach. *In: 12th International Conference on Computational Linguistics*, 1988, pp. 96-101.

Charniak, E., and Elsner, M. (2009). EM Works for Pronoun Anaphora Resolution. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics* (EACL 2009), Athens, Greece.

Daelemans, W. and van den Bosch, A. (2005). Memory-based language processing. *Cambridge University Press*, Cambridge

Dagan I., and Itai. A. (1990). Automatic processing of large corpora for the resolution of anaphora references. *In: 13th conference on Computational linguistics*, Vol. 3, Helsinki, Finland, pp.330-332.

Dakwale. P., Mujadia. V., Sharma. D.M. (2013). A Hybrid Approach for Anaphora Resolution in Hindi. *In: Proc of International Joint Conference on Natural Language Processing*, Nagoya, Japan, pp.977–981.

Dutta. K., Prakash. N. and Kaushik. S. (2008). Resolving Pronominal Anaphora in Hindi using Hobbs‟ algorithm," *Web Journal of Formal Computation and Cognitive Linguistics*, Issue 10, 2008.

Li., F., Shi., S., Chen., Y., and Lv, X. (2008). Chinese Pronominal Anaphora Resolution Based on Conditional Random Fields. *In: International Conference on Computer Science and Software Engineering*, Washington, DC, USA, pp. 731-734.

Hendrickx I., Hoste V., and Daelemans W. (2008). Semantic and syntactic features for Dutch coreference resolution. In Gelbukh A. (Ed.), *CICLing-2008 conference*, Vol. 4919 LNCS, Berlin, Springer Verlag, pp. 731-734.

Hobbs J. (1978). Resolving pronoun references. *Lingua 44*, pp. 339-352.

Grosz, B. J. (1977). The representation and use of focus in dialogue understanding. *Technical Report 151*, SRI International, 333 Ravenswood Ave, Menlo Park, Ca. 94025.

Joshi A. K., and Kuhn S. (1979). Centered logic: The role of entity centered sentence representation in natural language inferencing. *In: International Joint Conference on Artificial Intelligence*.

Joshi A. K., and Weinstein S. (1981). Control of inference: Role of some aspects of discourse structure – centering", *In: International Joint Conference on Artificial Intelligence*, pp. 385-387.

Kennedy, C., Boguraev, B. (1996) Anaphora for Everyone: Pronominal Anaphora Resolution without a Parser. In: *16th International Conference on Computational Linguistics COLING'96*, Copenhagen, Denmark, pp. 113–118.

Lappin S., and Leass H. J. (1994). An algorithm for pronominal anaphora resolution. *Computational Linguistics* 20 (4), pp. 535-561.

McCallum A., and Wellner. B. (2003). Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proceedings of the IJCAI Workshop on Information Integration on the Web*, pp. 79–84.

McCarthy, J. F. and Lehnert, W. G. (1995). Using decision trees for coreference resolution. In C. Mellish (Ed.), *Fourteenth International Conference on Artificial Intelligence*, pp. 1050-1055

Mitkov R. (1998). Robust pronoun resolution with limited knowledge. *In: 17th International Conference on Computational Linguistics* (COLING' 98/ACL'98), Montreal, Canada, pp. 869-875.

Mitkov, R. (1997). "Factors in anaphora resolution: they are not the only things that matter. A case study based on two different approaches". *In Proceedings of the ACL'97/EACL'97 workshop on Operational factors in practical, robust anaphora resolution*, Madrid, Spain.

Murthy K.N., Sobha L, Muthukumari B. (2007). Pronominal Resolution in Tamil Using Machine Learning Approach. *The First Workshop on Anaphora Resolution (WAR I)*, Ed Christer Johansson, Cambridge Scholars Publishing, 15 Angerton Gardens, Newcastle, NE5 2JA, UK, pp.39-50.

Ng V., and Cardie C. (2002). Improving machine learning approaches to coreference resolution. *In. 40th Annual Meeting of the Association for Computational Linguistics*, pp. 104-111.

Prasad R., and Strube,M.,(2000). Discourse Salience and Pronoun Resolution in Hindi, *Penn Working Papers in Linguistics*, Vol 6.3, pp. 189-208.

Ram, R.V.S. and Sobha Lalitha Devi. (2013)."Pronominal Resolution in Tamil Using Tree CRFs", *In Proceedings of 6th Language and Technology Conference, Human Language Technologies as a challenge for Computer Science and Linguistics* - 2013, Poznan, Poland

Recasens M., M`arquez L., Sapena E., Mart´I M.A., Taul´e M., Hoste V., Poesio M., Versley Y. (2010). SemEval-2010 Task 1: Coreference Resolution in Multiple Languages. In Proceedings of the *5th International Workshop on Semantic Evaluation, ACL 2010*, Uppsala, Sweden, .pages 1–8.

Recasens M., Hovy E. (2009). A Deeper Look into Features for Coreference Resolution. Lalitha Devi, S., Branco, A. and Mitkov, R. (eds.), *Anaphora Processing and Applications (DAARC 2009)*, LNAI 5847, Springer-Verlag Berlin Heidelberg, pp 535-561.

Rich, E. and LuperFoy S., (1988) An architecture for anaphora resolution. *In: Proceedings of the Second Conference on Applied Natural Language Processing*, Austin, Texas.

Senapati A., Garain U. (2013). GuiTAR-based Pronominal Anaphora Resolution in Bengal. *In: Proceedings of 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria pp 126–130.

Sikdar U.K, Ekbal A., Saha S., Uryupina O., Poesio M. (2013). Adapting a State-of-the-art Anaphora Resolution System for Resource-poor Language. *In proceedings of International Joint Conference on Natural Language Processing*, Nagoya, Japan pp 815–821.

Sobha L. and Patnaik B. N. (2000). Vasisth: An Anaphora Resolution System for Indian Languages. *In Proceedings of International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications*, Monastir, Tunisia.

Sobha L. and Patnaik,B.N. (2002). Vasisth: An anaphora resolution system for Malayalam and Hindi. *In Proceedings of Symposium on Translation Support Systems.*

Sobha L. (2007). Resolution of Pronominals in Tamil. Computing Theory and Application, *The IEEE Computer Society Press*, Los Alamitos, CA, pp. 475-79.

Sobha L., Pralayankar P. (2008). Algorithm for Anaphor Resolution in Sanskrit. *In Proceedings of 2nd Sanskrit Computational Linguistics Symposium*, Brown University, USA, 2008.

Sobha, Lalitha Devi., Vijay Sundar Ram and Pattabhi RK Rao. (2011). Resolution of Pronominal Anaphors using Linear and Tree CRFs. *In. 8th DAARC*, Faro, Portugal, 2011.

Sobha L., Sivaji Bandyopadhyay, Vijay Sundar Ram R., and Akilandeswari A. (2011). NLP Tool Contest @ICON2011 on Anaphora Resolution in Indian Languages. *In: Proceedings of ICON 2011*.

Soon W. H., Ng, and Lim D. (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics* 27 (4), pp.521-544.

Strube, M. and Hahn U., (1999). Functional centering: Grounding referential coherence in information structure. *Computational Linguistics*, 25(3) pp 309–344

Taku Kudo. 2005. CRF++, an open source toolkit for CRF, *http://crfpp.sourceforge.net* .

Uppalapu. B., and Sharma, D.M. (2009). Pronoun Resolution For Hindi. *In: Proceedings of 7th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 09)*, pp. 123-134.

Wick M., Singh S., and McCallum A. (2012). A Discriminative Hierarchical Model for Fast Coreference At Large Scale. *In: Proceedings of ACL 2012*.

Converting Phrase Structures to Dependency Structures in Sanskrit

Pawan Goyal
Department of CSE
Indian Institute of Technology
Kharagpur, India – 721302
pawang@cse.iitkgp.ernet.in

Amba Kulkarni
Department of Sanskrit Studies
University of Hyderabad
Hyderabad, India – 500046
apksh@uohyd.ernet.in

Abstract

Two annotations schemes for presenting the parsed structures are prevalent viz. the constituency structure and the dependency structure. While the constituency trees mark the relations due to positions, the dependency relations mark the semantic dependencies. Free word order languages like Sanskrit pose more problems for constituency parses since the elements within a phrase are dislocated. In this work, we show how the enriched constituency tree with the information of displacement can help construct the unlabelled dependency tree automatically.

1 Introduction

Sanskrit has a rich tradition of linguistic analysis with intense discussions and argumentations on various aspects of language analysis ranging from phonetics (śikṣā), grammar (vyākaraṇa), logic (nyāya), ritual exegesis (karmamīmāṃsā), and literary theory (alaṃkāraśāstra) which is not only useful for analysing Sanskrit but it also has much to offer computational linguistics in these areas. The series of symposia in Sanskrit Computational Linguistics (Huet et al., 2009; Kulkarni and Huet, 2009), the consortium project sponsored by the Technology Development for Indian Languages (TDIL) and the research of individual scholars and the collaborations (Goyal et al., 2012) among them resulted into a) development of several tools ranging from segmenters (Huet, 2009), morphological analysers (Kulkarni and Shukl, 2009), parsers (Goyal et al., 2009; Hellwig, 2009; Kumar, 2012; Kulkarni, 2013) to discourse annotators, b) lexical resources ranging from dictionaries, WordNet (Kulkarni et al., 2010) to Knowledge-Nets (Nair, 2011), and c) annotated corpora [http://sanskrit.uohyd.ernet.in/scl].

Pāṇinian grammar, the oldest dependency grammar, provides a formalism for annotation of the sentences. While the Sanskrit consortium has annotated a few thousand sentences following the dependency grammar, we also came across a very valuable source of annotation of Sanskrit sentences following the constituency structure (Gillon, 1996). The constituency structure was enriched to suite the requirements of Sanskrit. This aroused our curiosity to study the equivalence of the two annotation schemes.

The importance of dependency structure has been well recognised by several computational linguists (Culotta and Sorensen, 2004; Haghighi et al., 2005; Quirk et al., 2005) in the recent past. The dependency format is preferred over the constituency not only from evaluation point of view (Lin, 1998) but also because of its suitability (Marneffe et al., 2006) for a wide range of NLP tasks such as Machine Translation (MT), information extraction, question answering etc.. This has upsurged several works on converting a constituency structure into dependency. The parsers for English now produce the dependency parse as well. Xia and Palmer discuss three different algorithms to convert dependency structures to phrase structures for English (Xia and Palmer, 2001). Magerman gave a set of priority lists, in the form of a head percolation table to find heads of constituents (Magerman, 1994). Yamada and Matsumoto modified these head percolation rules further (Yamada and Matsumoto, 2003). Their method was reimplemented by Nivre, who also defined certain heuristics to infer the arc labels in the dependency tree produced (Nivre, 2006). Johansson and Nugues used a richer set of edge labels and

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1834–1843, Dublin, Ireland, August 23-29 2014.

introduced links to handle long-distance phenomena such as wh-movement, topicalization, expletives and gapping (Johansson and Nugues, 2007). Their conversion procedure made use of this extended structure in Penn Treebank. De et al. described a system for generating typed dependency parsed from the phrase structure parses (De Marneffe et al., 2006). (Palmer et al., 2009; Xia et al., 2009; Bhatt et al., 2009) discuss a multi-layered representation framework for Hindi and Urdu, where the information from syntactic as well as dependency parse is presented together.

In this work, we explore the relationship between enriched constituency structures and dependency structures for Sanskrit language, with main emphasis on the conversion from constituency to dependency structures. This work aims not only at designing an algorithm to convert Treebanks from one type of representation to the other, but also to judge the adequacy of the enriched constituency structure from parsing point of view. This paper has been organized as follows. Section 2 discusses the history and origin of this work. Section 3 describes the background of the constituency and dependency structures, utilized for Sanskrit language. Section 4 discusses the algorithm we used for converting constituency structure into dependency structure. Section 5 describes the results obtained by our approach, with some examples. Section 6 concludes this paper with the directions for future work.

2 Origin of the work

The dataset we are using in this work has its origin in the remarkable treatise on Sanskrit Syntax by Apte (Apte, 1885) which is the most authentic book on Sanskrit Syntax even after 125 years. The work was initiated in 1986 by Brendan Gillon, then a senior fellow at the American Institute of Indian Studies, at Deccan College, put all the prose exercise sentences from Apte's Student Guide onto 5×7 cards, assigning a syntactic parse to each sentence, giving each sentence an English translation and annotating each sentence for miscellaneous syntactic and semantic facts. On the basis of these sentences, Brendan Gillon published the grammar underlying his syntactic parse of these sentences (Gillon, 1996).

In 1991, Brendan Gillon transferred the material from a paper format to an electronic format, making revisions. An example sentence in this dataset is given below:

```
Example{3}
Source{1.1.3 (P) <U 4.5.3>} % Apte{7,3}

Parse
[S [INJ haa ] [ADV katham ]
   [NP1s [NP6 (mahaaraaja<Dasharathasya) ] (dharma<daaraa.h) ]
   [VP 0 [NP1 (priya<sakhii) [NP6 me ] [NP1 Kaushalyaa ] ] ] ]

Gloss{Oh, how is it that the legal wife of King Dasharatha is my dear
friend Kaushalyaa}

Comment{copula: covert: predicational: NP1s VP }
```

Each example is given a serial number, its source - the corresponding reference in Apte's book. Then, its constituency parse is provided in a tree structure. The Sanskrit text is transliterated into Roman using the Velthuis notation[1]. Finally, the gloss (translation) of the prose is provided along with some observations regarding syntax in the field 'comment'. The proper nouns are transliterated following the English convention of capitalisation. The constituency structure is enriched reflecting the morphological information such as the case marker. The underlying constituency structure of the compounds is also shown clearly marking the head of the compound. The requirement that constituency tree be a binary is also done away with, resulting into a more flat structure than the normal hierarchical phrase structure.

In 2004, Gérard Huet re-engineered the document in order to parse it mechanically, and he verified its correct syntactic structure after typographical corrections. He devised an abstract syntax to formalize this constituency structure. In the abstract syntax, the above constituency structure is represented as below:

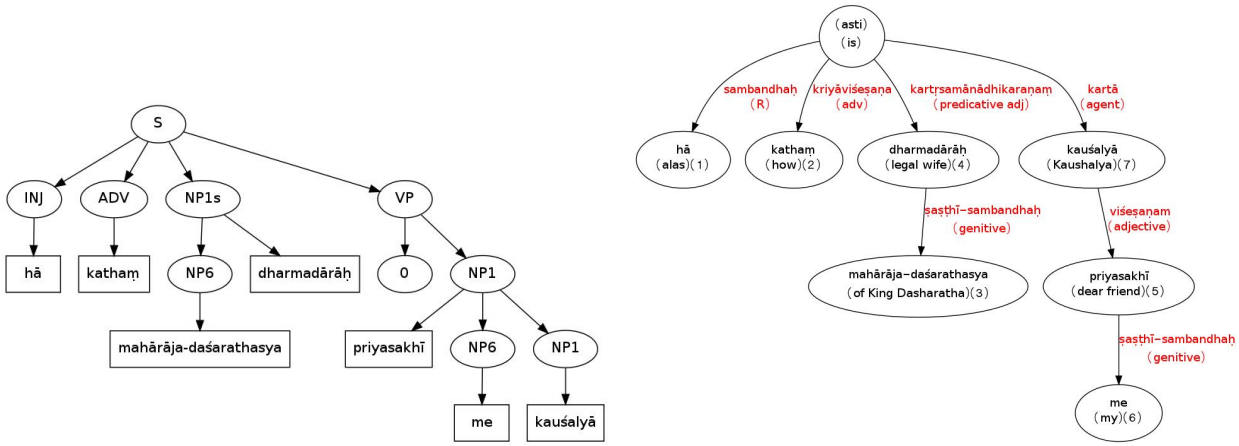
```
list Tag_tree.syntax =
[S
```

[1]Originally developed in 1991 by Frans Velthuis for use with his devnag Devanagari font, designed for the TeX typesetting system.

```
[INJ ("haa", 1); ADV [("katham", 2)];
 NP
  ([Case 1; Role Subject],
  [NP ([Case 6], [N (Compound (Stem <mahaaraaja>, Stem <Dasharathasya>),
   3)]);
   N (Compound (Stem <dharma>, Stem <daaraa.h>), 4)]);
 VP0
  [NP
    ([Case 1],
    [N (Compound (Stem <priya>, Stem <sakhii>), 5);
     NP ([Case 6], [N (Stem <me>, 6)]);
     NP ([Case 1], [N (Stem <Kaushalyaa>, 7)])]);
   NIL 8]]]
```

Each stem is given a unique index. The syntax, while preserving the original structure of the text, gives additional structuring with the word numbers, explicit case markers and stems for the compounds. While these constituent trees preserve much of the tagging related information, they still do not have the gender and number information for the substantives, for instance. This information can enhance the constituency representation further.

The same set of sentences were also parsed manually by Sheetal Pokar, a research scholar at the University of Hyderabad, showing the dependency structure. Sheetal followed the annotation guidelines developed by the Consortium of Institutes working on the Development of Sanskrit Computational Tools[2]. This tagset has a little above 40 tags marking various relations. The dependency tree for the example 3, discussed above, is shown in Figure 1. It is a directed tree with nodes corresponding to the words in the sentence and edges corresponding to the relation between the head and the modifier. Each node has a number indicating the word index. A generic relation *sambandhaḥ* (R) is used if the relation does not fall under any of the given tags. As one may notice, both the constituency as well as the dependency structures posit a NULL verb 'to be' *asti*. Among the Indian schools dealing with verbal cognition, not all schools accept the insertion of missing copula. We follow the grammarian school who accept this insertion.



(a) (b)

Figure 1: Example 3: (a). Constituency Parse and (b). Dependency Parse

While the two structures in Figure 1 mark different kind of information, we notice that the dominance relation in the constituency structure under each phrase corresponds to the the modifier-modified relation in the dependency tree. This was the main motivation to develop the converter to convert a phrase structure into an unlabelled dependency structure.

[2] http://sanskrit.uohyd.ernet.in/scl

3 Dependency and Constituency Structures

Verbal understanding of any utterance requires the knowledge of how words in that utterance are related to each other. There are two major representational frameworks for representing this knowledge as a parse tree viz. constituency and dependency parse trees. Constituency trees show how the individual units in a sentence are grouped together leading to semantically richer phrases in the constituency structures. The dependency structure, on the other hand, shows how each word is related to other words in the sentence either directly or indirectly.

The constituency structure derives from the subject-predicate division of Latin and Greek grammars that is based on term logic. Basic clause structure is understood in terms of a binary division of the clause into subject (noun phrase NP) and predicate (verb phrase VP). These ideas originated with Leonard Bloomfield (Bloomfield, 1962) and were further developed by a number of American structuralist linguists, including Harris (Harris, 1955) and Wells (Wells, 1947). Though these grammars were initially conceived as applying only to phrases, it was shown that such rules could be used for analyzing compounds as well as derivational morphology for Sanskrit. Gillon showed that the same extension works for classical Sanskrit as well (Gillon, 1995).

The dependency analysis dates back to Pāṇini who uses the syntactico-semantic relations called *kāraka* relations for the linguistic analysis of a sentence. In modern times, the seminal work of Tesnière (Tesnière, 1959) became the basis for the work on dependency grammar. Meaning-Text Theory (Melĉuk, 1988), Word grammar (Hudson, 1984), Functional Generative Description (Segall et al., 1986) are some of the flavours of the dependency grammar. A dependency parse is generally modelled as a directed tree with nodes representing the words and edges representing the possible relations between them. A typed dependency parse also labels the relations. For every element (word or morph) in a sentence, there is just one node in the syntactic structure.

Xia and Palmer (Xia and Palmer, 2001) discuss three different algorithms to convert dependency structures to phrase structures for English. They also attempt to clarify the differences in representational coverage of the two approaches. In the first stage, they identify the head of each constituent of the sentence, which is further modified to retrieve the semantic head. In the second phase, they label each of the dependency extracted with a grammatical relation using patterns defined over the phrase structure tree. (Palmer et al., 2009; Xia et al., 2009; Bhatt et al., 2009) discuss a multi-layered representation framework for Hindi and Urdu, where the information from syntactic as well as dependency parse is presented together. They first construct a dependency parse and then convert it into the constituency parse tree using conversion rules. A conversion rule is a (DS pattern, PS pattern) pair, where DS and PS correspond to dependency and phrase structure respectively.

The constituency parse we are dealing with being enriched with linguistic information pertaining to the morphology of the simple as well as compound words, it was much simpler to convert this structure into a dependency structure. In the next section, we will discuss our algorithm for converting a constituency structure to a dependency structure.

4 Conversion from Constituency to Dependency Structure in Sanskrit

The notion of 'head' is very important for both the constituency and dependency structures. In the constituency structure, the head determines the main properties of the phrase. Head may have several levels of projection. In the dependency structure, on the other hand, the head is linked to its dependents. The core of the algorithm is to identify the head of each phrase in the constituency tree and establish its relation with the head of its parent node. And also to establish the relation between the head with its dependents. The head for each XP is the node X within that XP. Thus the head for an NP is the noun, head for a VP is the verb, and so on. The head for the S is the head of a VP, in case it is a simple sentence, and head of the VP of the main clause in case it is a complex sentence. In case of complex sentences, we identified the main clause taking clues from the connectives. Each relation is named after the XP of the modifier. Our algorithm for finding the head node is implemented on the abstract syntax discussed in section 2 before. A rough outline of the algorithm is:

1)The head of VP is the ROOT node in the dependency tree.

a) In the case of sentences with sub-ordinate clauses, identify the main clause taking clues from the connectives.
b) Head of a clause is an auxiliary, if present, otherwise the main verb is the head.
c) In the case of sentences with quotative markers, the verb of the main clause is the head.
(The later rule is stronger than the previous.)
2) All the XPs within VP are dependent on the ROOT.
3) If S is the parent of VP, then all the XPs which are children of S are also dependent on this ROOT.

Finding the head for each node was not trivial though, as many of the parses involve dislocated phrases, which were not fully marked. We had to enrich the constituency trees by incorporating the dislocation information, which was provided in comments and was missing from the tree notations. We used '!' and '$' to indicate the dislocation. '!' indicates the position from where a component is dislocated, while '$' indicates the dislocated component. An example of a constituency parse, enriched with dislocation information is given below.

```
Example{2}
Source{1.1.2 (P) <V 3.28; V 3.6.3>} % Apte{7,2}

Parse
[S [ADV sarvatra ] [NP6 audarikasya $1]
   [VP 0 [NP1 abhyavahaaryam [PRT eva ] ] ]
   [NP1s !1 vi.saya.h ] ]

Gloss{In every case, a glutton's object is only food.}

Comment{copula: covert: predicational: VP NP1s
"eva" in predicate NP
left extraposition from NP1s of NP6 within MC, modulo adverbial ADV.}
```

This constituency tree involves one dislocated phrases. This information is marked with '!1'for the place from which it is dislocated and with '$1' for the phrase that has been dislocated. This dislocation information is used by our algorithm to find the right relata for the dislocated words. In case of more than one dislocated phrases, they are numbered sequentially.

5 Results and Discussions

We implemented our algorithm on a dataset of 232 sentences and matched the output of our algorithm with the Gold dataset, the dependency graph constructed manually for the sentences by Sheetal. Figure 2 shows the dependency structure for Example 3, produced by our conversion algorithm.

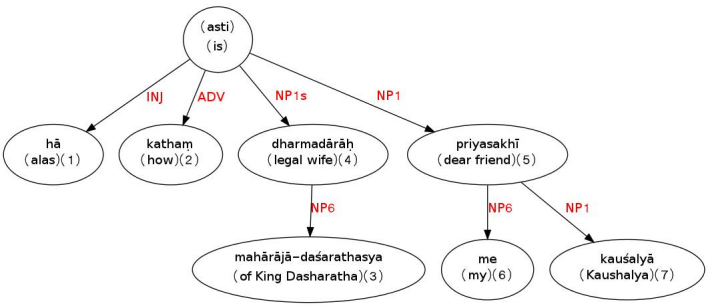


Figure 2: Dependency graph constructed by conversion from constituency structure

The conversion algorithm captures the relations between various constituents and produces a dependency graph. Labels of the dependency graph correspond to the intermediate nodes in the phrase structure. On comparing the graph in Figure 2 with the graph in Figure 1, we find that most of the original connections were captured. However, there was a mismatch in the relation between words *kauśalyā*, *priyasakhī* and *me*. This discrepancy is because of the non-agreement between the annotators as to which one is the head. In case of sentences with apposition, in Sanskrit, the two annotators have difference of opinion as to which among the two is the head. This resulted in the mismatch between the two graphs. The relations in this graph are labelled by the dominating XP.

Pāṇini's grammar provides rules for assigning case markers given the syntactico-semantic relation between the relata. Inverting these rules it should be possible to get the relation labels. These relation labels, however, will not be obtained deterministically. The non-determinism will lead to multiple labelled dependency structures. Hence we could not assign the labels from the tagset, and resorted to the names of the phrases which the word belongs to.

Below we give an example where we found an exact match between the manual dependency graph and the dependency graph, produced by our conversion algorithm, using the constituency parse of the sentence.

```
Example{29}
[S [VP [NP7 tatra ] [CNJ ca ]
       [NP5 [NP6 [AP6 (((nikhila<(dhara.nii<tala))<parya.tana)<khinnasya) ]
                  (nija<balasya) ]
            (vizraama<heto.h) ]
       [NP2 [AP2 katipayaan ] divasaan ]
       ati.s.that ] ]
Gloss{And he remained there for a few days in order to rest his army
exhausted from roaming the entire surface of the earth.}
```

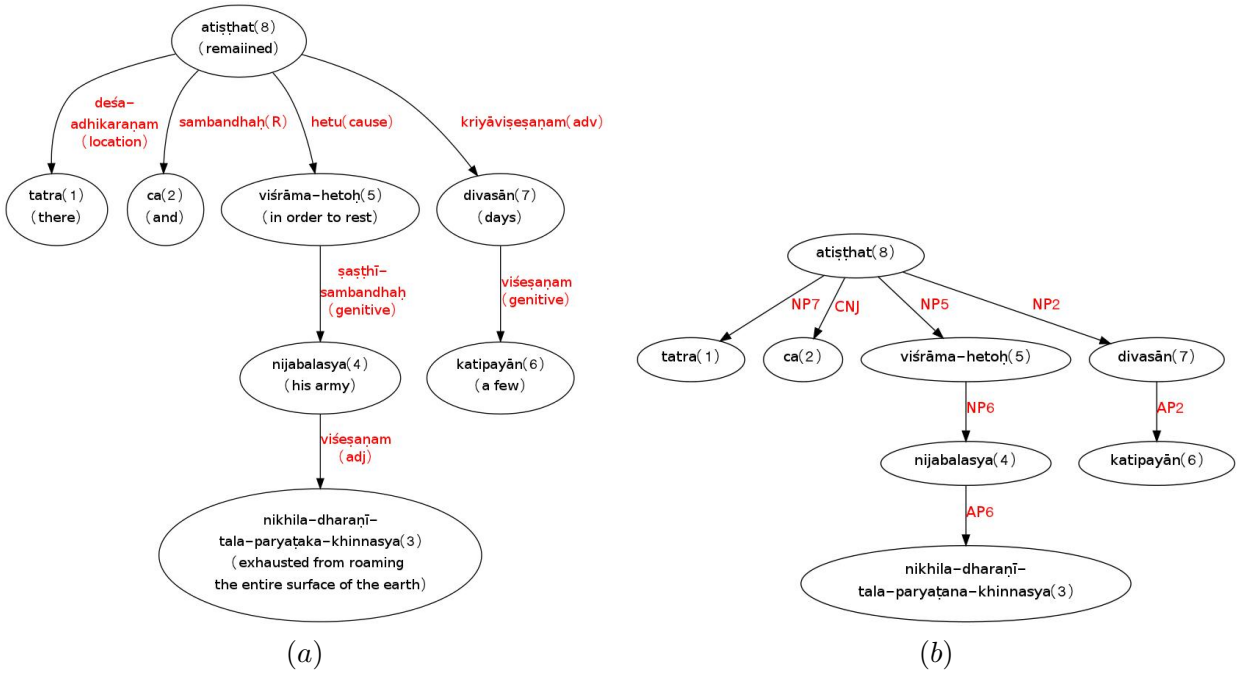


Figure 3: Example 29: (a). Dependency graph constructed manually and (b). Dependency graph constructed by conversion from constituency structure

Out of 232 cases, we found 97 such cases with exact match. For the rest of the cases,
1) In 40 cases, number of words in dependency and phrase-converted graph are different. For example, the words such as *kadācit* 'probably', *yadyapi* 'even if', *tathāpi* 'even then', *athavā* 'or' etc. were treated in one structure as a single word while in the other as two words. These words at morphological level consist of two morphemes which have independent existence. So it was natural to treat these as two words, in the constituency trees. However, at the semantic level, these two words indicate a single meaning which at times is non-compositional. The annotator of dependency graph has treated these as a single word.
2) In 95 cases, one or more relations do not match. These were due to various reasons such as

1. differences in the treatment and identification of adjectives,
2. disagreement in the attachment, and
3. cases of ellipsis, null head, and cases where the treatment of conjunct '*ca*' (and) differs.

These differences are very much important from linguistic analysis point of view. However due to space constraint we illustrate here only two cases where the treatment of the two annotators differ.

```
Example{72}
[S [VOC sakhi [VOC Vaasanti ] ]
   [VP 0 [NP4 du.hkhaaya [PRT eva ] [NP6 su-h.rdaam ] ] ]
   [NP1s [ADV idaaniim ] [NP6 Raamasya ] darshanam ] ]

Gloss{Oh my friend Vaasanti, seeing Raama now leads only to
the unhappiness of his friends.}
```

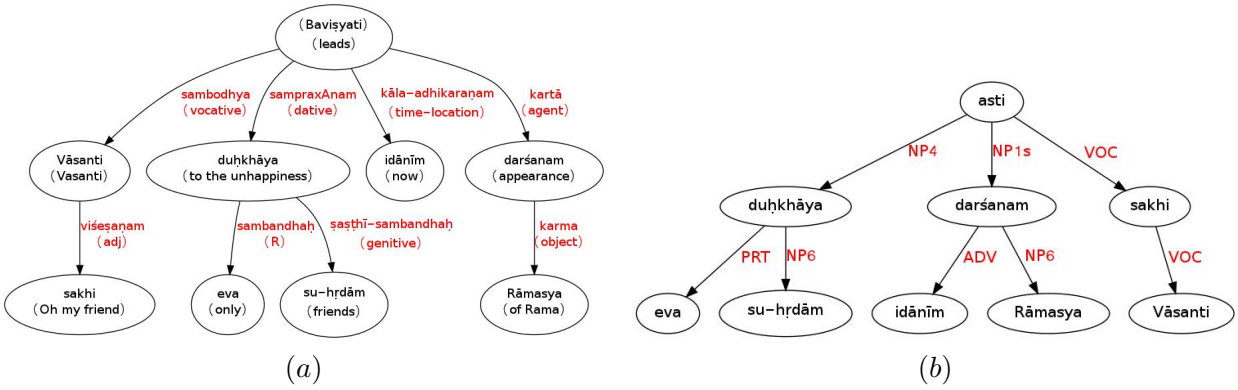


Figure 4: Example 72: (a). Dependency graph constructed manually and (b). Dependency graph constructed by conversion from constituency structure

In this example, there are two places where the annotators disagree.
a) The attachment of the word *idānīm* (now).
b) The decision of head in case of vocative with a modifier.
The null copula (VP 0) in the constituency structure is replaced by the Sanskrit verbal form *asti* uniformly. The annotator of dependency graph has provided a more appropriate verb *bhaviṣyati*. However, we ignore this difference.

Let us look at another example, where the ambiguity in morphological analysis has led to the ambiguity in the readings resulting in the disagreement in the parse.

```
Example{46}
[S [NP1s aarya.h ]
   [VP daapayatu [NP2 [NP6 me ]
                      [NP4 (Vaisha.mpaayana<aanayanaaya) ]
                      (gamana<abhyanuj~naam) ]
                 [NP3 taatena ] ] ]

Gloss{May you, sir, make my father give me permission to go to bring
Vaisha.mpaayana.}
```

In this example, the pronominal form *me* is ambiguous between two readings. It can be either a genitive or a dative of the first person pronoun *asmad* 'I'. The sub-ordinate clause is analysed by Gillon as 'the permission for going to bring Vaiśampāyan by me'. In Sanskrit the first person pronoun in such cases takes genitive case marker. Sheetal on the other hand has analysed it as 'the permission to me for going to bring Vaiśampāyan', where *me* is analysed with dative case. It is clear that 'to bring Vaiśampāyan' is the purpose for going. But since 'permission for going' is a compound in Sanskrit [3], and the 'permission' being the head of this compound, Sheetal avoided linking 'to bring Vaiśampāyana' with 'permission to go' as it results into an '*asamartha samāsa* (incompatible compound formation, where the external modifier connects the modifier component of a compound and not the head). This has resulted into the differences in annotation. Such compounds are not rare. Thus, in order to provide a correct parse, in case of dependency structure, it is necessary to show the internal structure of the compounds as well,

[3]In Sanskrit a compound is always written as a single word without any space in between.

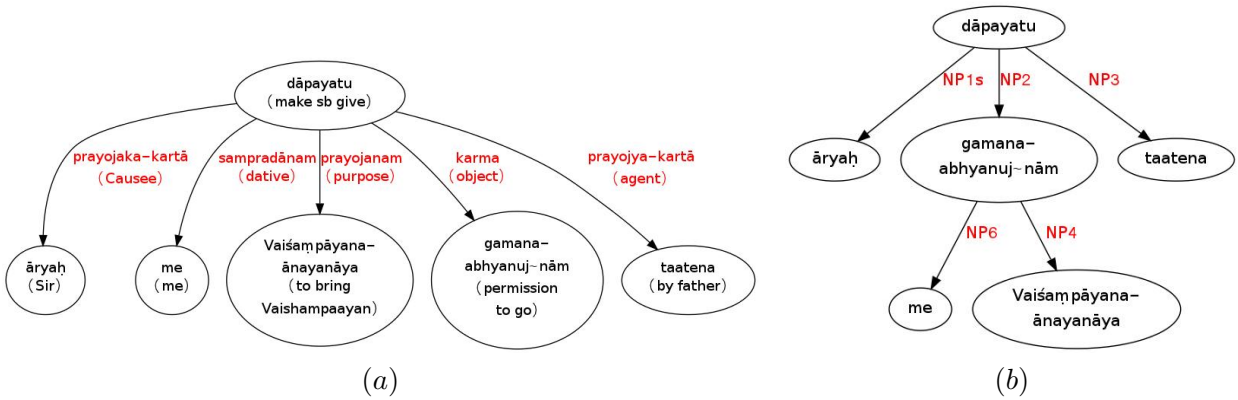


Figure 5: Example 46: (a). Dependency graph constructed manually and (b). Dependency graph constructed by conversion from constituency structure

rather than treating a compound unanalysed. This will allow one to connect the elements to the part of a compound other than the head. Similar treatment is necessary in the phrase structure annotation as well.

We end with example 2 from section 4, where the dislocation information in the constituency tree helps in retrieving the correct dependency structure. In this example, even though the word *audarikasya* has been displaced, the displacement information in the parse tree positions it at the correct place in the dependency tree constructed from the constituency structure.

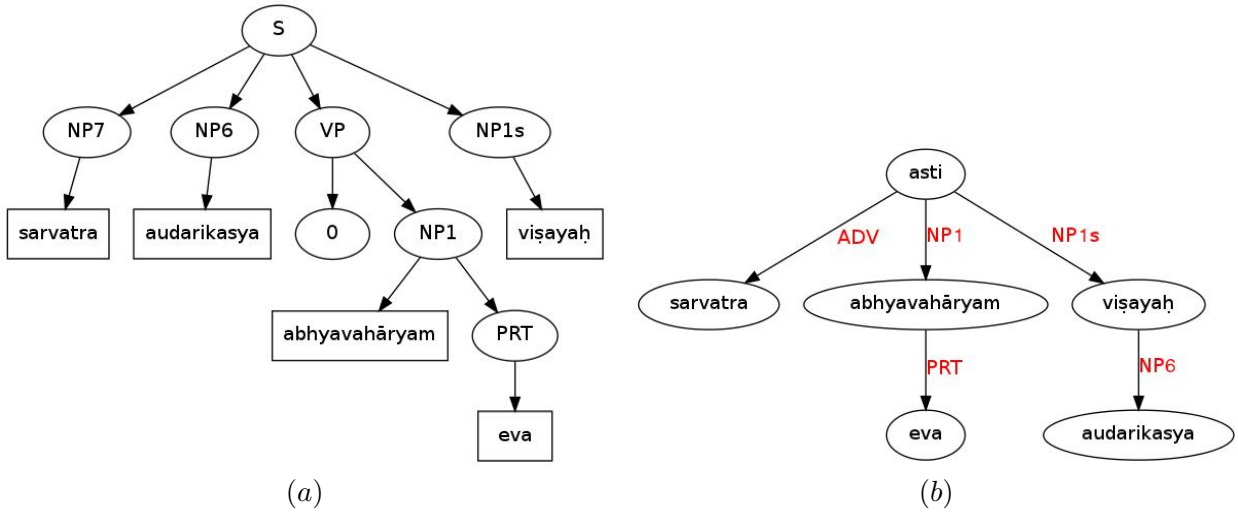


Figure 6: Example 2: (a). Constituency structure and (b). Dependency graph constructed by conversion from constituency structure

6 Conclusions and Future Work

This work focussed mainly on conversion from constituency to dependency structure. The sentences in our dataset are chosen from (Apte, 1885), which is an authentic book for higher learning of Sanskrit, covering a wide range of grammatical constructions. The tool was tested on a dataset of 232 sentences and the initial results were encouraging. Specifically, most of the cases of mismatch were linguistic issues and need further discussion. The phrase labels indicating the case labels is an important extension of the constituency trees to accommodate morphologically rich languages. The enriched constituency structure has an advantage of recording the word order, and at the same time marking the dislocation information. In this work, we have shown that such enriched constituency tree can help construct the unlabelled dependency tree automatically. Further, one may also try inferring the dependency relation names, and use statistical parsing to resolve non-determinism favouring popular usages.

Another interesting aspect would be to try the other way conversion, that is, from dependency to phrase structure. The main challenge for this conversion is to find out the projection table corresponding to each lexical item. This work will also be a first step towards an abstract syntax, which can inherit the properties of both the constituency and dependency structures. If so, this would be an alternative formalism for tagging the Sanskrit corpus.

Acknowledgements

The authors would like to acknowledge the discussions with Gérard Huet, INRIA Paris Rocquencourt, towards enriching the abstract syntax. The work was also supported by Emilie Aussant and Sylvie Archaimbault, laboratoire HTL, Université Paris Diderot.

References

Vāman Shivarām Apte. 1885. *The Student's Guide to Sanskrit Composition. A Treatise on Sanskrit Syntax for Use of Schools and Colleges*. Lokasamgraha Press, Poona, India.

Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 186–189. Association for Computational Linguistics.

Leonard Bloomfield. 1962. Language. *New York: Holt*.

A. Culotta and J. Sorensen. 2004. Dependency tree kernels for relation extraction. In *42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 423–429, Barcelona, Spain.

Monali Das and Amba Kulkarni. 2013. Discourse level tagger for mahābhāṣya - a Sanskrit commentary on pāṇini's grammar. In *Proceedings of the 10th International Conference on NLP, Delhi, India*.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Brendan S. Gillon. 1995. Autonomy of word formation: evidence from Classical Sanskrit. *Indian Linguistics, 56 (1-4)*, pages 15–52.

Brendan S Gillon. 1996. Word order in classical sanskrit. *Indian Linguistics*, 57(1-4):1–35.

Brendan S. Gillon. 2009. Tagging classical Sanskrit compounds. In Amba Kulkarni and Gérard Huet, editors, *Sanskrit Computational Linguistics 3*, pages 98–105. Springer-Verlag LNAI 5406.

Pawan Goyal, Vipul Arora, and Laxmidhar Behera. 2009. Analysis of Sanskrit text: Parsing and semantic relations. In Gérard Huet, Amba Kulkarni, and Peter Scharf, editors, *Sanskrit Computational Linguistics 1 & 2*, pages 200–218. Springer-Verlag LNAI 5402.

Pawan Goyal, Gérard Huet, Amba Kulkarni, Peter Scharf, and Ralph Bunker. 2012. A distributed platform for sanskrit processing. In *Proceedings of 24th COLING, Mumbai, India*.

A.D. Haghighi, A.Y. Ng, and C.D. Manning. 2005. Robust textual inference via graph matching. In *Human Language Technology Conference (HLT) and Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 387–394, Vancouver, Canada.

Zellig S Harris. 1955. From phoneme to morpheme. *Language*, 31(2):190–222.

Oliver Hellwig. 2009. Extracting dependency trees from Sanskrit texts. In Amba Kulkarni and Gérard Huet, editors, *Sanskrit Computational Linguistics 3*, pages 106–115. Springer-Verlag LNAI 5406.

R. Hudson. 1984. *Word Grammar*. Basil Blackwell, Oxford.

Gérard Huet, Amba Kulkarni, and Peter Scharf, editors. 2009. *Sanskrit Computational Linguistics 1 & 2*. Springer-Verlag LNAI 5402.

Gérard Huet. 2009. Formal structure of Sanskrit text: Requirements analysis for a mechanical Sanskrit processor. In Gérard Huet, Amba Kulkarni, and Peter Scharf, editors, *Sanskrit Computational Linguistics 1 & 2*. Springer-Verlag LNAI 5402.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for english. In *Proceedings of the 16th Nordic Conference on Computational Linguistics (NODALIDA)*, pages 105–112.

Amba Kulkarni and Monali Das. 2012. Discourse analysis of sanskrit texts. In *Proceedings of the workshop on Advances in Discourse Analysis and its Computational Aspects, 24th COLING, Mumbai, India*.

Amba Kulkarni and Gérard Huet, editors. 2009. *Sanskrit Computational Linguistics 3*. Springer-Verlag LNAI 5406.

Amba Kulkarni and K. V. Ramakrishnamacharyulu. 2013. Parsing Sanskrit texts: Some relation specific issues. In Malhar Kulkarni, editor, *Proceedings of the 5th International Sanskrit Computational Linguistics Symposium*. D. K. Printworld(P) Ltd.

Amba Kulkarni and Devanand Shukl. 2009. Sanskrit morphological analyser: Some issues. *Indian Linguistics*, 70(1-4):169–177.

Malhar Kulkarni, Chaitali Dangarikar, Irawati Kulkarni, Abhishek Nanda, and Pushpak Bhattacharyya. 2010. Introducing sanskrit wordnet. In Christiane Felbaum Pushpak Bhattacharyya and Piek Vossen, editors, *Principles, Construction and Application of Multilingual Wordnets, Proceedings of the Global Wordnet Conference, 2010*. Narosa Publishing House, New Delhi.

Amba Kulkarni. 2013. A deterministic dependency parser with dynamic programming for Sanskrit. In *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*, pages 157–166, Prague, Czech Republic, August. Charles University in Prague, Matfyzpress, Prague, Czech Republic.

Anil Kumar. 2012. *An automatic Sanskrit Compound Processing*. Ph.D. thesis, University of Hyderabad, Hyderabad.

Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Workshop on the evaluation of Parsing Systems, Granada, Spain.*

David Mitchell Magerman. 1994. *Natural Language Parsing As Statistical Pattern Recognition*. Ph.D. thesis, Stanford, CA, USA. UMI Order No. GAX94-22102.

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

I. Melĉuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press, Albany.

Sivaja Nair. 2011. *The Knowledge Structure in Amarakośa*. Ph.D. thesis, University of Hyderabad, Hyderabad.

Joakim Nivre. 2006. *Inductive dependency parsing*. Springer.

Martha Palmer, Rajesh Bhatt, Bhuvana Narasimhan, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure. In *The 7th International Conference on Natural Language Processing*, pages 14–17.

C. Quirk, A. Menezes, and C. Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal smt. In *43rd Annual Meeting of the Association for, Computational Linguistics (ACL)*, pages 271–279, Ann Arbor, USA.

Peter Scharf and Malcolm Hyman. 2009. *Linguistic Issues in Encoding Sanskrit*. Motilal Banarsidass, Delhi.

P. Segall, E. Hajiov, and J. Panevov. 1986. *The Meaning of the Sentence in its Semantic and Pragmatic Aspects*. Springer, Heidelberg.

L Tesnière. 1959. *Éléments de syntaxe structurale*. Klinksieck, Paris.

Rulon S Wells. 1947. Immediate constituents. *Language*, 23(2):81–117.

Fei Xia and Martha Palmer. 2001. Converting dependency structures to phrase structures. In *Proceedings of the first international conference on Human language technology research*, pages 1–5. Association for Computational Linguistics.

Fei Xia, Owen Rambow, Rajesh Bhatt, Martha Palmer, and Dipti Misra Sharma. 2009. Towards a multi-representational treebank. In *The 7th International Workshop on Treebanks and Linguistic Theories. Groningen, Netherlands*.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. *Proceedings of IWPT*, 3.

Uncertainty Detection in Hungarian Texts

Veronika Vincze[1,2]
[1]University of Szeged
Department of Informatics
[2]MTA-SZTE Research Group on Artificial Intelligence
vinczev@inf.u-szeged.hu

Abstract

Uncertainty detection is essential for many NLP applications. For instance, in information retrieval, it is of primary importance to distinguish among factual, negated and uncertain information. Current research on uncertainty detection has mostly focused on the English language, in contrast, here we present the first machine learning algorithm that aims at identifying linguistic markers of uncertainty in Hungarian texts from two domains: Wikipedia and news media. The system is based on sequence labeling and makes use of a rich feature set including orthographic, lexical, morphological, syntactic and semantic features as well. Having access to annotated data from two domains, we also focus on the domain specificities of uncertainty detection by comparing results obtained in indomain and cross-domain settings. Our results show that the domain of the text has significant influence on uncertainty detection.

1 Introduction

Uncertainty detection has become one of the most intensively studied problems of natural language processing (NLP) in these days (Morante and Sporleder, 2012). For several NLP applications, it is essential to distinguish between factual and nonfactual, i.e. negated or uncertain information: for instance, in medical information retrieval, it must be known whether the patient definitely suffers, probably suffers or does not suffer from an illness. This type of information can only be revealed from the texts of the documents if reliable uncertainty detectors are available, which are able to identify linguistic markers of uncertainty, i.e. cues within the text. To the best of our knowledge, uncertainty detectors have been mostly developed for the English language (Morante and Sporleder, 2012; Farkas et al., 2010). Here, we present our machine learning based uncertainty detector developed for Hungarian, a morphologically rich language, and report our results on a manually annotated uncertainty corpus, which contains texts from two domains: first, Hungarian Wikipedia texts and second, pieces of news from a Hungarian news portal.

The main contributions of this paper are the following:

- it presents the first uncertainty corpus for Hungarian;
- it reports the first results on uncertainty detection in Hungarian texts;
- it introduces new features in the machine learning setting like semantic and pragmatic features;
- we show that there are domain specificities in the distribution of uncertainty cues in Hungarian texts;
- we show that domain specificities have a considerable effect on the efficiency of machine learning.

The structure of the paper is the following. First, related work on uncertainty detection is presented. Then our corpus is described in detail, which is followed by the elaboration of machine learning methods and results on uncertainty detection. The paper concludes with a discussion of results and possible ways for future work are also outlined.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1844–1853, Dublin, Ireland, August 23-29 2014.

2 Related Work

In these days, identifying uncertainty cues is one of the popular topics in NLP. This is supported by the CoNLL-2010 Shared Task, which aimed at detecting uncertainty cues in biological papers and Wikipedia articles written in English (Farkas et al., 2010). Moreover, a special issue of the journal Computational Linguistics (Vol. 38, No. 2) was recently dedicated to detecting modality and negation in natural language texts (Morante and Sporleder, 2012). As indicated above, most earlier research on uncertainty detection focused on the English language. As for the domains of the texts, newspapers (Saurí and Pustejovsky, 2009), biological or medical texts (Szarvas et al., 2012; Morante et al., 2009; Farkas et al., 2010; Kim et al., 2008), Wikipedia articles (Ganter and Strube, 2009; Farkas et al., 2010; Szarvas et al., 2012) and most recently social media texts (Wei et al., 2013) have been selected for the experiments.

Systems for uncertainty detection were originally rule-based (Light et al., 2004; Chapman et al., 2007) but recently, they exploit machine learning methods, usually applying a supervised approach (see e.g. Medlock and Briscoe (2007), Morante et al. (2009), Özgür and Radev (2009), Szarvas et al. (2012) and the systems of the CoNLL-2010 Shared Task (Farkas et al., 2010)). In harmony with the latest tendencies, our system here is also based on supervised machine learning techniques, which employs a rich feature set of lexical, morphological, syntactic and semantic features and also exploits contextual features.

Supervised machine learning methods require annotated corpora. There have been several corpora annotated for uncertainty in different domains such as biology (Medlock and Briscoe, 2007; Kim et al., 2008; Settles et al., 2008; Shatkay et al., 2008; Vincze et al., 2008; Nawaz et al., 2010), medicine (Uzuner et al., 2009), news media (Saurí and Pustejovsky, 2009; Wilson, 2008; Rubin et al., 2005; Rubin, 2010), encyclopedia (Farkas et al., 2010), reviews (Konstantinova et al., 2012; Cruz Díaz, 2013) and social media (Wei et al., 2013). For our experiments, however, we make use of the first Hungarian uncertainty corpus created for the purpose of this study.

3 Experiments

In this section, we present our methodology to detect uncertainty cues in Hungarian. We first describe the uncertainty categories applied and report some statistics on the corpus. Then we describe our machine learning approach based on a rich feature set.

3.1 The hUnCertainty Corpus

For the purpose of this study, we manually annotated texts from two domains. First, we randomly selected 1,081 paragraphs from the Hungarian Wikipedia dump. This selection contains 9,722 sentences and 180,000 tokens. Second, we downloaded 300 pieces of criminal news from a Hungarian news portal (`http://www.hvg.hu`), which altogether consist of 5,481 sentences and 94,000 tokens. In total, the hUnCertainty corpus consists of 15,203 sentences and 274,000 tokens.

During annotation, we followed the categorization of uncertainty phenomena as described in Szarvas et al. (2012) and Vincze (2013) with some slight modifications, due to the morphologically rich nature of Hungarian (for instance, modal auxiliaries like *may* correspond to a derivational suffix in Hungarian, which required that in the case of *jöhet* "may come" the whole word was annotated as uncertain, not just the suffix *-het*). Here we just briefly summarize uncertainty categories that were annotated – for a detailed discussion, please refer to Szarvas et al. (2012) and Vincze (2013).

Linguistic uncertainty is traditionally connected to modality and the semantics of the sentence. For instance, the sentence *It may be raining* does not contain enough information to determine whether it is really raining (semantic uncertainty). There are several phenomena that are categorized as semantic uncertainty. A proposition is **epistemically** uncertain if its truth value cannot be determined on the basis of world knowledge. **Conditionals** and **investigations** also belong to this group – the latter is especially frequent in research papers, where authors usually formulate the research question with the help of linguistic devices expressing this type of uncertainty. Non-epistemic types of modality may also be listed here such as **doxastic** uncertainty, which is related to beliefs.

However, there are other uncertainty phenomena that only become uncertain within the context of communication. For instance, the sentence *Many researchers think that COLING will be the best conference of the year* does not reveal how many (and which) researchers think that, hence the source of the proposition about COLING remains uncertain. This is a type of discourse-level uncertainty, more specifically, it is called **weasel** (Ganter and Strube, 2009). On the other hand, **hedges** make the meaning of words fuzzy: they blur the exact meaning of some quality/quantity. Finally, **peacock** cues express unprovable (or unproven) evaluations, qualifications, understatements and exaggerations.

Some examples of uncertainty cues are offered here (in English, for the sake of simplicity):

EPISTEMIC: It **may** be raining.

DYNAMIC: I **have to** go.

DOXASTIC: He **believes** that the Earth is flat.

INVESTIGATION: We **examined** the role of NF-kappa B in protein activation.

CONDITION: **If** it rains, we**'ll** stay in.

WEASEL: **Some** note that the number of deaths during confrontations with police is relatively proportional for a city the size of Cincinnati.

HEDGE: Magdalene Asylums were a **generally** accepted social institution until well into the second half of the 20th century.

PEACOCK: The main source of their inspiration was native Georgia, with its **rich** and **complex** history and culture, its **breathtaking** landscapes and its **courageous** and **hardworking** people.

Table 1 reports some statistics on the frequency of uncertainty cues in Hungarian and it is also visualized in Figure 1. It is revealed that the domain of the texts has a strong effect on the distribution of uncertainty cues: the distribution of semantic uncertainty cues and discourse-level uncertainty cues is balanced in the news subcorpus but in the Wikipedia corpus, about 85% of the cues belong to the discourse-level uncertainty type.

Regarding different classes of uncertainty, we should mention that while weasels constitute the most frequent cue category in Wikipedia texts, they occur less frequently in the news corpus. On the other hand, doxastic cues are frequent in the news corpus but in Wikipedia texts, their number is considerably smaller.

Uncertainty cue	**Wikipedia**		**News**		**Total**	
	#	%	#	%	#	%
Weasel	2150	35.95	258	10.93	2408	28.87
Hedge	2100	35.12	800	33.88	2900	34.77
Peacock	788	13.18	94	3.98	882	10.57
Discourse-level total	5038	84.25	1152	48.79	6190	74.21
Epistemic	441	7.37	358	15.16	799	9.58
Doxastic	316	5.28	710	30.07	1026	13.30
Conditional	154	2.58	128	5.42	282	3.38
Investigation	31	0.52	13	0.55	44	0.53
Semantic total	942	15.75	1209	51.21	2151	25.79
Total	5980	100	2361	100	8341	100

Table 1: Uncertainty cues.

3.2 Machine Learning Methods

In order to automatically identify uncertainty cues, we developed a machine learning method to be discussed below. In our experiments, we used the above-described corpus and morphologically and syntactically parsed it with the help of the toolkit `magyarlanc` (Zsibrita et al., 2013).

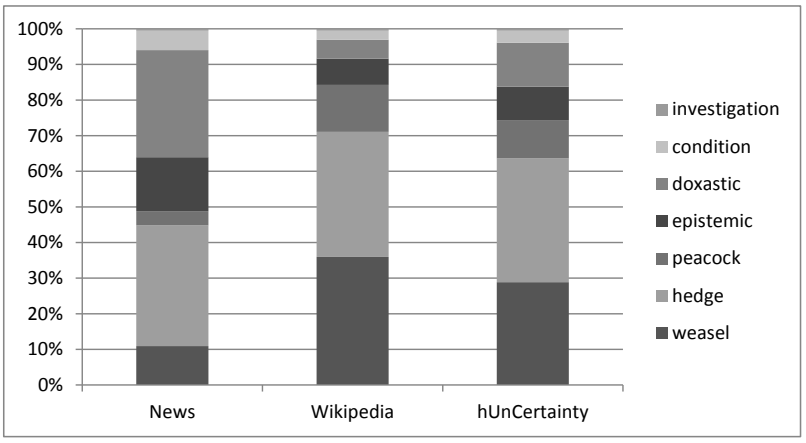


Figure 1: Distribution of cues across domains.

On the basis of results reported in earlier literature, sequence labeling proved to be one of the most successful methods on English uncertainty detection (see e.g. Szarvas et al. (2012)), hence we also applied a method based on conditional random fields (CRF) (Lafferty et al., 2001) in our experiments. We used the MALLET implementation (McCallum, 2002) of CRF with the following rich feature set:

- **Orthographic features:** we investigated whether the word contains punctuation marks, digits, uppercase or lowercase letters, the length of the word, consonant bi- and trigrams...
- **Lexical features:** we automatically collected uncertainty cues from the English corpora annotated on the basis of similar linguistic principles and manually translated these lists into Hungarian. Lists were used as binary features: if the lemma of the given word occurred in one of the lists, the feature was assigned the value *true*, else it was *false*.
- **Morphological features:** for each word, its part of speech and lemma were noted. As mentioned before, modality and mood are morphologically expressed in Hungarian (e.g. in *csinálhatnánk* do-MOD-COND-1PL "we could do", the suffix *-hat* refers to modality and the suffix *-ná* refers to conditional) hence for each verb, it was investigated whether it had a modal suffix, whether it was in the conditional mood and whether its form was first person plural or third person plural as these two latter verbal forms are typical instances of expressing generic phrases or generalizations in Hungarian, which are related to weasels. For each noun, its number (i.e. singular/plural) was marked as feature. For each pronoun, we checked whether it was an indefinite one since indefinite pronouns like *valaki* "someone" or *valamilyen* "some" are often used as weasel cues. For each adjective, we marked whether it was comparative or superlative as they can occur as peacock cues.
- **Syntactic features:** for each word, its dependency label was marked. For each noun, it was checked whether it had a determiner as determinerless nouns may be used as weasels in Hungarian. For each verb, it was checked whether it had a subject[1].
- **Semantic/pragmatic features:** we manually compiled a list of speech act verbs in Hungarian and checked whether the given verb was one of them. Besides, we translated lists of English words with

[1]Hungarian is a pro-drop language, hence the subject is not obligatorily present in the clause. Moreover, applying a third person plural verb without a subject is a common way to express generalization in Hungarian, which is one typical strategy of weasels.

positive and negative content developed for sentiment analysis (Liu, 2012) and checked whether the lemma of the given word occurred in these lists.

As contextual features for each word, we applied as features the POS tags and dependency labels of words within a window of size two. Although earlier research on English uncertainty detection mostly made use of orthographical, morphological and syntactic information (see e.g. Szarvas et al. (2012)), here we included some new feature types in our feature set, namely, pragmatic and semantic features.

Based on this feature set, we carried out our experiments. Since only 3% of the tokens in the corpus function as uncertainty cues, it seemed necessary to filter the training database: half of the cueless sentences were randomly selected and deleted from the training dataset. Moreover, as there were only 44 investigation cues in the data, we omitted this class from training and evaluation as well, due to sparseness problems.

First, we applied ten-fold cross validation on the corpus. Since we had two domains of texts at hand, it enabled us to experiment with the two domains separately as well: ten-fold cross validation was carried out for both domains individually and we also made use of cross-domain settings, where one of the domains was used as the training database but the evaluation was performed on the other domain. For evaluation, we used the metrics precision, recall and F-score. The results of our experiments will be presented in Section 4.

3.3 Baseline Methods

As a baseline, we applied a simple dictionary lookup method. Lists mentioned among the lexical features were utilized here: whenever the lemma of the given word matched one of the words in the list, we tagged it as an uncertainty cue of the type determined by the given list.

4 Results

Table 2 shows the results of the baseline and machine learning experiments on the hUnCertainty corpus, obtained by ten-fold cross validation.

	Dictionary lookup			**Machine learning**			**Difference**		
Type	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score
Weasel	18.12	35.92	24.09	52.48	30.73	38.76	+34.37	-5.19	+14.68
Hedge	55.10	32.42	40.82	61.26	48.94	54.41	+6.17	+16.52	+13.59
Peacock	21.66	30.77	25.42	32.61	11.88	17.41	+10.95	-18.89	-8.01
Epistemic	42.46	30.02	35.18	63.18	34.07	44.27	+20.72	+4.04	+9.09
Doxastic	29.30	46.16	35.85	52.42	46.26	49.15	+23.12	+0.10	+13.30
Condition	31.73	62.90	42.18	51.41	25.80	34.35	+19.68	-37.10	-7.83
Micro P/R/F	29.09	35.74	32.07	55.95	37.46	44.87	+26.86	+1.72	+12.80

Table 2: Results on the hUnCertainty corpus.

The results of the machine learning approach have outperformed those achieved by the baseline dictionary lookup method, except for two classes. This is primarily due to better precision, which has grown for each uncertainty category in the case of sequence labeling. However, recall values are more diverse: for hedges and epistemic cues, it has grown, for doxastic cues it has not changed significantly, but for peacocks and conditional cues we can see a serious decrease. The low recall values might be the reason why the F-score obtained by the dictionary lookup method is higher than the one obtained by machine learning in the case of peacocks and conditionals.

We also experimented separately on the two domains. Table 3 shows those on the news subcorpus, whereas Table 4 shows the results achieved on the Wikipedia subcorpus.

In both domains, we can observe that machine learning methods outperform the baseline dictionary lookup method, except for the peacock and conditional cue classes. However, there are domain differences in the results. First, weasels seem to be much hard to detect in the news subcorpus than in the

	Dictionary lookup			Machine learning			Difference		
Type	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score
Weasel	3.24	17.83	5.48	37.50	15.12	21.55	+34.26	-2.71	+16.06
Hedge	53.61	39.05	45.18	61.55	49.69	54.99	+7.94	+10.64	+9.80
Peacock	13.82	31.91	19.29	47.06	8.51	14.41	+33.23	-23.40	-4.88
Epistemic	31.90	20.67	25.08	56.63	39.39	46.46	+24.73	+18.72	+21.37
Doxastic	33.50	37.61	35.43	57.05	51.83	54.32	+23.55	+14.23	+18.88
Condition	35.27	57.03	43.58	54.39	24.22	33.51	+19.12	-32.81	-10.07
Micro P/R/F	23.21	34.17	27.65	57.31	41.93	48.43	+34.10	+7.76	+20.78

Table 3: Results on the news subcorpus.

	Dictionary lookup			Machine learning			Difference		
Type	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score
Weasel	26.03	38.50	31.06	59.26	34.74	43.80	+33.23	-3.76	+12.74
Hedge	55.86	29.92	38.97	64.59	50.02	56.38	+8.73	+20.10	+17.41
Peacock	23.29	30.63	26.46	37.85	13.8	20.22	+14.56	-16.83	-6.24
Epistemic	49.57	37.34	42.59	63.95	36.03	46.09	+14.38	-1.31	+3.50
Doxastic	25.24	65.20	36.40	54.31	33.54	41.47	+29.07	-31.66	+5.07
Condition	29.66	67.74	41.26	47.12	31.61	37.84	+17.46	-36.13	-3.42
Micro P/R/F	32.28	36.40	34.21	59.70	37.5	46.06	+27.42	+1.10	+11.85

Table 4: Results on the Wikipedia subcorpus.

Wikipedia subcorpus (21.55 vs. 43.8 in terms of F-score). Second, peacocks are also harder to detect in the news subcorpus (F-scores of 14.41 vs. 20.22). Third, there is a considerable gap between the recall scores in the case of doxastic cues: in the Wikipedia subcorpus, the dictionary lookup method outperforms CRF (the difference is 36.13 percentage points) but in the news subcorpus, CRF achieves higher recall with 14.23 percentage points.

To further explore domain differences, we carried out some cross validation experiments. First, we trained our CRF model on the Wikipedia domain and then evaluated it on the news domain. Later, the model was trained on the news domain and evaluated on the Wikipedia domain. Tables 5 and 6 present the results, respectively, contrasted to the results achieved in the indomain settings. It is also striking that although the gain in micro F-score is almost the same in the two settings, the biggest difference can be observed for semantic uncertainty classes in the case of the Wikipedia → news setting, while the difference is much bigger for discourse-level uncertainty types in the news → Wikipedia setting.

	Cross validation			Indomain ten fold			Difference		
Type	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score
Weasel	17.53	19.77	18.58	37.50	15.12	21.55	+19.97	-4.65	+2.97
Hedge	57.40	39.30	46.66	61.55	49.69	54.99	+4.15	+10.39	+8.33
Peacock	22.81	13.83	17.22	47.06	8.51	14.41	+24.25	-5.32	-2.80
Epistemic	50.00	16.76	25.10	56.63	39.39	46.46	+6.63	+22.63	+21.35
Doxastic	46.63	10.70	17.41	57.05	51.83	54.32	+10.43	+41.13	+36.91
Condition	62.96	26.56	37.36	54.39	24.22	33.51	-8.58	-2.34	-3.85
Micro P/R/F	44.48	23.35	30.62	57.31	41.93	48.43	+12.83	+18.58	+17.81

Table 5: Cross-domain results: Wikipedia → news.

As some uncertainty detectors aim at identifying uncertain sentences only, that is, they handle the task at the sentence level and do not pay attention to the detection of individual cues (Medlock and Briscoe, 2007), we also applied a more relaxed evaluation metric. If at least one of the tokens within the sentence was labeled as an uncertainty cue – regardless of its type –, the sentence was considered as uncertain.

	Cross validation			Indomain ten fold			Difference		
Type	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score
Weasel	71.26	6.87	12.53	59.26	34.74	43.8	-12.00	+27.87	+31.27
Hedge	63.48	26.33	37.22	64.59	50.02	56.38	+1.11	+23.69	+19.16
Peacock	43.14	5.57	9.87	37.85	13.80	20.22	-5.29	+8.23	+10.35
Epistemic	78.65	30.57	44.03	63.95	36.03	46.09	-14.70	+5.46	+2.06
Doxastic	39.55	33.23	36.12	54.31	33.54	41.47	+14.76	+0.31	+5.35
Condition	47.31	28.39	35.48	47.12	31.61	37.84	-0.19	+3.22	+2.36
Micro P/R/F	59.98	18.00	27.68	59.7	37.5	46.06	-0.28	+19.50	+18.38

Table 6: Cross-domain results: news → Wikipedia.

Results on the identification of uncertain sentences are summarized in Table 7, in terms of precision, recall and F-score. It is revealed that here there are no sharp differences in performance as far as the indomain settings are concerned since the system can achieve an F-score of about 70 in both domains and on the whole corpus as well. However, in the cross-domain settings lower precision values and F-scores can be observed, while recall values basically remain the same with regard to the indomain settings.

Evaluation setting	Precision	Recall	F-score
hUnCertainty 10 fold	62.20	78.06	69.23
News 10 fold	67.38	78.01	72.30
Wikipedia 10 fold	60.32	80.05	68.80
Wikipedia → news	45.88	74.21	56.70
News → Wikipedia	35.73	84.61	50.24

Table 7: Machine learning results at the sentence level.

5 Discussion

Our results prove that a sequence labeling approach can be efficiently used for the automatic identification of uncertainty cues in Hungarian texts. With our baseline dictionary lookup method, the best results were achieved on the epistemic, conditional and hedge cues while the sequence labeling approach was the most successful on the hedge, epistemic and doxastic cues. All of this indicates that hedge and epistemic cues are the easiest to detect. On the other hand, uncertainty types where there was a small difference between the results achieved by the two approaches (for instance, semantic uncertainty cues in the Wikipedia subcorpus) are mostly expressed by lexical means and these cues are less ambiguous. In this setting, the detection of discourse-level uncertainty categories, however, profits more from machine learning, which is most probably due to the fact that here context (discourse) plays a more important rule hence a sequence labeling algorithm is more appropriate for the task, which takes into account contextual information as well.

In the case of peacocks and conditional cues the sequence labeling approach obtained worse results than dictionary lookup: in each case, precision got higher but recall seriously decreased. This suggests that these classes highly rely on lexical features and our machine learning system needs further improvement, with special regard to specific (lexical) features defined for these uncertainty categories.

As for domain differences, we found that the distribution of uncertainty cues differs in the two subcorpora, weasels being more frequent in Wikipedia whereas doxastic cues are more probable to occur in the news subcorpus. Domain differences concerning weasels and doxastic cues are highlighted in the cross domain experiments as well. When the training dataset contains fewer cues of the given uncertainty type, the performance falls back on the target domain: when trained on the news subcorpus, an F-score of 12.53 can be obtained for weasels in the Wikipedia subcorpus, which is 31.27 points less than the indomain results. Similarly, an F-score of 17.41 can be obtained for doxastic cues in the news domain

when Wikipedia is used as the training set but the indomain setting yields an F-score of 54.32.

All of the above facts may be related to the characteristics of the texts. Weasels are sourceless propositions and in the news media, it is indispensable to know who the source of the news is, thus, pieces are usually reported with their source provided and so, propositions with no explicit source (i.e. weasels) occur rarely in the news subcorpus. On the other hand, doxastic cues are related to beliefs and the news subcorpus consists of criminal news (mostly related to murders). When describing the possible reasons behind each criminal act, phrases that refer to beliefs and mental states are often used and thus this type of uncertainty is likely to be present in such pieces of news but not in Wikipedia articles.

In the cross domain experiments, indomain results outperform those obtained by the cross domain models. The difference in performance is significant (t-test, $p = 0.042$ for the news subcorpus and $p = 0.0103$ for the Wikipedia subcorpus). That is, the choice of the training dataset significantly affects the results, which indicates that there really are domain differences in uncertainty detection. There are only two exceptions that do not correspond to these tendencies: the peacock and conditional cues in the Wikipedia $\rightarrow$ news setting. The reason why a model trained on a different domain can perform better might lie in the size of the subcorpora. The Wikipedia domain contains much more peacock cues than the news domain and although the domains are different, training on a dataset with more cue instances seems to be beneficial for the results.

If we evaluate the models' performance at the sentence level rather than at the cue level, it can be observed that better results can be achieved, especially with regard to recall values. One reason for that may be that a single uncertain sentence may include more than one cues and should one of them be missed, it does not seriously harm performance (in case at least one cue per sentence is correctly detected).

If our results are compared to those achieved on semantic uncertainty cues found in English Wikipedia articles (Szarvas et al., 2012), it can be seen that the task seems to be somewhat easier in English than in Hungarian: that paper reports F-scores from 0.6 to 0.8. One possible reason for this is that there are typological differences between English and Hungarian and so, uncertainty marking is rather lexically determined in English but in Hungarian, morphology also plays an essential role. For instance, the modal suffixes *-hat/-het* correspond to the auxiliaries *may* and *might* and while in English they function as separate lexical items, in Hungarian they are always attached to the verbal stem and never occur on their own. This is reflected in the number of different cues as well: in the English dataset, there are 166 different semantic cues while in Hungarian, there are 319 (and note that the Hungarian corpus is about half of the size of the English one). As such, applying the word form or the lemma as features may result in relatively high F-scores in English, where the word form itself denotes uncertainty, but these features are less effective in Hungarian without any morphological features included. Another language-specific feature is that Hungarian is a pro-drop language, so in some cases, the pronominal subject may be omitted from the sentence. Subjectless sentences are a typical strategy in Hungarian to express sourceless statements (weasels), but the subject can be deleted due to syntactic ellipsis as well, thus distinguishing between subjectless sentences that denote uncertainty and those that do not is a special task in Hungarian uncertainty detection.

The outputs of the machine learning system were further investigated, in order to find the most typical errors our system made. It was revealed that the most problematic issue was the disambiguation of ambiguous cues. For instance, the words *számos* "several" or *sok* "many" may function as hedges or weasels, or *nagy* "big" may be a hedge or a peacock, depending on the context. Such cues were often misclassified by the system. Another common source of errors was that some cues have non-cue meanings as well, like the verb *tart*, which can be a doxastic cue with the meaning "think" but when it means "keep", it is not uncertain at all. The identification of epistemic cues that include negation words was also not straightforward: multiword cues such as *nem zárható ki* "it cannot be excluded" or *nem tudni* "it is not known" were not marked as cues by the system.

6 Conclusions

In this paper, we presented the first results on Hungarian uncertainty detection. For this, we made use of a manually annotated corpus, which contains texts from two domains: Wikipedia articles and pieces of news from a news portal. We contrasted the cue distribution in the two domains and we also experimented with uncertainty detection. For this purpose, we applied a supervised machine learning approach, which was based on sequence labeling and exploited a rich feature set. We reported the first results on uncertainty detection for Hungarian, which also prove that the performance on uncertainty detection is influenced by the domain of the texts. We hope that this study will enhance research on uncertainty detection for languages other than English.

In the future, we would like to improve our methods, especially in order to achieve better recall at the cue level. Furthermore, we would like to investigate domain specificities in more detail and we would also like to carry out some domain adaptation experiments as well.

Acknowledgments

This research was supported by the European Union and the State of Hungary, co-financed by the European Social Fund in the framework of TÁMOP-4.2.4.A/2-11/1-2012-0001 "National Excellence Program".

References

Wendy W. Chapman, David Chu, and John N. Dowling. 2007. Context: An algorithm for identifying contextual features from clinical text. In *Proceedings of the ACL Workshop on BioNLP 2007*, pages 81–88.

Noa P. Cruz Díaz. 2013. Detecting negated and uncertain information in biomedical and review texts. In *Proceedings of the Student Research Workshop associated with RANLP 2013*, pages 45–50, Hissar, Bulgaria, September. RANLP 2013 Organising Committee.

Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. 2010. The CoNLL-2010 Shared Task: Learning to Detect Hedges and their Scope in Natural Language Text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning (CoNLL-2010): Shared Task*, pages 1–12, Uppsala, Sweden, July. Association for Computational Linguistics.

Viola Ganter and Michael Strube. 2009. Finding Hedges by Chasing Weasels: Hedge Detection Using Wikipedia Tags and Shallow Linguistic Features. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 173–176, Suntec, Singapore, August. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9(Suppl 10).

Natalia Konstantinova, Sheila C.M. de Sousa, Noa P. Cruz, Manuel J. Mana, Maite Taboada, and Ruslan Mitkov. 2012. A review corpus annotated for negation, speculation and their scope. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML-01, 18th Int. Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann.

Marc Light, Xin Ying Qiu, and Padmini Srinivasan. 2004. The language of bioscience: Facts, speculations, and statements in between. In *Proc. of the HLT-NAACL 2004 Workshop: Biolink 2004, Linking Biological Literature, Ontologies and Databases*, pages 17–24.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

Andrew Kachites McCallum. 2002. *MALLET: A Machine Learning for Language Toolkit*. http://mallet.cs.umass.edu.

Ben Medlock and Ted Briscoe. 2007. Weakly Supervised Learning for Hedge Classification in Scientific Literature. In *Proceedings of the ACL*, pages 992–999, Prague, Czech Republic, June.

Roser Morante and Caroline Sporleder. 2012. Modality and negation: An introduction to the special issue. *Computational Linguistics*, 38:223–260, June.

Roser Morante, Vincent van Asch, and Antal van den Bosch. 2009. Joint memory-based learning of syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL*, pages 25–30.

Raheel Nawaz, Paul Thompson, and Sophia Ananiadou. 2010. Evaluating a meta-knowledge annotation scheme for bio-events. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing*, pages 69–77, Uppsala, Sweden, July. University of Antwerp.

Arzucan Özgür and Dragomir R. Radev. 2009. Detecting speculations and their scopes in scientific text. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1398–1407, Singapore, August. Association for Computational Linguistics.

Victoria L. Rubin, Elizabeth D. Liddy, and Noriko Kando. 2005. Certainty identification in texts: Categorization model and manual tagging results. In J.G. Shanahan, J. Qu, and J. Wiebe, editors, *Computing attitude and affect in text: Theory and applications (the information retrieval series)*, New York. Springer Verlag.

Victoria L. Rubin. 2010. Epistemic modality: From uncertainty to certainty in the context of information seeking as interactions with texts. *Information Processing & Management*, 46(5):533–540.

Roser Saurí and James Pustejovsky. 2009. FactBank: a corpus annotated with event factuality. *Language Resources and Evaluation*, 43:227–268.

Burr Settles, Mark Craven, and Lewis Friedland. 2008. Active learning with real annotation costs. In *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, pages 1–10.

Hagit Shatkay, Fengxia Pan, Andrey Rzhetsky, and W. John Wilbur. 2008. Multi-dimensional classification of biomedical text: Toward automated, practical provision of high-utility text to diverse users. *Bioinformatics*, 24(18):2086–2093.

György Szarvas, Veronika Vincze, Richárd Farkas, György Móra, and Iryna Gurevych. 2012. Cross-genre and cross-domain detection of semantic uncertainty. *Computational Linguistics*, 38:335–367, June.

Özlem Uzuner, Xiaoran Zhang, and Tawanda Sibanda. 2009. Machine Learning and Rule-based Approaches to Assertion Classification. *Journal of the American Medical Informatics Association*, 16(1):109–115, January.

Veronika Vincze, György Szarvas, Richárd Farkas, György Móra, and János Csirik. 2008. The BioScope Corpus: Biomedical Texts Annotated for Uncertainty, Negation and their Scopes. *BMC Bioinformatics*, 9(Suppl 11):S9.

Veronika Vincze. 2013. Weasels, Hedges and Peacocks: Discourse-level Uncertainty in Wikipedia Articles. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 383–391, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Zhongyu Wei, Junwen Chen, Wei Gao, Binyang Li, Lanjun Zhou, Yulan He, and Kam-Fai Wong. 2013. An empirical study on uncertainty identification in social media context. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 58–62, Sofia, Bulgaria, August. Association for Computational Linguistics.

Theresa Ann Wilson. 2008. *Fine-grained Subjectivity and Sentiment Analysis: Recognizing the Intensity, Polarity, and Attitudes of Private States*. Ph.D. thesis, University of Pittsburgh, Pittsburgh.

János Zsibrita, Veronika Vincze, and Richárd Farkas. 2013. magyarlanc: A Toolkit for Morphological and Dependency Parsing of Hungarian. In *Proceedings of RANLP-2013*, pages 763–771, Hissar, Bulgaria.

Rediscovering Annotation Projection for Cross-Lingual Parser Induction

Jörg Tiedemann
Uppsala University
Department of Linguistics and Philology
firstname.lastname@lingfil.uu.se

Abstract

Previous research on annotation projection for parser induction across languages showed only limited success and often required substantial language-specific post-processing to fix inconsistencies and to lift the performance onto a useful level. Model transfer was introduced as another quite successful alternative and much research has been devoted to this paradigm recently. In this paper, we revisit annotation projection and show that the previously reported results are mainly spoiled by the flaws of evaluation with incompatible annotation schemes. Lexicalized parsers created on projected data are especially harmed by such discrepancies. However, recently developed cross-lingually harmonized annotation schemes remove this obstacle and restore the abilities of syntactic annotation projection. We demonstrate this by applying projection strategies to a number of European languages and a selection of human and machine-translated data. Our results outperform the simple direct transfer approach by a large margin and also pave the road to cross-lingual parsing without gold POS labels.

1 Introduction

Linguistic resources and tools exist only for a minority of the world's languages. However, many NLP applications require robust tools and the development of language-specific resources is expensive and time consuming. Many of the common tools are based on data-driven techniques and they often require strong supervision to achieve reasonable results for real world applications. Fully unsupervised techniques are not a good alternative yet for tasks like data-driven syntactic parsing and, therefore, cross-lingual learning has been proposed as a possible solution to quickly create initial tools for otherwise unsupported languages (Ganchev and Das, 2013).

In syntactic parsing, two main strategies have been explored in cross-lingual learning: annotation projection and model transfer. The first strategy relies on parallel corpora and automatic word alignment that make it possible to map linguistics annotation from a source language to a new target language (Yarowsky et al., 2001; Hwa et al., 2005; Täckström et al., 2013a). The basic idea is that existing tools and models are used to process the source side of a parallel corpus and that projection heuristics guided by alignment can be used to transfer the automatic annotation to the target language text. Using the projected annotation assuming that it is sufficiently correct, models can then be trained for the target language. However, directly projecting syntactic structure results in a rather poor performance when applied to resources that were developed separately for individual languages (Hwa et al., 2005). Extensive additional post-processing in form of transformation rules is required to achieve reasonable scores. Furthermore, incompatible tagsets make it impossible to directly transfer labeled annotation to a new language and previous literature on cross-lingual parsing via annotation projection is, therefore, bound to the evaluation of unlabeled attachment scores (UAS). Less frequent, but also possible, is the scenario where the source side of the corpus contains manual annotation (Agić et al., 2012). This addresses the problem created by projecting noisy annotations, but it presupposes parallel corpora with manual annotation, which are rarely available. Additionally, the problem of incompatible annotation still remains.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1854–1864, Dublin, Ireland, August 23-29 2014.

The second strategy, model transfer instead relies on universal features and the transfer of model parameters from one language to another. The main idea is to reduce the need of language-specific information, e.g. using delexicalized parsers that ignore lexical information. Drawing from a harmonized POS tagset (Petrov et al., 2012), transfer models have been used for a variety of languages. The advantage over annotation projection approaches is that no parallel data is required (at least in the basic settings) and that training can be performed on gold standard annotation. However, it requires a common feature representation across languages (McDonald et al., 2013), which can be a strong bottleneck. There are also several extensions to improve the performance of transfer models. One idea is to use multiple source languages to increase the statistical ground for the learning process (McDonald et al., 2011; Naseem et al., 2012), a strategy that can also be used in the case of annotation projection. Another idea is to enhance models by cross-lingual word clusters (Täckström et al., 2012) and to use target language adaptation techniques with prior knowledge of language properties and their relatedness when using multiple sources in training (Täckström et al., 2013b). Based on the success of these techniques, model transfer has dominated recent research on cross-lingual learning.

In this paper, we return to annotation projection as a powerful tool for porting syntactic parsers to new languages. Building on the availability of cross-lingually harmonized data sets, we show that projection performs well and outperforms direct transfer models by a large margin in contrast to previous findings on projection with incompatible treebanks. In the following, we first revisit the projection algorithms proposed earlier and discuss issues with transferring labels across languages. After that we report experimental results with various settings using human translations and machine-translated data. Finally, we also look at parsing results without gold standard POS labeling, which is ultimately required when porting parsers to new languages that lack appropriate resources.

2 Syntactic Annotation Projection

Hwa et al. (2005) propose a direct projection algorithm for syntactic dependency annotation. The algorithm defines several heuristics to map source side annotations to target languages using word alignments in a parallel corpus. The main difficulties with the projection arise with none-one-to-one links and unaligned tokens. Each of the following alignment types are adressed by the algorithm separately:

one-to-one: Copy relations $R(s_i, s_j)$ between source words s_i and s_j to relations $R(t_x, t_y)$ if s_i is aligned to t_x and s_j is aligned to t_y and nothing else.

unaligned source: Create an empty (dummy) word in the target language sentence that takes all relations (incoming and outgoing arcs) of the unaligned source language word.

one-to-many: Create an empty target word t_z that acts as the parent of all aligned target words $t_x, .., t_y$. Remove the alignments between s_i and $t_x, .., t_y$ and align s_i to the new empty word t_z instead.

many-to-one: Delete all alignments between $s_i, .., s_j$ and t_x except the link between the head of $s_i, .., s_j$ and t_x.

many-to-many: Perform the rule for one-to-many alignments first and then perform the rule for many-to-one alignments.

unaligned target: Remove all unaligned target words.

In contrast to Hwa et al. (2005), we are also interested in labeled attachment and the projection of POS annotation. Therefore, we copy labels through the alignment using the heuristics listed above. Figure 1 illustrates some of the cases discussed. There are some important implications due to the treatment of complex alignment types. The direct projection algorithm frequently creates dummy nodes and relations that have no correspondence in the source language. Here, we need to make some decisions on how to project the annotation from source to target sentences.

First of all, we decided to name all additional tokens created by the algorithm with the same string *DUMMY*. An alternative would be to invent unique names for each newly created token within each sentence but this would blow up the vocabulary and would not add useful information to the data.

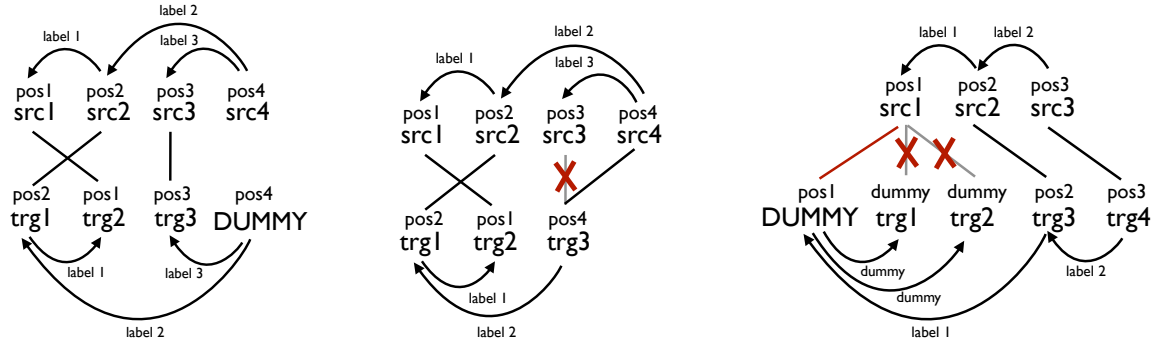


Figure 1: Annotation projection heuristics for special alignment types: Unaligned source words (left image), many-to-one alignments (center), one-to-many alignments (right image).

The second problem is related to the auxiliary relations that are created when treating one-to-many alignments. In these cases, multiple words are attached to newly created dummy nodes. However, no corresponding labels exist in the source language that would allow us to infer appropriate labels for these additional attachments. One possibility would be to use a specific label from the existing set of dependency relations, for example 'mwe'. However, one-to-many alignments do not always refer to proper multi-word expressions but often represent other grammatical or structural differences like the relation between the English preposition *'of'* which is linked together with the determiner *'the'* to the German determiner *'der'* in sentences like *'Resumption OF THE session'* translated to German *'Wiederaufnahme DER Sitzung'*. Therefore, we decided to label these additional dependency with a new unique label *dummy* instead of selecting an existing one.

Yet another problem arises with the projection of POS annotation. Similar to the labeling of dependency relations, we have to decide how to transfer POS tags to the target language in cases of one-to-many alignments. In our implementation, we transfer the source language label only to the newly created dummy node which dominates all target language words linked to the source language word in the projected dependency tree. The daughter nodes, however, obtain the label *dummy* even as their POS annotation. Alternatively, we may project the POS tag to all linked tokens according to the original alignment but our guiding principle is to resolve link ambiguity first using the heuristics in the direct projection algorithm and then to transfer annotation.

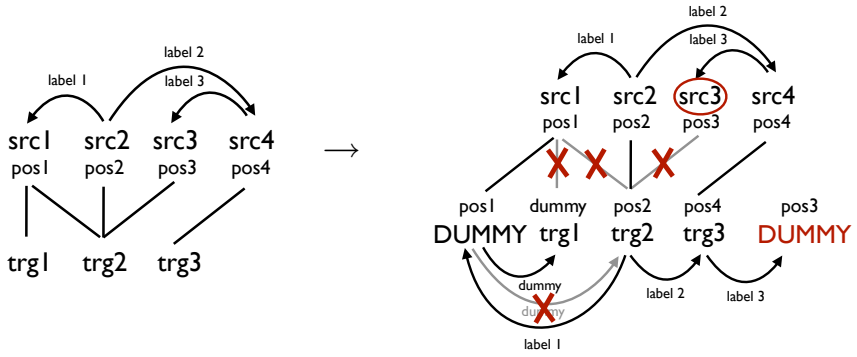


Figure 2: A complex example for annotation projection with many-to-many word alignments.

Finally, we also need to look at the interaction between the various projection heuristics. Figure 2 illustrates a complex case with many-to-many word alignments. Resolving the alignment ambiguity is not entirely straightforward. In our implementation, we start by looking at all one-to-many alignments and resolve them according to the definitions of the projection algorithm. In our example, this creates a *DUMMY* node that dominates target words *trg1* and *trg2* and links between *src1* and (*trg1*,*trg2*) are deleted. We label the new relations with *dummy*. The next step considers many-to-one alignments,

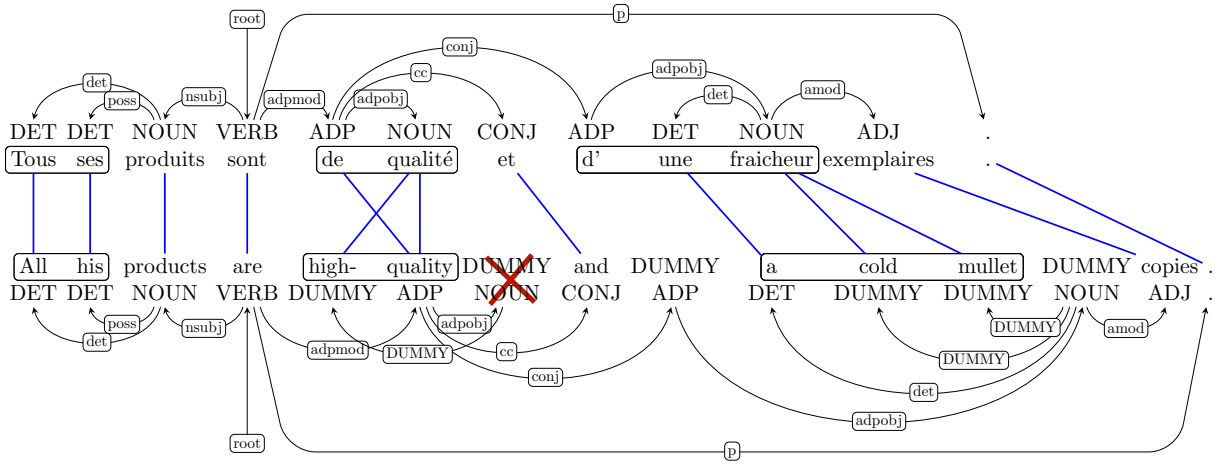


Figure 3: A complete projection example from a translated treebank including transitive relations over a *DUMMY* node that can safely be collapsed (which also removes the non-projectivity of the projected tree). The resulting relation between *quality* and *high-* will be labeled as *adpobj*. Note that projection errors appear due to the ambiguous alignments between *de qualité* and *high- quality*. Boxes indicate phrases that are translated as units by the SMT engine.

which, using the remaining links, is source words (*src2*,*src3*) aligned to *trg2*. According to the algorithm we delete the link between *src3* and *trg2* (because *src2* dominates *src3* in the source language tree) and proceed. This, however, creates an unaligned source language word (*src3*), which we treat in the next step. The unaligned token gives rise to the second *DUMMY* word, which is attached to *trg3* as the result of the alignment between *src4* and *trg3* and the relation between *src4* and *src3*. Finally, we can map all other relations according to the one-to-one alignment rule. This, however, creates a conflict with the already existing *dummy* relation between the first *DUMMY* word and *trg2*. Mapping according to the one-to-one rule turns the relation around and attaches the *DUMMY* word to *trg2* and labels the relation with *label 1*. Now, we could remove the second *DUMMY* node according to the rule about unaligned target language words. However, this rule should not apply to these special nodes as they may play a crucial role to keep elements connected in the final target language tree.

Another difficult case, which is not illustrated here, is when many-to-one alignments need to be resolved but the aligned source language words are siblings in the syntactic tree and no unique head can be identified. In our implementation, we randomly pick a node but more linguistically informed guesses would probably be better. Yet another difficult decision is the placement of the *DUMMY* nodes. We decided to put them next to the head node they attach to. Other heuristics are possible and all placements greatly influence the projectivity of the resulting tree structure. One final adjustment that we apply is the removal of unary productions over *DUMMY* nodes. We collaps all relations that run with single attachments via *DUMMY* nodes to reduce the number of these uninformative tokens. This may also have positive effects on projectivity as we can see in the example in Figure 3.

3 Machine-Translated Treebanks

Another strategy for annotation projection is based on automatic translation. Machine translation models can be used to create synthetic parallel data for projecting annotations from one language to another (Tiedemann et al., 2014). Recent advances in machine translation (MT) are now making this a realistic alternative. The use of direct treebank translation instead of existing parallel corpora has several important advantages. First of all, we skip the use of an error-prone annotation step when producing the source language side of the training data. Starting with a noisy source language annotation, we accumulate two sources of errors in annotation projection. However, with direct translation we can start with the gold standard annotation provided in the original treebank. Furthermore, we avoid problems of domain shifts which is typically the case when applying a parser trained on one domain to texts (a parallel corpus in

DELEXICALIZED	DE	EN	ES	FR	SV
DE	62.71	43.20	46.09	46.09	50.64
EN	46.62	77.66	55.65	56.46	**57.68**
ES	44.03	46.73	68.21	**57.91**	53.82
FR	43.91	46.75	**59.65**	67.51	52.01
SV	**50.69**	**49.13**	53.62	51.97	70.22

MCDONALD ET AL. (2013)	DE	EN	ES	FR	SV
DE	64.84	47.09	48.14	49.59	53.57
EN	48.11	78.54	56.86	58.20	**57.04**
ES	45.52	47.87	70.29	**63.65**	53.09
FR	45.96	47.41	**62.56**	73.37	52.25
SV	**52.19**	**49.71**	54.72	54.96	70.90

Table 1: Baselines – labeled attachment score (LAS) for delexicalized transfer parsing; results of McDonald et al. (2013) included for reference.

our case) coming from another domain. Finally, we can also assume that machine translation produces output which is closer to the original text than most human translations will be in any parallel corpus. Even if this may sound as a disadvantage, for projection this is preferred. Being close to the original source makes it easier to map annotation from one language to another as we expect a lower degree of grammatical and structural divergences that originate in the linguistic freedom human translators can apply. Furthermore, common statistical MT models inherently provide alignments between words and phrases, which removes the requirement to apply yet another error-prone alignment step on the parallel data. In the experiments below we, therefore, explore the translation strategy as yet another way of applying annotation projection.

4 Experiments

In the following, we show our experimental results using annotation projection in several cross-lingual scenarios. However, we start by presenting a delexicalized baseline, which is, to our knowledge, the only previous model that has been presented for labeled dependency parsing across languages using the recently created Universal Treebank. We will use this baseline as reference point even though our projection models are not directly comparable with delexicalized direct transfer models. Note that all results below are computed on the held-out test data sections of the Universal Treebank if not stated otherwise.

4.1 Delexicalized Baselines

McDonald et al. (2013) present the Universal Treebank that comes with a harmonized syntactic annotation scheme across six languages. This data set enables cross-lingual learning of labeled dependency parsing models. McDonald et al. (2013) propose delexicalized models as a simple baseline for model transfer and present encouraging labeled attachment scores (LAS) especially for closely related languages. As a reference, we have created similar baseline models using the same data set but a slightly different setup, which is compatible with the experiments we present later. Table 1 summarizes the scores in terms of LAS for all language pairs in the data set.[1] In our setup, we apply MaltParser (Nivre et al., 2006) and optimize feature models and learning parameters using MaltOptimizer (Ballesteros and Nivre, 2012). For all cross-lingual experiments (columns represent target languages we test on), we always use the same feature model and parameters as we have found for the source language treebank. Contrasting our models with the scores from McDonald et al. (2013), we can see that they are comparable with some differences that are due to the tools and learning parameters they apply which are along the lines of Zhang and Nivre (2011).

4.2 Annotation Projection with Human Translations

Our first batch of projection experiments considers parallel data taken from the well-known Europarl corpus, which is frequently used in research on statistical machine translation (SMT). It contains large quantities of translated proceedings from the European Parliament for all but one language (namely

[1]Note that we include punctuation in our evaluation. Ignoring punctuation leads to slightly higher scores but we do not report those numbers here.

UAS on CoNLL data

	DE	EN	ES	SV
DE	–	41.60	47.89	**58.80**
EN	49.67	–	51.44	58.66
ES	46.14	37.78	–	52.53
SV	**57.99**	**51.57**	**57.25**	–

UAS on Universal Treebank data

	DE	EN	ES	SV
DE	–	56.21	65.18	70.27
EN	63.17	–	68.02	70.40
ES	61.98	56.16	–	**71.06**
SV	**64.78**	**58.93**	**69.15**	–

Table 2: Unlabeled attachment scores for projected treebank models; comparing CoNLL data to Universal Treebank data for evaluation.

Korean) that are included in the Universal Treebank v1. The entire corpus (version 7) contains over two million sentences in each language and we use increasing amounts of the corpus to investigate the impact on cross-lingual parser induction. The corpus comes with automatic sentence alignments and is quite clean with respect to translation quality and sentence alignment accuracy. It is, therefore, well suited for our initial experiments with annotation projection even though the domain does not necessarily match the one included in the treebank test sets.

Another important prerequisite for annotation projection is word alignment. Following the typical setup, we rely on automatic word alignment produced by models developed for statistical machine translation. Similar to Hwa et al. (2005), we apply GIZA++ (Och and Ney, 2003) to align the corpus for all language pairs in all translation directions using IBM model 4 Viterbi alignments. In contrast to Hwa et al. (2005), we then use symmetrization heuristics to combine forward and backward alignments, which is common practice in the SMT community. In particular, we apply the popular grow-diag-final-and heuristics as implemented in the Moses toolbox (Koehn et al., 2007).

Let us first look at unlabeled attachment scores to compare results that can be achieved with harmonized annotation in contrast to the ones that we can see on the cross-lingually incompatible data from the CoNLL shared task (Buchholz and Marsi, 2006). Table 2 lists the scores that we obtain when applying our implementation of the direct projection algorithm.[2] As expected, the performance on the CoNLL data is rather poor, which confirms the findings of Hwa et al. (2005) even though our scores are significantly above their results without post-correction. The scores on the Universal Treebank data, however, are up to about 20 UAS points higher than the corresponding results on CoNLL data but without any of the extensive post-processing transformations proposed by Hwa et al. (2005).

LAS on Universal Treebank data

	DE	EN	ES	FR	SV
DE	–	49.44	56.58	58.75	61.04
EN	**56.59**	–	60.07	62.78	**62.15**
ES	54.04	47.90	–	**65.50**	61.45
FR	53.93	**51.23**	**65.03**	–	58.71
SV	56.13	49.18	60.82	62.00	–

data set: 40,000 sentences

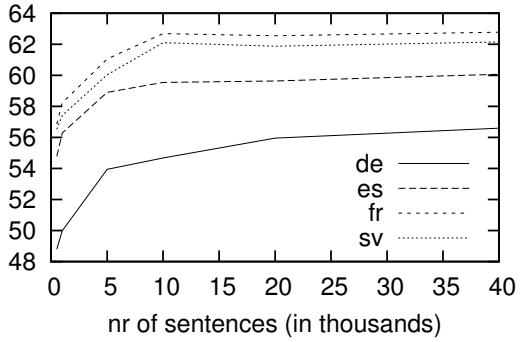


Figure 4: Annotation projection on Europarl data: LAS for induced parser models. The Figure to the right plots the learning curves for increasing training data for projections from English to the other languages.

Moreover, the real power of the harmonized annotation in the Universal Treebank comes from the possibility to obtain attachment labels. The table in Figure 4 shows the labeled attachment scores obtained for training on 40,000 sentences[3] of each language pair. Next to the table in Figure 4 we also show the

[2]We leave out French in this comparison as there is no French treebank in the CoNLL data.

[3]Note that there may be repeated sentences in the data.

with original source side annotation					
	DE	EN	ES	FR	SV
DE	–	53.02	54.96	58.20	59.65
EN	52.93	–	61.25	64.58	**63.82**
ES	50.88	50.28	–	**66.17**	60.48
FR	50.46	**53.95**	**65.46**	–	59.05
SV	**53.69**	51.51	60.58	60.19	–

jackknifing for source side annotation					
	DE	EN	ES	FR	SV
DE	–	50.27	54.91	56.00	57.91
EN	52.65	–	61.28	63.86	**63.72**
ES	49.19	50.04	–	**64.43**	59.65
FR	49.37	**53.25**	**64.41**	–	57.78
SV	**54.83**	50.25	60.27	60.04	–

Table 3: Cross-lingual parsing results (LAS) using translated treebanks (phrase-based model) and DCA-based annotation projection. The table to the left contrasts the result with two-sample jackknifing experiments where the source side dependencies are created by automatically parsing each half of the treebank using a model trained on the other half of the training data.

learning curves for increasing amounts of training data using the example of data projected from English to other languages. The figure illustrates that the LAS levels out at around 10,000 - 20,000 sentences and this trend is essentially the same for all other languages as well.

4.3 Annotation Projection with Synthetic Machine-Translated Data

The next possibility we would like to explore is the use of synthetic parallel data. Annotating parallel data with a statistical parser may lead to quite a lot of noise especially when the domain does not match the original training data. Starting with noisy source language annotations, the projection algorithm may transfer errors to the target language that can cause problems for the target language parsing model induced from that data. Using machine translation and the original source language treebanks, we avoid this kind of error propagation. Furthermore, we suspect that human translations are more difficult to align on the word level then machine translated data which are inherently based on word alignments and, therefore, tend to be more literal and consistent (Carpuat and Simard, 2012). Using statistical MT as our translation model, we can also obtain such alignment as a given output from the decoding process, which makes it unnecessary to run yet another error-prone process such as automatic word alignment. Furthermore, the treebank data is too small to be used alone with generative statistical alignment models. Concatenating the data with larger parallel data would help but domain mismatches may, again, negatively influence the alignment performance.

In the following, we show the cross-lingual scores obtained by translating all treebanks in the Universal Treebank to all other languages. We leave out Korean here again, because no SMT training data is included in Europarl for that language. The translation models are trained on the entire Europarl corpus using a standard setup for phrase-based SMT and the Moses toolbox for training, tuning and decoding (Koehn et al., 2007). For tuning we use MERT (Och, 2003) and the newstest 2012 data provided by the annual workshop on statistical machine translation.[4] and for language modeling, we use a combination of Europarl and News data provided from the same source. The language model is a standard 5-gram model estimated from the monolingual data using modified Kneser-Ney smoothing without pruning (applying KenLM tools (Heafield et al., 2013)).

Table 3 summarizes the labeled attachment scores obtained with our projection approach on synthetic machine-translated data. The main observation we can make here is that this approach is very robust with respect to the noise introduced by the translation engine. Automatic translation is a difficult task on its own but we still achieve results that are similar to the ones from the projection approach on human translated data. Note that our training data is now much smaller[5] compared to the data sizes used in Section 4.2 and, still, we outperform those models in several cases. This seems to prove that it can be a clear advantage to start with gold annotations in the source language and to have a close alignment between source and target language. An indication for this effect is illustrated by the contrastive jackknifing experiments shown in Table 3. The scores are generally lower with two minor exceptions. Note

[4] http://www.statmt.org/wmt14

[5] Most treebanks includes 2,000-5,000 sentences, except English with about 40,000 sentences.

that this experiment does not cover domain shift problems. Another trend that can be seen in our results is that some languages such as German are more difficult to translate to (which can be confirmed by the SMT literature) leading to lower cross-lingual parsing performance.

4.4 The Impact of Word Alignment

Crucial for the success of annotation projection is the quality of the word alignment used to map information from the source to the target language. Not only alignment errors cause problems but also ambiguous alignments can lead to projection difficulties as we have discussed before. In the previous sections, we relied on symmetrized word alignments that are common in the SMT community, which are based on Viterbi alignments created by the final IBM model 4 in the typical training pipeline. Even though this is a reasonable setup for training phrase-based SMT models (as presented in the previous section), the chosen symmetrization heuristics (grow-diag-final-and) may not be well suited for accurate annotation projection. In particular, it is known that these heuristics focus on recall and tend to add many additional links that may not be useful for our projection task and even lead to some confusion as depicted in the example in Figure 3.

In order to investigate the impact of word alignment, we, therefore, decided to look at other sym-

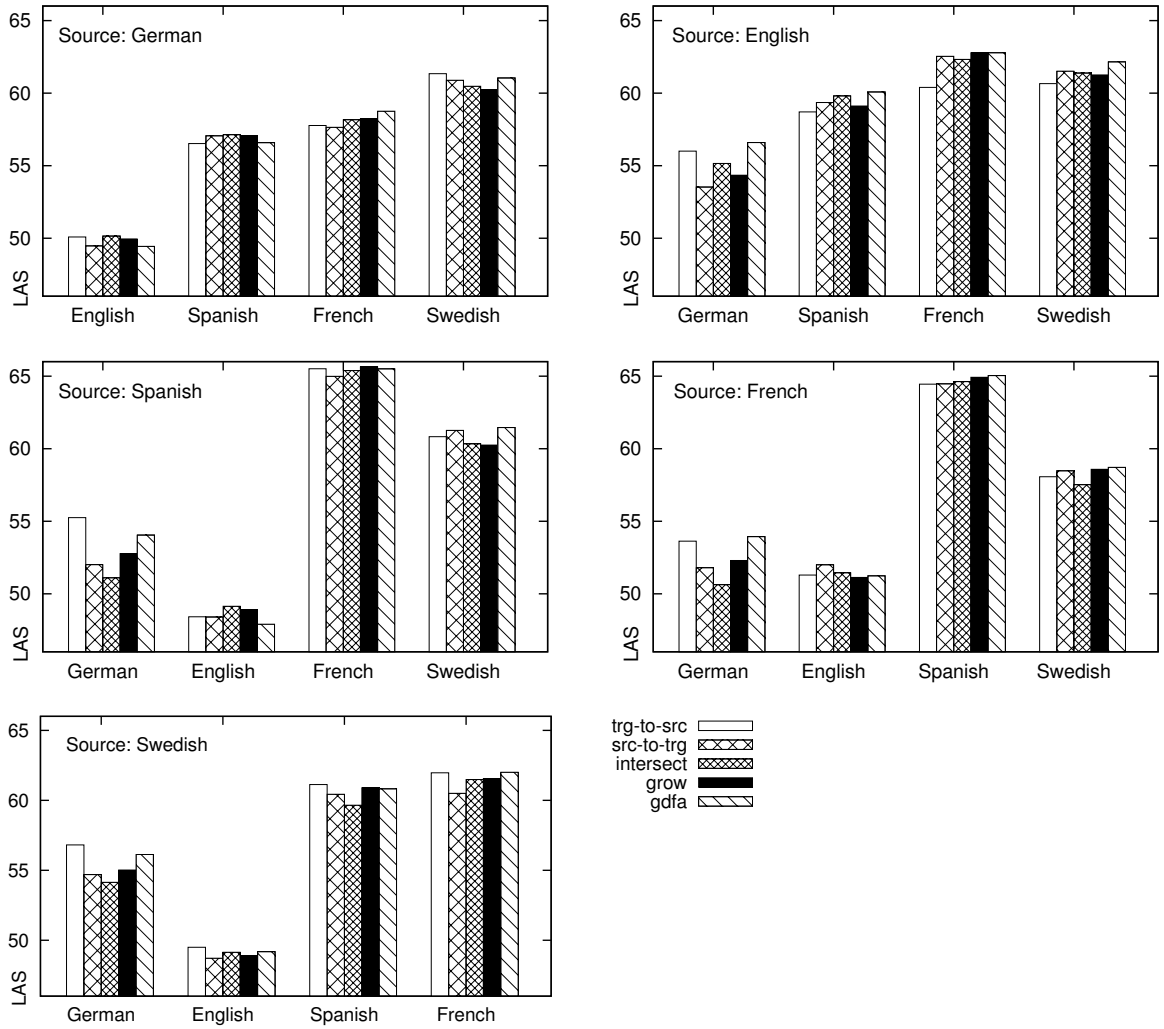


Figure 5: The impact of word alignment symmetrization on projection and parsing accuracy. *src-to-trg* and *trg-to-src* refer to the original directional Viterbi word alignments created by IBM model 4 in both directions; *intersect* refers to the intersection of both IBM 4 alignments; *grow* and *gdfa* (grow-diag-final-and) refer to popular symmetrization heuristics used in the SMT community.

metrization heuristics and their effect on projection and the quality of the parser model trained on the projected data. For this, we return to the setup of projecting annotations on human translations using the Europarl corpus with the same settings as described in Section 4.2 (using 40,000 sentences for the projection). We now compare five different word alignments based on IBM model 4 trained on the entire corpus for each language pair. First of all, we look at the original directional word alignment from source to target language and vice versa. We then include the intersection of these two directional link sets to represent a symmetrization heuristics that produces very sparse but high precision word alignments. Finally, we also consider the *grow* heuristics that adds adjacent alignment points coming from the union of directional alignment links to the sparse intersection of the same. In this way, the resulting word alignment covers most words while keeping precision at a rather high level. All of these alignment types are then contrasted with the *grow-diag-final-and* heuristics that we use in our default setup.

Figure 5 plots the parsing performance across languages based on the projection with the various alignment techniques listed above. A general observation is that the differences are rather small in most cases. Projecting annotation using the direct correspondence assumption seems to be quite robust with respect to alignment noise. In our experiments, no specific tendencies can be identified that would allow to draw immediate conclusions and to give clear recommendations for our task. Somewhat surprisingly we can see that the recall-oriented alignment heuristics (*grow-diag-final-and*) actually perform quite well in many cases, leading either to the best performing model or to one that is very close to the best result. However, in some cases, these models fall behind the ones based on alignment intersections (for instance Spanish-English) or directional word alignments (for example for Spanish-German, French-English, Swedish-German). A striking difference can be seen in the annotations projected to German. There, the target-to-source alignment performs pretty well and outperforms in two cases all other alignment types in the down-stream task. Furthermore, the intersection falls far behind in three of these cases, which indicates that both alignment directions are probably very different from each other leading to a very sparse word alignment when intersecting them. One possible reason for the success of the directional alignment might be that it favors the mapping to a compounding language such as German that frequently requires many-to-one links. However, the same effect cannot be seen for the other compounding language in our test set, Swedish.

4.5 Parsing Without Golden POS Labels

For a truly unsupported language, it does not make sense to assume a high quality POS tagger. Nevertheless, most cross-lingual experiments test their performance on data with human annotated golden POS labels. This is similar to the tradition of monolingual parsing where test accuracy is measured with perfect tokenization and completely correct POS annotation. In practice, this would not be realistic where new data needs to be parsed without proper tagging and unambiguous tokenization.

Direct transfer models are even more dependent on POS labels as those are the only source of information they can work with when making attachment decisions. Annotation projection approaches, on the other hand, are able to transfer POS information as well, which allows to train tagger models on projected data. In this section, we would like to test the feasibility of such an idea to see if we can truly port a parser to a new language without additional assumptions.

The first step is to train tagger models on our projected data sets. For this, we use the translated treebanks and a simple word-by-word translation approach in which we translate single-word-phrases only in our standard SMT model. The word-by-word translation model assures that we do not contaminate the data with DUMMY nodes and labels even though the translation quality lags behind the more powerful phrase-based models with larger translation options. We train standard Markov taggers with suffix backoff using HunPos (Halácsy et al., 2007) on each of the projected training data sets from the Universal Treebank. Table 4 summarizes the performance of all tagger models tested on the test sets in the treebank. The tagger all use the same universal POS tagset with its 12 labels as used in the Universal Treebank (Petrov et al., 2012). As we can see, the performance of those taggers is not great but still rather informative with overall accuracy values around 80%. The drop from source data to projected data is about 10-15 absolute points, which is, however, quite dramatic. Assuming that this is the best we can

POS	DE	EN	ES	FR	SV
DE	**95.24**	73.15	69.31	72.41	79.01
EN	82.04	**97.56**	79.91	81.23	84.44
ES	77.27	77.43	**95.37**	83.97	78.26
FR	80.99	78.74	88.47	**95.08**	79.62
SV	78.40	71.45	70.11	66.77	**95.86**

DELEXICALIZED MODELS

LAS	DE	EN	ES	FR	SV
DE	–	33.38	34.37	36.59	39.15
EN	36.55	–	45.53	47.71	**48.92**
ES	35.07	39.87	–	**51.40**	42.95
FR	35.89	**40.40**	**51.55**	–	40.30
SV	**37.87**	39.80	43.62	41.61	–

TRANSLATED TREEBANK MODELS

LAS	DE	EN	ES	FR	SV
DE	–	41.29	42.16	46.26	46.79
EN	42.24	–	50.54	53.63	**53.78**
ES	38.61	43.70	–	**57.58**	47.01
FR	**42.65**	**48.37**	**57.78**	–	45.55
SV	41.37	42.34	49.38	46.00	–

Table 4: Top (POS): Accuracy of POS tagging models trained on translated treebanks (word-by-word model). Bottom (LAS): Cross-lingual parser models tested on automatically POS tagged test sets. The delexicalized baseline (left) and the translated treebank model using word-by-word translation (right).

achieve for the target language, we now have to look at the parsing performance when relying on such noisy annotation.

Firstly, we look at the delexicalized baselines. The bottom-left part of Table 4 lists the labeled attachment scores when gold POS labels are replaced with automatic tags created by the corresponding projection tagger. The drop is huge and the original scores that were well above 50-70% go down to not more than 30-40% LAS. Clearly, this was to be expected as proper POS labeling is crucial for these models. Let us now look at the annotation projection approach using a translated treebank as our parallel data set. Table 4 on the bottom-right lists the corresponding labeled attachment scores with automatic POS tags. As expected, the performance is considerably lower than with golden POS labels, which are still the most informative features in those models. However, the performance remains in a range of above 40-50% LAS. Clearly, the lexical features help to keep the performance up at a higher level than the delexicalized baselines. We believe, that this difference can be crucial when porting language tools to new languages and that the models can be further optimized to rely less on golden POS tags.

5 Conclusions

In this paper we revisit annotation projection for cross-lingual parser induction. We show that annotation can successfully be transfered to target languages if the annotation is harmonized across languages. Despite previous negative results on diverse treebanks we demonstrate that direct projection works very well for a number of languages and outperforms direct delexicalized transfer models by a large margin. The approach is also quite robust with respect to word alignment. Furthermore, we show that machine translation can be a useful alternative for this strategy and that projected data can also be used to induce basic information such as POS labels in combination with syntactic parser models.

Acknowledgements

This work was supported by the Swedish Research Council (Vetenskapsrådet), project 2012-916. I would also like to thank Joakim Nivre, Željko Agić and the anonymous reviewers for helpful comments and suggestions.

References

Željko Agić, Danijela Merkler, and Daša Berović. 2012. Slovene-Croatian Treebank Transfer Using Bilingual Lexicon Improves Croatian Dependency Parsing. In *Proceedings of IS-LTC 2012*, pages 5–9.

Miguel Ballesteros and Joakim Nivre. 2012. MaltOptimizer: An Optimization Tool for MaltParser. In *Proceedings of EACL 2012*, pages 58–62.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of CoNLL 2006*, pages 149–164.

Marine Carpuat and Michel Simard. 2012. The trouble with smt consistency. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 442–449, Montréal, Canada.

Kuzman Ganchev and Dipanjan Das. 2013. Cross-Lingual Discriminative Learning of Sequence Models with Posterior Regularization. In *Proceedings of EMNLP 2013*, pages 1996–2006.

Péter Halácsy, András Kornai, and Csaba Oravecz. 2007. Poster paper: Hunpos – an open source trigram tagger. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 209–212, Prague, Czech Republic.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable Modified Kneser-Ney Language Model Estimation. In *Proceedings of ACL 2013*, pages 690–696.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping Parsers via Syntactic Projection across Parallel Texts. *Natural Language Engineering*, 11(3):311–325.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Christopher J. Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of ACL 2007*, pages 177–180.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-Source Transfer of Delexicalized Dependency Parsers. In *Proceedings of EMNLP 2011*, pages 62–72.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal Dependency Annotation for Multilingual Parsing. In *Proceedings of ACL 2013*, pages 92–97.

Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective Sharing for Multilingual Dependency Parsing. In *Proceedings of ACL 2012*, pages 629–637.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. MaltParser: A Data-Driven Parser-Generator for Dependency Parsing. In *Proceedings of LREC 2006*, pages 2216–2219.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–52.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of ACL 2003*, pages 160–167.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A Universal Part-of-Speech Tagset. In *Proceedings of LREC 2012*, pages 2089–2096.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual Word Clusters for Direct Transfer of Linguistic Structure. In *Proceedings of NAACL 2012*, pages 477–487.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013a. Token and Type Constraints for Cross-lingual Part-of-speech Tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12.

Oscar Täckström, Ryan McDonald, and Joakim Nivre. 2013b. Target Language Adaptation of Discriminative Transfer Parsers. In *Proceedings of NAACL 2013*, pages 1061–1071.

Jörg Tiedemann, Željko Agić, and Joakim Nivre. 2014. Treebank translation for cross-lingual parser induction. In *Proceedings of the 18th Conference Natural Language Processing and Computational Natural Language Learning (CoNLL)*, Baltimore, Maryland, USA.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing Multilingual Text Analysis Tools via Robust Projection Across Aligned Corpora. In *Proceedings of HLT 2011*, pages 1–8.

Yue Zhang and Joakim Nivre. 2011. Transition-based Dependency Parsing with Rich Non-local Features. In *Proceedings of ACL 2011*, pages 188–193.

Synchronous Constituent Context Model for Inducing Bilingual Synchronous Structures

Xiangyu Duan **Min Zhang*** **Qiaoming Zhu**
School of Computer Science & Technology, Soochow University
{xiangyuduan;minzhang;qmzhu}@suda.edu.cn

Abstract

Traditional Statistical Machine Translation (SMT) systems heuristically extract synchronous structures from word alignments, while synchronous grammar induction provides better solutions that can discard heuristic method and directly obtain statistically sound bilingual synchronous structures. This paper proposes Synchronous Constituent Context Model (SCCM) for synchronous grammar induction. The SCCM is different to all previous synchronous grammar induction systems in that the SCCM does not use the Context Free Grammars to model the bilingual parallel corpus, but models bilingual constituents and contexts directly. The experiments show that valuable synchronous structures can be found by the SCCM, and the end-to-end machine translation experiment shows that the SCCM improves the quality of SMT results.

1 Introduction

Traditional Statistical Machine Translation (SMT) learns translation model from bilingual corpus that is sentence aligned. No large-scale hand aligned structures inside the parallel sentences are usually available to the SMT community, while the aligned structures are essential for training the translation model. Thus, various unsupervised methods had been explored to automatically obtain aligned structures inside the parallel sentences. Currently, the dominant method is a two step pipeline that obtains word alignments by unsupervised learning (Brown et al., 1993) at the first step, then obtains aligned structures at the second step by heuristically extracting all bilingual structures that are consistent with the word alignments.

The second step in this two step pipeline is problematic due to its obtained aligned structures, whose counts are heuristically collected and violate valid translation derivations, while most SMT decoders perform translation via valid translation derivations. This problem leads to researches on synchronous grammar induction that discards the heuristic method and the two separate steps pipeline.

Synchronous grammar induction aims to directly obtain aligned structures by using one statistically sound model. The aligned structures in synchronous grammar induction are hierarchical/syntax level (Cohn and Blunsom, 2009) synchronous structures, which can be modeled by Synchronous Context Free Grammars (SCFGs) (Cohn and Blunsom, 2009; Levenberg et al., 2012; Xiao et al., 2012; Xiao and Xiong, 2013) or a kind of SCFGs variant - Inversion Transduction Grammars (ITGs) (Neubig et al., 2011; Cohn and Haffari, 2013). Both SCFGs and ITGs are studied in recent years by using generative or discriminative modeling.

This paper departs from using the above two traditional CFGs-based grammars, and proposes Synchronous Constituent Context Model (SCCM) which models synchronous constituents and contexts directly so that bilingual translational equivalences can be directly modeled. The proposed SCCM is inspired by researches on monolingual grammar induction, whose experience is valuable to the synchronous grammar induction community due to its standard evaluation on released monolingual treebanks, while no hand annotated bilingual synchronous treebank is available for evaluating synchronous

*Corresponding Author

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1865–1874, Dublin, Ireland, August 23-29 2014.

grammar induction. According to the evaluation results, the state-of-the-art monolingual grammar induction was achieved by Bayesian modeling of the Constituent Context Model (CCM) (Duan et al., 2013; Klein and Manning, 2002), while traditional CFGs based monolingual grammar induction methods perform well below the CCM.

In view of the significant achievements of the CCM in monolingual grammar induction, we propose the SCCM to apply the CCM to the bilingual case. The tremendous possible constituents and contexts incurred in this bilingual case put a challenge for the SCCM to model such kind of sparse variables. We further propose a non-parametric Bayesian Modeling of the SCCM to cope with the sparse variables. Experiments on Chinese-English machine translation show that meaningful synchronous phrases can be detected by our SCCM, and the performance of the end-to-end SMT is significantly improved.

The rest of the paper is structured as follows: we propose the SCCM in Section 2. The non-parametric Bayesian modeling of the SCCM is presented in Section 3, followed by the presentation of posterior inference for the Bayesian SCCM. Then experiments and results are presented. Conclusion are presented in the final section.

2 Synchronous Constituent Context Model (SCCM)

We propose the SCCM to model synchronous structures explicitly. Unlike Synchronous Context Free Grammars (SCFGs) which are defined on latent production rules of parallel corpus, the SCCM deals with both synchronous tree spans ($syn\ spans$) and non-synchronous spans ($non\text{-}syn\ spans$). All spans are represented by two kinds of strings: bilingual constituents and bilingual contexts. The SCCM is a generative model defined over such representations.

2.1 Bilingual Constituents and Contexts

By extending the concept of constituents and contexts introduced in (Klein and Manning, 2002), we define bilingual constituents and contexts as follows. Bilingual constituents are pairs of contiguous surface strings of sentence spans (bilingual subsequences), bilingual contexts are tokens preceding and following the bilingual constituents. In the SCCM, each bi-span in a sentence pair, either a $syn\ span$ or a $non\text{-}syn\ span$, is represented by a bilingual constituent and a bilingual context.

Fig. 1 gives an illustration of the bilingual constituents and contexts. In Fig. 1-(a), a latent synchronous tree over the example sentence pair is illustrated. With the word alignments shown in the sentence pair, the latent tree over the target sentence "$e_1\ e_2\ e_3$" can be inferred. For the ease of presentation, the latent target side tree is neglected in Fig. 1-(a).

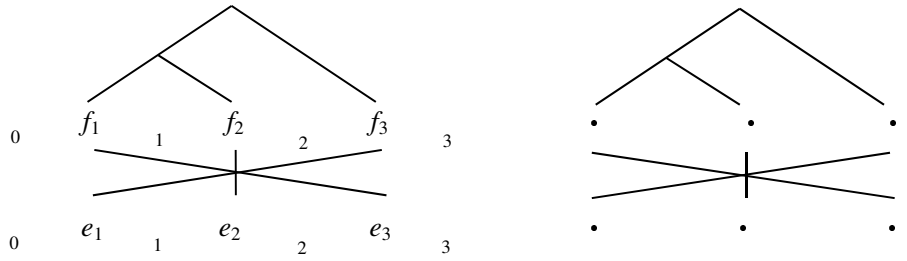
Given the synchronous tree, two sets of bilingual constituents and contexts can be extracted as shown in the two tables of Fig. 1. One is about $syn\ span$s, the other is about $non\text{-}syn\ span$s. $\diamond$ appearing in the contexts denotes a sentence boundary. *nil* appearing in the constituents of the non-tree spans denotes an empty span, which is actually a space between two terminals (or between a terminal and $\diamond$).

2.2 Generative Model

The SCCM computes the joint probability of a sentence pair S and its synchronous tree T as below:

$$\begin{aligned} &P(S,T) = P(S|T)P(T) = P(S|T)P(B)P(T|B) \\ &= P(S|T)P(B) \prod_{\substack{0\le i\le j\le m \\ 0\le p\le q\le n}} P(\alpha_{ij,pq}|B_{ij,pq})P(\beta_{ij,pq}|B_{ij,pq}) \end{aligned} \tag{1}$$

where B denotes a synchronous bracketing skeleton, in which no words are populated. Fig. 1-(b) shows the skeleton of Fig. 1-(a). The skeleton B is considered being filled by the synchronous tree T, and $P(T|B)$ is decomposed into conditional probabilities of bilingual constituents α and contexts β conditioning on $B_{ij,pq}$, a Boolean variable indicating whether the under-consideration bi-span $<i,j><p,q>$ is a $syn\ span$ or not. In particular, $\alpha_{ij,pq}$ denotes the bilingual constituent spanning from i to j on source side sentence, and spanning from p to q on target side sentence. $\beta_{ij,pq}$ denotes the context of $\alpha_{ij,pq}$.



(*a*) (*b*)

syn span	<0,1><2,3>	<1,2> <1,2>	<2,3><0,1>	<0,2><1,3>	<0,3><0,3>
constituent	$(f_1)(e_3)$	$(f_2)(e_2)$	$(f_3)(e_1)$	$(f_1 f_2)(e_2\ e_3)$	$(f_1 f_2 f_3)(e_1\ e_2\ e_3)$
context	$(\diamond\text{-}f_2)(e_2\text{-}\diamond)$	$(f_1\text{-}f_3)(e_1\text{-}\ e_3)$	$(f_2\text{-}\diamond)(\diamond\text{-}\ e_2)$	$(\diamond\text{-}f_3)(e_1\text{-}\diamond)$	$(\diamond\text{-}\diamond)(\diamond\text{-}\diamond)$

non-syn span	<1,3><0,1>	<1,1><1,2>	…
constituent	$(f_2 f_3)(e_1)$	$(nil)(e_2)$	…
context	$(f_1\text{-}\diamond)(\diamond\text{-}e_2)$	$(f_1\text{-}f_2)(e_1\text{-}\ e_3)$	…

Figure 1: Illustration of bilingual constituents and contexts over a sentence pair which consists of a source side sentence "$f_1\ f_2\ f_3$" and a target side sentence "$e_1\ e_2\ e_3$". In (a), the bottom numbers around each word are indexes for denoting spans. A synchronous tree is illustrated in (a), based on which two sets of bilingual constituents and contexts are extracted as shown in the two tables below the tree. Take a $syn\ span$ <1,2><1,2> for example, the source side span <1,2> is "f_2" and the target side span <1,2> is "e_2". They constitutes a bilingual constituent "$(f_2)(e_2)$", whose context is "$(f_1\text{-}f_3)(e_1\text{-}e_3)$" that is preceding and following the bilingual constituent. Figure (b) shows the skeleton of figure (a).

$B_{ij,pq}$ is defined as below:

$$B_{ij,pq} = \begin{cases} 1 & if\ bispan < i,j >< p,q > is\ a\ syn\ span \\ 0 & otherwise \end{cases}$$

In the SCCM, skeletons Bs are restricted to be binary branching and are distributed uniformly. Furthermore, since T and S are consistent, $P(S|T)$ is always equal to 1 in Eq. (1). Therefore, we can infer (with the expansion of the continued multiplication operator of Eq. (1)):

$$P(S,T) \propto \prod_{<i,j><p,q>\in T} (P(\alpha_{ij,pq}|B_{ij,pq}=1)P(\beta_{ij,pq}|B_{ij,pq}=1)) \prod_{<i,j><p,q>\notin T} (P(\alpha_{ij,pq}|B_{ij,pq}=0)P(\beta_{ij,pq}|B_{ij,pq}=0)) \quad (2)$$

where $<i,j><p,q> \in T$ indicates that bi-span $<i,j><p,q>$ is a $syn\ span$ contained in T, $<i,j><p,q> \notin T$ indicates otherwise case. Formula (2) is the basis for Bayesian modeling of the SCCM and the posterior inference that are proposed in the following sections.

3 Bayesian Modeling for the SCCM

For the SCCM, the posterior of a synchronous tree T given the observation of a sentence pair S is: $P(T|S) \propto P(S,T)$. As shown in formula (2), it turns out that the posterior $P(T|S)$ depends on the four kinds of distributions:

$$P(\alpha_{ij,pq}|B_{ij,pq}=1) \qquad P(\beta_{ij,pq}|B_{ij,pq}=1)$$
$$P(\alpha_{ij,pq}|B_{ij,pq}=0) \qquad P(\beta_{ij,pq}|B_{ij,pq}=0)$$

We propose to define two kinds of Bayesian priors over the constituents related variables $\alpha_{ij,pq}|B_{ij,pq}$ and the contexts related variables $\beta_{ij,pq}|B_{ij,pq}$ respectively. Since constituents exhibits richer appearances than contexts, the proposed Bayesian prior over $\alpha_{ij,pq}|B_{ij,pq}$ is more complicate than that over $\beta_{ij,pq}|B_{ij,pq}$.

Specifically, one of the non-parametric Bayesian priors, the Pitman-Yor-Process (PYP) prior, is defined on $\alpha_{ij,pq}|B_{ij,pq}$. The PYP prior can produce the power-law distribution (Goldwater et al., 2009) that is commonly observed in natural languages, and can flexibly model distributions on layer structures due to its defined distribution on distribution hierarchy. The PYP prior had been successfully applied on many NLP tasks such as language modeling (YeeWhye, 2006), word segmentation (Johnson et al., 2007b; Goldwater et al., 2011), dependency grammar induction (Cohen et al., 2008; Cohn et al., 2010), grammar refinement (Liang et al., 2007; Finkel et al., 2007) and Tree-Substitution Grammar induction (Cohn et al., 2010). We use the PYP to model the constituents' layered structure by using the PYP's distribution hierarchy. On $\beta_{ij,pq}|B_{ij,pq}$, we use the Dirichlet distribution for its simplicity because contexts appear in much fewer kinds of surface strings than those of constituents.

3.1 The PYP Prior over Bilingual Constituents

Constituents consist of both words and POS tags. Though in much monolingual grammar induction works, only POS tag sequences were used as the observed constituents for their significant hints of phrases (Klein and Manning, 2002; Cohn et al., 2010), our work needs considering raw words as observation data too because word alignments encode the important translation correspondence and contribute to synchronous bi-spans. But it causes severe data sparse problem due to the quite large number of unique constituents consisting of both words and POS tags. Besides, constituents can be extremely long which intensify the data sparse problem. So, solely using the surface strings of constituents is impractical.

In this section, we propose a hierarchical representation of constituents to overcome the data sparse problem and use the PYP prior on this kind of representation. From top to bottom, the hierarchical representation encodes the information of a bilingual constituent from fine-grained level to coarse-grained levels. The probability of a fine-grained level can be backed-off to the probabilities of coarse-grained levels.

The first (top) level of the hierarchical representation is the bilingual constituent itself. The second level is composed of two sequences: one is word sequence, the other is POS tags sequence. The third level mainly decomposes the second level into boundaries and middle words/POSs. Since the target of inducing synchronous structures in this paper is to induce the latent phrasal equivalences of a parallel sentence, boundaries of bilingual constituents play the key role of identifying phrasal equivalences. The third level is the function to make use of boundaries. Fig. 2 gives an illustration of the hierarchical representation.

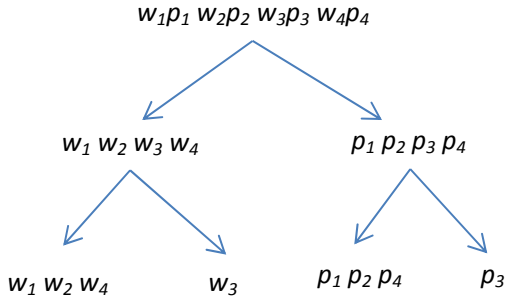


Figure 2: Illustration of the hierarchical representation of a bilingual constituent "w_1p_1 w_2p_2 w_3p_3 w_4p_4". Here w and p denote word and POS respectively, and the suffixes denote positions. Note that both w and p are composite, w denotes a source side word and a target side word, and p denotes the POS case. The second level decomposes the first level into a word sequence and a POS sequence, and the third level decomposes further into boundaries and middle words/POSs. The boundary width in this figure is two for left side boundary and one for right side boundary.

The PYP prior encodes distribution on distribution. Recursively using the PYP prior can create a distribution hierarchy, which is appropriate for modeling the distribution over the hierarchical representations of constituents. Smoothing is fulfilled through backing-off fine-grained level distributions to coarse-grained level distributions.

3.1.1 The PYP Hierarchy

We define the PYP hierarchy over the hierarchical representation of bilingual constituents in a top-down manner. For the topmost (first) level:

$$\begin{aligned} \alpha_{ij,pq}|B_{ij,pq} = b &\sim G_b^{first} \\ G_b^{first} &\sim PYP(d_b^{first}, \theta_b^{first}, P_{word-pos}(.|B_{ij,pq} = b)) \end{aligned}$$

The PYP has three parameters: $(d_b^{first}, \theta_b^{first}, P_{word-pos})$. $P_{word-pos}(.|B_{ij,pq} = b)$ is a $base\ distribution$ over infinite space of bilingual constituents conditioned on span type b, which provides the back-off probability of $P(\alpha_{ij,pq}|B_{ij,pq} = b)$. The remaining parameters d_b^{first} and θ_b^{first} control the strength of the base distribution.

The back-off probability $P_{word-pos}(\alpha_{ij,pq} = x|B_{ij,pq} = b)$ is defined as below:

$$P_{word-pos}(\alpha_{ij,pq} = x|B_{ij,pq} = b)) = P_{word}(Rw(x)|b) \times P_{pos}(Rp(x)|b)$$

where $Rw(x)$ is the function returning a word sequence of a bilingual constituent x, $Rp(x)$ returning the correspondent POS sequence. This is the second level of the hierarchical representation of bilingual constituents as illustrated in Fig. 2. Further, $Rw(x)$ and $Rp(x)$ are decomposed into the third level of the hierarchy. Taking $Rw(x)$ for example:

$$P_{word}(Rw(x)|B_{ij,pq} = b)) = P_{word-bound}(Rwb(x)|b) \times \frac{1}{|W|^{|Rw(x)|-|Rwb(x)|}}$$

where Rwb is a function returning a word sequence's boundary representation, $|W|$ is the vocabulary size, $|Rw(x)| - |Rwb(x)|$ is the number of the words in $Rw(x)$ excluding those in the boundary representation. The above equation models the generation of a word sequence with surface string $Rw(x)$ (given b) by first generating its boundary representation $Rwb(x)$, then generating its middle words from a uniform distribution over the vocabulary. $P_{pos}(Rp(x)|B_{ij,pq} = b))$ is defined similarly.

We put the Dirichlet prior over $P_{word-bound}(Rwb(x)|b)$:

$$\begin{aligned} Rwb(x)|b &\sim Discrete(G_b^{Rwb}) \\ G_b^{Rwb} &\sim Dirichlet(\tau_b) \end{aligned}$$

For $P_{pos-bound}(Rpb(x)|b)$, similar definition to $P_{word-bound}(Rwb(x)|b)$ is applied.

3.2 The Dirichlet Prior over Bilingual Contexts

The Dirichlet prior is defined as below:

$$\begin{aligned} \beta_{ij,pq}|B_{ij,pq} = b &\sim Discrete(G_b^{Dir}) \\ G_b^{Dir} &\sim Dirichlet(\tau_b) \end{aligned}$$

A context $\beta_{ij,pq}$ (given the specific span type b) is drawn $i.i.d$ according to a multinomial parameter G_b^{Dir}, which is drawn from the Dirichlet distribution with a real value parameter τ_b.

4 MCMC Sampling for Inferring the Latent Synchronous Trees

We approximate the distribution over latent synchronous trees by sampling them from the posterior $P(T|S)$, where T is a latent synchronous tree of a sentence pair S. As presented in the beginning of section 3, the posterior depends on $P(\alpha_{ij,pq}|B_{ij,pq} = b)$ and $P(\beta_{ij,pq}|B_{ij,pq} = b)$, on which we put the PYP prior and the Dirichlet prior respectively. Because of integrating out all Gs in all of the priors, interdependency between samples of $\alpha_{ij,pq}|B_{ij,pq} = b$ or $\beta_{ij,pq}|B_{ij,pq} = b$ is introduced, resulting in simultaneously obtaining multiple samples impractical. On the other hand, blocked sampling, which obtains sentence-level samples simultaneously (Blunsom and Cohn, 2010; Cohn et al., 2010; Johnson et al., 2007a) is attractive for the fast mixing speed and the easy application of standard dynamic programming algorithms.

4.1 Metropolis-Hastings (MH) Sampler

We apply a MH sampler similar to (Johnson et al., 2007a) to overcome the difficulty of obtaining multiple samples simultaneously from posterior. The MH sampler is a MCMC technique that draws samples from a true distribution by first drawing samples simultaneously from a proposal distribution, and then correcting the samples to the true distribution by using an accept/reject test. In practical, the proposal distribution is designed to facilitate the use of blocked sampling that applies standard dynamic programming, and the resulting samples are corrected by the accept/reject test to the true distribution.

In our case, the proposal distribution is the Maximum-a-Posteriori (MAP) estimate of $P(\alpha_{i,j}|B_{i,j} = b)$ and $P(\beta_{i,j}|B_{i,j} = b)$, and the blocked sampling of T applies a dynamic programming algorithm that is based on the inside chart derived from a transformation of Eq. (1):

$$P(S,T) = K(S) \prod_{<i,j><p,q>\in T} \phi(ij,pq)$$

$$where\ \phi(ij,pq) = \frac{P(\alpha_{ij,pq}|B_{ij,pq}=1)P(\beta_{ij,pq}|B_{ij,pq}=1)}{P(\alpha_{ij,pq}|B_{ij,pq}=0)P(\beta_{ij,pq}|B_{ij,pq}=0)}$$

$K(S)$ is a constant given S. The inside chart I can be constructed recursively as below:

$$I_{ij,pq} = \begin{cases} \phi(ij,pq) & if\ j-i \leq 1\ and\ q-p \leq 1 \\ \phi(ij,pq) \sum\limits_{\substack{i\leq u\leq j \\ p\leq v\leq q}} (I_{iu,pv}I_{uj,vq} + I_{iu,vq}I_{uj,pv}) & otherwise \end{cases}$$

Based on this inside chart, a synchronous tree can be top-down sampled (Johnson et al., 2007a), then is accepted or rejected by the MH-test to correct to the true distribution.

5 Experiments

The experiments were conducted on both a pilot word alignment task and an end-to-end Chinese-to-English machine translation task to test the quality of the learned synchronous structures by the SCCM. The bi-side monolingual gold bracketings contained in Penn treebanks were not used for evaluating the quality of the learned synchronous structures because of great syntactic divergence between source tree and target tree, which results in that gold monolingual syntactic trees on both sides are asynchronous (large number of tree nodes can not be aligned). The synchronous grammar induction community assumes the existence of synchronous grammar for MT, and do not evaluate synchronous grammar induction on monolingual gold treebanks because of their asynchronous property. The synchronous grammar induction community is not the same with the multilingual grammar induction community, which targets at inducing bi-side monolingual syntactic trees. Due to the same reason, our synchronous bracketing induction method was not evaluated on bi-side monolingual bracketing trees which are asynchronous.

5.1 Sampler Configuration

Our sampler was initialised with trees through a random split process. Firstly, we used GIZA++ model 4 to get source-to-target and target-to-source word alignments, and used grow-diag-final-and (gdfa) heuristic to extract reliable word alignments for each sentence pair. Secondly, we randomly split each sentence pair in a top-down manner, and make sure that each split is consistent with the GIZA++ gdfa word alignments. For example, given a sentence pair of m source words and n target words, we randomly choose a split point at each side and the alignment type (straight alignment or inverted alignment), then recursively build bi-spans further on each new split. Finally, a synchronous binary tree is built at the end of this process [1]. Note that all splits must be consistent with the GIZA++ gdfa word alignments. When a piece of word alignments (such as non-ITG alignment structure) do not permit binary split, we keep this structure unsplitted and continue split only on its sub-structures that are ITG derivable.

Our sampler ran 200 iterations for all data. After each sampling iteration, we resample all the hyperparameters using slice-sampling, with the following priors: $d \sim Beta(1,1)$, $\theta \sim Gamma(10, 0.1)$.

The time complexity of our inference algorithm is $O(n^6)$, which is not practical in applications. We reduce the time complexity by only considering bi-spans that do not violate GIZA++ intersection word alignments (intersection of source-to-target and target-to-source word alignments) (Cohn and Haffari, 2013).

5.2 Word Alignment Task

5.2.1 Experimental Setting

Since there are no annotated synchronous treebanks, we evaluate the SCCM indirectly by evaluating its output word alignments on a gold standard English Chinese parallel tree bank with hand aligned word alignments referred as HIT corpus[2]. The HIT corpus, which was collected from English learning text books in China as well as example sentences in dictionaries, was originally designed for annotating bilingual tree node alignments. The annotation strictly reserves the semantic equivalence of the aligned sub-tree pair. The byproduct of this corpus is the hand aligned word alignments, which was utilized to evaluate word alignment performance[3]. The word segmentation, tokenization and parse-tree in the corpus were manually constructed or checked. The statistics of the HIT corpus are shown in table 1.

Table 1: Corpus statistics of the HIT corpus.

	ch	en
sent	16131	
word	210k	209k
avg. len.	13.06	13.0

5.2.2 Results

We adopt the commonly used metric: the alignment error rate (AER) to evaluate our proposed alignments (a) against hand-annotated alignments, which are marked with sure (s) and possible (p) alignments. The AER is given by (the lower the better):

$$AER(a, s, p) = 1 - \frac{|a \cap s| + |a \cap p|}{|a| + |s|}$$

In the HIT corpus, only sure alignments were annotated, possible alignments were bypassed because of the strict annotation standard of semantic equivalence.

The word alignments evaluation results are reported in Table 2. The baseline was GIZA++ model 4 in both directions with symmetrization by the grow-diag-final-and heuristic (Koehn et al., 2003). A

[1] The initialization with different random split bi-trees results in marginal variance of performances.

[2] HIT corpus is designed and constructed by HIT-MITLAB. http://mitlab.hit.edu.cn/index.php/resources.html

[3] We did not use annotated tree node alignments for synchronous structure evaluation because the coverage of tree nodes that can be aligned is quite low. The reason of low coverage is that Chinese and English exhibit great syntax divergences from monolingual treebank point of view.

released induction system - PIALIGN (Neubig et al., 2011)[4] was also experimented to compare with our proposed induction system - SCCM.

PIALIGN is a model that generalizes adaptor grammars for machine translation (MT), while our model is to generalize CCM for MT. Adaptor grammars has been successfully applied on shallow unsupervised tasks such as morphlogical/word analysis, while CCM has obtained state-of-the-art performance on the more complex unsupervised task - inducing syntactic trees. In view of CCM's successful monolingual application, we generalize it to bilingual case. In depth comparison: our SCCM deals with both consituents and distituents, and contexts of them, while PIALIGN only deals with constituents. Furthermore, SCCM does not model non-terminal rewriting rules, while PIALIGN model those rules which can rewrite a non-terminal into a complete subtree as adaptor grammars does. In addition, PIALIGN adopts a beam search algorithm of (Saers et al., 2009). Through setting small beam size, PIALIGN's time complexity is almost $O(n^3)$. But as critisized by (Cohn and Haffari, 2013), their heuristic beam search algorithm does not meet either of the Markov Chain Monte Carlo (MCMC) criteria of ergodicity or detailed balance. Our method adopts MCMC sampling (Johnson et al., 2007a) which meets the MCMC criteria.

We can see that the two induction systems perform significantly better than GIZA++, and our proposed SCCM performs better than PIALIGN. Manual evaluation for the quality of the phrase pairs generated from word alignments is also reported in Table 2. We considered the top-100 high frequency phrase pairs that are beyond word level and less than six words on both sides, and report the proportion of reasonably well phrase pairs through manual check. We found that more good phrase pairs can be derived from the SCCM's word alignments than from others.

Table 2: Quality of word alignments and their generated phrase pairs.

	AER	good phrase pairs proportion
GIZA++	0.322	0.493
PIALIGN	0.263	0.531
SCCM	0.255	0.534

5.3 Machine Translation Task

5.3.1 Experimental Setting

A released tourism-related domain machine translation data was used in our experiment. It consists of a parallel corpus extracted from the $Basic\,Travel\,Expression\,Corpus$ (BTEC), which had been used in evaluation campaigns of the yearly International Workshop on Spoken Language Translation (IWSLT). Table 3 lists statistics of the corpus used in the experiment.

Table 3: Statistics of the corpus used by IWSLT

	ch	en
sent	23k	
word	190k	213k
avg. len.	8.3	9.2

We used CSTAR03 as development set, used IWSLT04 and IWSLT05 official test set for test. A 4-gram language model with modified Kneser-Ney smoothing was trained on English side of parallel corpus. We use minimum error rate training (Och, 2003) with nbest list size 100 to optimize the feature weights for maximum development BLEU. Experimental results were evaluated by case-insensitive BLEU-4 (Papineni et al., 2001). Closest reference sentence length was used for brevity penalty.

5.3.2 Results

Following (Levenberg et al., 2012; Neubig et al., 2011; Cohn and Haffari, 2013), we evaluate our model by using the SCCM's output word alignments to construct a phrase table. As a baseline, we train a phrase-based model using the moses toolkit [5] based on the word alignments obtained using GIZA++

[4]http://www.phontron.com/pialign/

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

model 4 in both directions and symmetrized using the grow-diag-final-and heuristic (Koehn et al., 2003). For comparison with CFG-based induction systems, word alignments generated by the PIALIGN were also used to train a phrase-based model.

In the end-to-end MT evaluation, we used the standard set of features: relative-frequency and lexical translation model probabilities in both directions; distance-based distortion model; language model and word count. The evaluation results are reported in table 4. Word alignments derived by the two induction systems can be more helpful to obtain better translations than GIZA++ derived word alignments. The SCCM, while departing from traditional CFG-based methods, achieves comparable translation performance to the PIALIGN.

Table 4: BLEU on both the development set: CSTAR03, and the two test sets: IWSLT04 and IWSLT05.

	CSTAR03	IWSLT04	IWSLT05
GIZA++	0.4304	0.4190	0.4866
PIALIGN	0.4661	0.4556	0.5248
SCCM	0.4560	0.4469	0.5193

6 Conclusion

A new model for synchronous structure induction is proposed in this paper. Different to all the previous works that are based on Context Free Grammars, our proposed SCCM deals with bilingual constituents and contexts explicitely so that bilingual translational equivalences can be directly modeled. A non-parametric Bayesian modeling of the SCCM is applied to cope with the sparse representations of bilingual constituents and contexts. Both intrinsic evaluation on word alignments and extrinsic evaluation on end-to-end machine translation were conducted. The intrinsic evaluation show that the highest quality word alignments were obtained by our proposed SCCM. Such statistically sound word alignments of the SCCM were used in the extrinsic evaluation on machine translation, showing that significantly better translations were achieved than those obtained by using the word alignments of GIZA++, the widely used word aligner in the two-step pipeline.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant No. 61273319, and grant No. 61373095. Thanks for the helpful advices of anonymous reviewers.

References

Phil Blunsom and Trevor Cohn. 2010. Unsupervised induction of tree substitution grammars for dependency parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1204–1213. Association for Computational Linguistics.

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

Shay B Cohen, Kevin Gimpel, and Noah A Smith. 2008. Logistic normal priors for unsupervised probabilistic grammar induction. In *Proceedings of the Advances in Neural Information Processing Systems*.

Trevor Cohn and Phil Blunsom. 2009. A bayesian model of syntax-directed tree to string grammar induction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 352–361. Association for Computational Linguistics.

Trevor Cohn and Gholamreza Haffari. 2013. An infinite hierarchical bayesian model of phrasal translation. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*.

Trevor Cohn, Phil Blunsom, and Sharon Goldwater. 2010. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096.

Xiangyu Duan, Zhang Min, and Chen Wenliang. 2013. Smoothing for bracketing induction. In *Proceedings of 23rd International Joint Conference on Artificial Intelligence*. AAAI Press/International Joint Conferences on Artificial Intelligence.

Jenny Rose Finkel, Trond Grenager, and Christopher D Manning. 2007. The infinite tree. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 272.

Sharon Goldwater, Thomas L Griffiths, and Mark Johnson. 2011. Producing power-law distributions and damping word frequencies with two-stage language models. *Journal of Machine Learning Research*, 12:2335–2382.

Mark Johnson, Thomas Griffiths, and Sharon Goldwater. 2007a. Bayesian inference for pcfgs via markov chain monte carlo. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 139–146.

Mark Johnson, Thomas L Griffiths, and Sharon Goldwater. 2007b. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. *Proceedings of Advances in neural information processing systems*, 19:641.

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

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.

Abby Levenberg, Chris Dyer, and Phil Blunsom. 2012. A bayesian model for learning scfgs with discontiguous rules. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 223–232. Association for Computational Linguistics.

P. Liang, S. Petrov, M. I. Jordan, and D. Klein. 2007. The infinite PCFG using hierarchical Dirichlet processes. In *Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP/CoNLL)*.

Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. 2011. An unsupervised model for joint phrase alignment and extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 632–641. Association for Computational Linguistics.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics.

Markus Saers, Joakim Nivre, and Dekai Wu. 2009. Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 29–32. Association for Computational Linguistics.

Xinyan Xiao and Deyi Xiong. 2013. Max-margin synchronous grammar induction for machine translation. In *EMNLP*.

Xinyan Xiao, Deyi Xiong, Yang Liu, Qun Liu, and Shouxun Lin. 2012. Unsupervised discriminative induction of synchronous grammar for machine translation. In *COLING*, pages 2883–2898.

Teh YeeWhye. 2006. A bayesian interpretation of interpolated kneser-ney. In *Technical Report TRA2/06*. School of Computing, National University of Singapore.

Syntactic Parsing and Compound Recognition via Dual Decomposition: Application to French

Joseph Le Roux[1] and **Matthieu Constant**[2] and **Antoine Rozenknop**[1]
(1) LIPN, Université Paris 13 – Sorbonne Paris Cité, CNRS
(2) LIGM, Université Paris Est, CNRS
leroux@univ-paris13.fr, mconstan@univ-mlv.fr, antoine.rozenknop@lipn.univ-paris13.fr

Abstract

In this paper we show how the task of syntactic parsing of non-segmented texts, including compound recognition, can be represented as constraints between phrase-structure parsers and CRF sequence labellers. In order to build a joint system we use dual decomposition, a way to combine several elementary systems which has proven successful in various NLP tasks. We evaluate this proposition on the French SPMRL corpus. This method compares favorably with pipeline architectures and improves state-of-the-art results.

1 Introduction

Dual decomposition (DD), which can be used as a method to combine several elementary systems, has already been successfully applied to many NLP tasks, in particular syntactic parsing, see (Rush et al., 2010; Koo et al., 2010) *inter alia*. Intuitively, the principle can be described quite simply: at decoding time, the combined systems seek for a consensus on common subtasks, in general the prediction of some parts of the overall structure, via an iterative process imposing penalties where the systems disagree. If the systems converge to a solution, it is formally guaranteed to be optimal. Besides, this approach is quite flexible and easy to implement. One can add or remove elementary systems without rebuilding the architecture from the ground up. Moreover, the statistical models for the subsystems can often be estimated independently at training time.

In this paper we show how syntactic parsing of unsegmented texts, integrating compound recognition, can be represented by constraints between phrase-structure parsers and sequence labellers, either for compound recognition or part-of-speech (POS) tagging, and solved using DD. We compare this approach experimentally with pipeline architectures: our system demonstrates state-of-the-art performance. While this paper focuses on French, the approach is generic and can be applied to any treebank with compound information, and more generally to tasks combining segmentation and parsing.

This paper is structured as follows. First, we describe the data we use to build our elementary systems. Second, we present related work in compound recognition, in particular for French, and the type of information one is able to incorporate in tag sets. Third, we show how CRF-based sequence labellers with these different tag sets can be combined using DD to obtain an efficient decoding algorithm. Fourth, we extend our method to add phrase-structure parsers in the combination. Finally, we empirically evaluate these systems and compare them with pipeline architectures.

2 Data

We use the phrase-structure treebank released for the SPMRL 2013 shared task (Seddah et al., 2013). This corresponds to a new version of the French Treebank (Abeillé et al., 2003). One of the key differences between French data and other treebanks of the shared task is the annotation of compounds. Compounds are sequences of words with a certain degree of semantic non-compositionality. They form

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1875–1885, Dublin, Ireland, August 23-29 2014.

a single lexical unit to which one can assign a single POS. In the SPRML corpus 15% of the tokens belong to a compound, or 12.7% if we omit numerals: the training, development and test sets respectively comprise 23658, 2120 and 4049 compounds.

In the treebank, compounds are annotated as subtrees whose roots are labelled with the POS of the compounds with a + suffix. Each leaf under a compound is the daughter of its own POS tag, which is in turn the daughter of the root of the compound. For example, the tree in Figure 1 contains a subtree with the compound adverb *pour l'instant* (so far) whose category `ADV+` dominates the preposition `pour`, the determiner *l'*, and the noun *instant*.

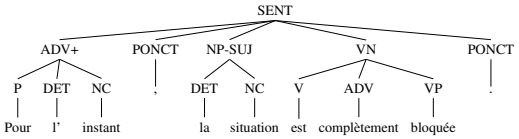


Figure 1: Syntatic annotation in the SPRML FTB: *So far, the situation has been completely blocked.*

The sequence labellers used in the experiments are able to exploit external lexical resources that will help coping with data missing from the training corpus. These resources are dictionaries, consisting of triplets `(flexed form, lemma, POS tag)`, where form and lemma may be compound or simple. Several such dictionaries exist for French. We use:

- DELA (Courtois et al., 1997) contains a million entries, among which 110,000 are compounds;
- Lefff (Sagot, 2010) contains 500,000 entries, among which 25,000 are compounds;
- Prolex (Piton et al., 1999) is a toponym dictionary with approximately 100,000 entries.

The described resources are additional to the SPMRL shared task data (Seddah et al., 2013), but were also used in (Constant et al., 2013a) for the shared task.

3 Compound Recognition

3.1 Related Work

The compound recognition traditionally relies on 2 types of information: lexical and syntactic clues. A strong lexical association between the tokens of a compound can be detected using a compound dictionary or by measuring a degree of *relatedness*, which can be learnt on a corpus. Some recent approaches use sequence labellers. The linear chain CRF model (Lafferty et al., 2001) is widely used, see for example (Vincze et al., 2011; Constant and Tellier, 2012). It has proven to be a very adequate model: it is flexible enough to incorporate information from labelled data and external resources (POS taggers, compound lexicons or named entity recognisers).

The compound recognition may also be directly performed by syntactic parsers learnt from corpora where compounds are marked, such as the one we use in this paper[1] (Arun and Keller, 2005; Green et al., 2011; Green et al., 2013; Constant et al., 2013b), but these results are contradictory. Green et al. (2011) experimentally show that a lexicalised model is better than an unlexicalised one. On the other hand, Constant et al. (2013b) show that, using products of PCFG-LA (Petrov, 2010), unlexicalised models can be more accurate. They obtain performance on a par with a linear chain CRF system without external information. But such information is difficult to incorporate directly in a PCFG-LA model. Constant et al. (2012) resort to a reranker to add arbitrary features in the parse selection process, but their system showed inferior performance compared with a CRF model with access to the same external information.

[1]Such an approach has been used already for joint named entity recognition and parsing based on CRF (Finkel and Manning, 2009).

3.2 Annotation schemes

Compound recognition can be seen as a segmentation task which consists in assigning to each token a label with segmentation information. We use label `B` if the token is the beginning of a word (single or compound), and label `I` if the token is inside a compound, but not in initial position. This lexical segmentation can be enriched with additional information, for example POS tags of compounds or tokens in compounds, and gives a variety of tag sets. This leads us to define 5 simple tag sets for our problem, each with very simple information, that will be combined in the next section. These tag sets are exemplified on a sentence with the compound *vin rouge* (red wine).

1. (basic) recognition with two labels (`B` and `I`)
   Luc/B aime/B le/B vin/B rouge/I (Luc likes red wine)
2. (partial) recognition with compound POS tags: `[BI]-POS` for tokens in compounds; `B` for others
   Luc/B aime/B le/B vin/B-NC+ rouge/I-NC+
3. (partial-internal) recognition with token POS tags in compounds
   Luc/B aime/B le/B vin/B-NC rouge/I-ADJ
4. (complete) recognition with POS tags for all tokens; in compounds use compound POS tags
   Luc/B-NPP aime/B-V le/B-DET vin/B-NC+ rouge/I-NC+
5. (complete-internal) recognition with POS tags for all tokens; in compounds use token POS tags
   Luc/B-NPP aime/B-V le/B-DET vin/B-NC rouge/I-ADJ

4 Dual decomposition for compound recognition using CRFs

4.1 A maximisation problem

4.1.1 CRF

A conditional random field (Lafferty et al., 2001), or CRF, is a tuple $c = (\Sigma, \mathcal{L}_c, w_c, \{f_p^c\}_p)$ which defines the conditional probability of a sequence of labels $y \in \mathcal{L}_c^*$ given a sequence of words of the same length $x \in \Sigma^*$ as a logistic regression of the form:

$$P_c(y|x) = \frac{\exp\left(\sum_{p\in\mathcal{P}(x)} w_c \cdot f_p^c(x, y_p)\right)}{Z(w_c, x)}, \text{ where} \tag{1}$$

- $w_c \in \mathbb{R}^d$ is a d-dimensional weight vector, where d is the number of features of the system,
- Z is the partition function
- $\mathcal{P}(x)$ is a set of *places*, in our case the set of unigram and bigram decompositions of sequences of words. A place p is of the form $[i]_x$ for unigrams and $[i, i+1]_x$ for bigrams. We omit x when context is unambiguous.
- y_p is the restriction of y to the place p, and we will write y_i for $y_{[i]}$ and $y_i y_{i+1}$ for $y_{[i,i+1]}$
- f_p^c is the feature function for the place p that projects (x, y_p) on $\mathbb{R}^d$.

Our goal is to find the best labelling, i.e. the one that maximises the conditional probability given a sequence of tokens. One can observe that this labelling also maximises the numerator of Equation 1, as $Z(w_c, x)$ does not depend on y. We therefore write:

$$\hat{y}^c = \arg\max_y \sigma_c(x, y) = \arg\max_y \sum_{p\in\mathcal{P}(x)} w_c \cdot f_p^c(x, y_p) \tag{2}$$

4.1.2 Viterbi Algorithm for CRFs

Since our combination of CRF systems relies on the Viterbi algorithm, we review it briefly. For a given input sentence $x = x_1 \dots x_n$, we represent the problem of finding the best labelling with a CRF c as a best path algorithm in a directed acyclic graph $\mathcal{G}_c = (\mathcal{V}, \mathcal{E})$ built from a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. Nodes are pairs (x_i, l) where x_i is an input token and l is an admissible label for x_i.[2] Edges connect nodes of the form (x_i, l) to nodes of the form (x_{i+1}, l') and the weights of these arcs are given by c. In order to find the weight of the best path in this graph, that corresponds to the score of the best labelling, we use Algorithm 1.[3] One can remark that the score of a node decomposes as a score s_1, computed from a vector of unigram features, written $f^c_{[i]}(x, l)$, and a score s_2 computed from a vector of bigram features, written $f^c_{[i-1,i]}(x, l', l)$.[4] The Viterbi algorithm has a time complexity linear in the length of the sentence and quadratic in the number of labels of the CRF.

Algorithm 1 Viterbi Algorithm for CRFs with unigram and bigram features

Viterbi$(\mathcal{G}_c, w_c, \{f^c_p\}_p, \Lambda^{BI}, \Lambda^{IB})$:
for all node v **do**
  $\pi[v] = -\infty$
end for
$\pi[$ (<s>, START) $] = w \cdot f_{p_0}(x,$ START$)$
for all non initial node $v = (x_i, l)$ in topological order **do**
  $s_1 \leftarrow w_c \cdot f^c_{[i]}(x, l)$
  $s_2 \leftarrow -\infty$
  for all incoming edge $v' = (x_{i-1}, l') \rightarrow v$ **do**
    $t \leftarrow \pi[v'] + w_c \cdot f^c_{[i-1,i]}(x, l', l)$
    $t \leftarrow t - b(l')i(l) \cdot \Lambda^{BI}[i] - i(l')b(l) \cdot \Lambda^{IB}[i]$ ▷ only for DD: we ignore this line otherwise
    if $t > s_2$ **then**
      $s_2 = t$
    end if
  end for
  $\pi[v] \leftarrow s_1 + s_2$
end for
return the best scoring path $\pi[$ (</s>, STOP) $]$

4.2 Dual decomposition for CRF combinations

In Section 3.2 we described several annotation schemes that lead to different CRF models. These schemes give the same lexical segmentation information but they use more or less rich part-of-speech tag sets. It is not clear *a priori* if the richness of the tag set has a beneficial effect over segmentation prediction: there is a compromise between linguistic informativeness and data sparseness. Instead of trying to find the best annotation scheme, we propose a consensus-based joint system between several CRF-based sequence labellers for the task of text segmentation relying on dual decomposition (Rush et al., 2010). This system maximises the sum of the scores of combined CRFs, while enforcing global consistency between systems in terms of constraints over the admissible solutions. These constraints are specifically realised as reparametrisations of the elementary CRFs until a consensus is reached. Since we deal with several annotation schemes, we will use predicates to abstract from them:

- $b(l)$ is true if l starts with B;
- $i(l)$ is true if l starts with I;
- $bi(i, y)$ is true if $b(y_{i-1})$ and $i(y_i)$ are true;
- $ib(i, y)$ is true if $i(y_{i-1})$ and $b(y_i)$ are true.

For a labelling y, we define 2 boolean vectors that indicate where the compounds begin and end:

[2] We also include two additional nodes: an initial state (<s>, START) and a final state (</s>, STOP).
[3] Algorithm 1 calculates the score and backpointers must be added to retrieve the corresponding path.
[4] This algorithm takes as input 2 vectors that will be used for DD and will be explained in § 4.2. They can be ignored now.

- $D(y)$, such that $D(y)[i] = 1$ if $bi(i, y)$, and 0 otherwise;
- $F(y)$, such that $F(y)[i] = 1$ if $ib(i, y)$, and 0 otherwise.

As we want to *combine* CRFs, the solution of our system will be a *tuple* of label sequences with the same compound segmentation. For an input sequence x, this new maximisation problem is:

$$(P): \qquad \textit{find} \max_{(y^1,\dots,y^n)} \sum_{c=1}^{n} \sigma_c(y^c) = \sum_{c=1}^{n} \sum_{p\in\mathcal{P}(x)} w_c \cdot f_p^c(x, y_p^c) \tag{3}$$

$$\textit{s.t.}\ \exists u_1, u_2\, \forall c \in [\![1, n]\!],\ D(y^c) = u_1, F(y^c) = u_2 \tag{4}$$

Objective (3) indicates that we seek for a tuple for which the sum of the scores of its elements is maximal. Constraints (4) imply that the compound frontiers – transitions `B` to `I` and `I` to `B` – must be the same for each element of the tuple. There are several ways to tackle this problem. The first one is by swapping the sum signs in (3) and noticing that the problem could then be represented by a joint system relying on dynamic programming – a CRF for which labels would be elements of the product $\mathcal{L} = \mathcal{L}_1 \times \cdots \times \mathcal{L}_n$ – and for which it is straightforward to define a weight vector and feature functions. We can therefore reuse the Viterbi algorithm but the complexity is quadratic in the size of $\mathcal{L}$, which is impractical[5].

In any case, this approach would be inadequate for inclusion of parsers, and we therefore rely on lagrangian relaxation. We modify the objective by introducing Lagrange multipliers, two real vectors Λ_c^{BI} and Λ_c^{IB} indexed by bigram places[6] for each CRF c of the combination. We obtain a new problem with the same solution as the previous one, since constraints (4) are garanteed to be satisfied at optimality:

$$\max_{(y^1,\dots,y^n,u_1,u_2)} \min_{(\Lambda^{BI},\Lambda^{IB})} \sum_{c=1}^{n} \sigma_c(y^c) - \sum_{c=1}^{n} \left[(D(y^c) - u_1) \cdot \Lambda_c^{BI} + (F(y^c) - u_2) \cdot \Lambda_c^{IB}\right] \tag{5}$$

The next step is dualisation, which gives an upper bound of our problem:

$$\min_{(\Lambda^{BI},\Lambda^{IB})} \max_{(y^1,\dots,y^n,u_1,u_2)} \sum_{c=1}^{n} \sigma_c(y^c) - \sum_{c=1}^{n} D(y^c)\cdot\Lambda_c^{BI} + u_1 \sum_{c=1}^{n} \Lambda_c^{BI} - \sum_{c=1}^{n} F(y^c)\cdot\Lambda_c^{IB} + u_2 \sum_{c=1}^{n} \Lambda_c^{IB} \tag{6}$$

We then remark that $\sum_{c=1}^{n} \Lambda_c^{BI}$ and $\sum_{c=1}^{n} \Lambda_c^{IB}$ must be zeros at optimum (if the problem is feasible).[7] It is convenient to convert this remark to hard constraints in order to remove any reference to vectors u_i – and therefore to the coupling contraints – from the objective. We obtain the constrained problem with the same optimal solution :

$$(Du): \qquad \textit{find} \min_{(\Lambda^{BI},\Lambda^{IB})} \sum_{c=1}^{n} \max_{y^c} \left[\sigma_c(y^c) - D(y^c)\cdot\Lambda_c^{BI} - F(y^c)\cdot\Lambda_c^{IB}\right] \tag{7}$$

$$\textit{s.t.}\ \sum_{c=1}^{n} \Lambda_c^{BI} = 0 \text{ and } \sum_{c=1}^{n} \Lambda_c^{IB} = 0 \tag{8}$$

In order to solve (Du) we use the projected subgradient descent method that has already been used in many problems, for example MRF decoding (Komodakis et al., 2007). For the problem at hand, this method gives Algorithm 2. This iterative algorithm consists in reparametrising the elementary CRFs of the system, by modifying the weights associated with the bigram features in places that correspond to compound frontiers, penalising them on CRFs that diverge from the average solution. This is performed

[5] One could object that some combinations are forbidden. It remains that the number of labels still grows exponentially.

[6] Bigram places are identified by their second position.

[7] Otherwise the sum expressions would be unbounded and their maximum is $+\infty$ for an appropriate value of u_i.

by amending the vectors Λ_c^{BI} and Λ_c^{IB} that are updated at each iteration proportionally to the difference between the feature vectors for c and the average values of these vectors. Hence the farther a solution is from the consensus, the more penalised it gets at the next iteration. This algorithm stops when the updates are null for all CRFs: in this case the consensus is reached.

Algorithm 2 Best segmentation with combined CRF system via subgradient descent

Require: n CRF $c = (\Sigma, \mathcal{L}_c, w_c, \{f_p^c\}_p)$, an input sentence x, a maximum number of iterations τ, stepsizes $\{\alpha_t\}_{0 \leq t \leq \tau}$

for all CRF c, bigram end position i, bigram label pair (l, m) **do**
  $\Lambda_c^{BI}[i, l, m]^{(0)} = 0; \quad \Lambda_c^{IB}[i, l, m]^{(0)} = 0$
end for
for $t = 0 \rightarrow \tau$ **do**
  for all CRF c **do**
  $y^{c^{(t)}} = Viterbi(\mathcal{G}_c, w_c, f_c, \Lambda_c^{BI^{(t)}}, \Lambda_c^{IB^{(t)}})$
  end for
  for all CRF c **do**
    $\Delta_c^{BI^{(t)}} \leftarrow \alpha_t \left(D\left(y^{c^{(t)}}\right) - \frac{\sum_{1 \leq d \leq n} D\left(y^{d^{(t)}}\right)}{n} \right); \quad \Delta_c^{IB^{(t)}} \leftarrow \alpha_t \left(F\left(y^{c^{(t)}}\right) - \frac{\sum_{1 \leq d \leq n} F\left(y^{d^{(t)}}\right)}{n} \right)$
    $\Lambda_c^{BI^{(t+1)}} \leftarrow \Lambda_c^{BI^{(t)}} + \Delta_c^{BI^{(t)}}; \quad \Lambda_c^{IB^{(t+1)}} \leftarrow \Lambda_c^{IB^{(t)}} + \Delta_c^{IB^{(t)}}$
  end for
  if $\Delta_c^{BI^{(t)}} = 0$ and $\Delta_c^{IB^{(t)}} = 0$ for all c **then**
    Exit loop
  end if
end for
return $(y^{1^{(t)}}, \cdots, y^{n^{(t)}})$

We set the maximum number of iteration τ to 1000. For the step size, we use a common heuristic: $\alpha_t = \frac{1}{1+k}$ where k is the number of times that (Du) has increased between two successive iterations.

4.3 Experimental results for CRF combinations

We modified the `wapiti` software (Lavergne et al., 2010) with Algorithm 2. Table 1 reports segmentation results on the development set with the different tag sets, the best DD combination, and the best voting system.[8]

Tag Set CRF / combination	Recall	Precision	F-score
partial-internal	79.59	85.49	82.44
partial	78.98	85.57	82.14
basic	79.74	84.65	82.12
complete	79.69	83.10	81.36
complete-internal	79.03	82.66	80.80
`MWE` basic complete partial-internal	80.82	86.07	**83.36**
vote (basic complete partial-internal)	80.49	85.46	82.90

Table 1: Segmentation scores of CRF systems (dev)

System	F-score (all)	F-score (compounds)
complete	94.29	78.32
`MWE`	94.59	**80.00**

Table 2: Segmentation + POS tagging (dev)

We see that the best system is a combination of 3 CRFs (tag sets *basic*, *complete* and *partial-internal*) with DD, that we call `MWE` in the remaining of the paper. The subgradient descent converges on all instances in 2.14 iterations on average. The DD combination is better than the voting system.

We can also evaluate the POS tagging accuracy of the system for systems including the *complete* tag set. We compare the results of the *complete* CRF with the `MWE` combination on Table 2. The second column gives the F-score of the complete task, segmentation and POS tagging. The third column restricts the evaluation to compounds. Again, the `MWE` combination outperforms the single system.

[8] Each system has one vote and in case of a draw, we pick the best system's decision.

In some preliminary experiments, the weights of the CRF systems were based on unigram features mainly – i.e. those described in (Constant et al., 2012). As our CRFs are constrained on transitions from `B` to `I` and `I` to `B`, penalising systems resulted in modifying (low) bigram weights and had only a minor effect on the predictions and consequently the projected gradient algorithm was slow to converge. We therefore added bigram templates for some selected unigram templates, so that our system can converge in a reasonable time. Adding these bigram features resulted in slower elementary CRFs. On average the enriched CRFs were 1.8 times slower that their preliminary counterparts.

5 Dual Decomposition to combine parsers and sequence labellers

We now present an extension of the previous method to incorporate phrase-structure parsers in the combination. Our approach relies on the following requirement for the systems to agree: if the parser predicts a compound between positions i and j, then the CRFs must predict compound frontiers at the same positions. In this definition, like in previous CRF combinations, only the positions are taken into account, not the grammatical categories. From a parse tree a, we define two feature vectors:

- $D(a)$, such that $D(a)[i] = 1$ if a contains a subtree for a compound starting at position $i-1$
- $F(a)$, such that $F(a)[i] = 1$ if a contains a subtree for a compound ending at position $i-1$

In other words, $D(a)[i]$ indicates whether the CRFs should label position $i-1$ with `B` and position i with `I`, while $F(a)[i]$ indicates whether the CRFs should label position $i-1$ with `I` and position i with `B`. See Figure 2 for an example.

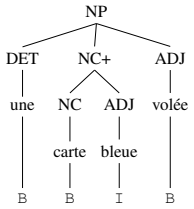


Figure 2: Parser and CRF alignments (A stolen credit card)

5.1 Parsing with probabilistic context-free grammars

We follow the type of reasoning we used in § 4.2. With a PCFG g, we can define the score of a parse for an input sentence x as the logarithm of the probability assigned to this parse by g. Finding the best parse takes a form analogous to the one in Equation 2, and we can write the CKY algorithm as a best path algorithm with penalties on nodes, as we did for the Viterbi algorithm previously. This is closely related to the PCFG combinations of (Le Roux et al., 2013). We introduce penalties through two real vectors Λ^{BI} and Λ^{IB} indexed by compound positions. The modified CKY is presented in Algorithm 3[9] where the parse forest $\mathcal{F}$ is assumed to be already available and we note w the vector of rule log-probabilities.

5.2 System combination

As in § 4.2, our problem amounts to finding a tuple that now consists of a parse tree and several labellings. All systems must agree on compound frontiers. Our objective is:

$$(P') : \quad \textit{find} \max_{(a,y^1,\ldots,y^n)} \sigma_p(a) + \delta \sum_{c=1}^{n} \sigma_c(y^c) \tag{9}$$

$$\textit{s.t.} \ \exists u_1, u_2 \, \forall c \in [\![1,n]\!], \, D(y^c) = u_1, F(y^c) = u_2, D(a) = u_1, F(a) = u_2 \tag{10}$$

[9] Without loss of generality, only binary rules are taken into account.

Algorithm 3 CKY with node penalties for compound start/end positions

CKY$(\mathcal{F}, w, \Lambda^{BI}, \Lambda^{IB})$:
for all node v in the forest $\mathcal{F}$ **do**
    $\pi[v] = -\infty$
end for
for all leaf node x **do**
    $\pi[x] = 0$
end for
for all non-terminal node (A, i, j) in topological order **do**
    for all incoming hyperedge $u = (B, i, k)(C, k+1, j) \rightarrow (A, i, j)$ **do**
        $s \leftarrow \pi[(B, i, k)] + \pi[(C, k+1, j)] + w_{A \rightarrow BC}$ ▷ $w_{A \rightarrow BC}$ is the score for rule $A \rightarrow BC$
        if A is a compound label **then**
            $s \leftarrow s - \Lambda^{BI}[i] - \Lambda^{IB}[j+1]$
        end if
        if $s > \pi[(A, i, j)]$ **then**
            $\pi[(A, i, j)] \leftarrow s$
        end if
    end for
end for
return hyperpath with score $\pi[(\text{ROOT}, 1, n)]$

We use δ to set the relative weights of the CRFs and the PCFG. It will be tuned on the development set. We then reuse the same procedure as before: lagrangian relaxation, dualisation, and projected subgradient descent. Algorithm 4 presents the function we derive from these operations.

Algorithm 4 Find the best segmentation with a PCFG and CRFs

Require: a PCFG parser p, n CRFs, an input sentence x, a bound τ
set Lagrange multipliers (penalty vectors) to zero
for $t = 0 \rightarrow \tau$ **do**
    for all CRF c **do**
        $y^{c^{(t)}} \leftarrow Viterbi(\mathcal{G}_c, w_c, f_c, \Lambda_c^{BI^{(t)}}, \Lambda_c^{IB^{(t)}})$
    end for
    $a^{(t)} \leftarrow CKY(\mathcal{F}, w, \Lambda_p^{BI^{(t)}}, \Lambda_p^{IB^{(t)}})$
    for all CRF c and parser p **do**
        Update penalty vectors proportionally to the difference between corresponding solution and average solution
    end for
    if *update* is null for all c and p **then**
        Exit loop
    end if
end for
return $(a^{(t)}, y^{1^{(t)}}, \cdots, y^{n^{(t)}})$

Algorithm 4 follows the method used in § 4 and simply adds the PCFG parser as another subsystem. This method can then be extended further: for instance, we can add a POS tagger (Rush et al., 2010) or multiple PCFG parsers (Le Roux et al., 2013). Due to lack of space, we omit the presentation of these systems, but we experiment with them in the following section.

6 Experiments

For this series of experiments, we used `wapiti` as in § 4.3 and the `LORG` PCFG-LA parser in the configuration presented in (Le Roux et al., 2013) that we modified by implementing Algorithm 4. This parser already implements a combination of parsers based on DD, a very competitive baseline.

For parse evaluation, we used the EVALB tool, modified by the SPMRL 2013 organisers, in order to compare our results with the shared task participants. We evaluated several configurations: (i) the `LORG` parser alone, a combination of 4 PCFG-LA parsers as in (Le Roux et al., 2013), (ii) a pipeline of `POS`, a CRF-based POS tagger, and `LORG`, (iii) joint `LORG` and `POS`, using DD as in (Rush et al., 2010), (iv) joint `LORG` and `MWE` (our best CRF combination for compound segmentation) using DD, and (v) joint `LORG`, `POS` et `MWE` using DD. We also compare these architectures with 2 additional pipelines, in which we first run `MWE` and then merge compounds as single tokens. The converted sentences are then sent to a version of `LORG` learnt on this type of corpus. After parsing, compounds are *unmerged*,

replaced with the corresponding subtree. In one of these two architectures, we add a POS tagger.

The evaluations for the parsing task of all these configurations are summarised in Table 3. The best system is the DD joint system combining the POS tagger, the parser and the compound recognisers.

System	Recall	Precision	Fscore	EX	Tag
LORG	82.01	82.37	82.19	18.06	97.35
pipeline POS → LORG	82.36	82.59	82.47	19.22	97.73
DD POS + LORG	82.48	82.73	82.61	19.19	97.84
DD MWE + LORG	82.91	83.07	82.99	19.19	97.41
DD POS + MWE + LORG	83.38	83.42	**83.40**	20.73	97.85
pipeline MWE MERGE → LORG → UNMERGE	82.56	82.63	82.59	18.79	97.39
pipeline MWE MERGE/POS → LORG → UNMERGE	82.73	82.64	82.69	20.02	97.57

Table 3: Parse evaluation on *dev* set (recall, precision and F-score, exactness and POS tagging).

Table 4 shows evaluation results of our best system and comparisons with baseline or alternative configurations on the SPMRL 2013 *test* set.

Parsing The DD method performs better than our baseline, and better than the best system in the SPMRL 2013 shared task (Björkelund et al., 2013). This system is a pipeline consisting of a morpho-syntactic tagger with a very rich and informative tag set, a product of PCFG-LAs, and a parse reranker. Although this approach is quite different from ours, we believe our system is more accurate overall because our method is more resilient to an error from one of its components.

Compound recognition and labelling For the task of recognition alone, where only the frontiers are evaluated, the DD combinations of CRFs performs better than the best single CRF which itself performs better than the parser alone, but the complete architecture is again the best system. If we also evaluate compound POS tags, we get similar results. The DD combination is always beneficial.

System	Recall	Precision	Fscore	EX	Tag
LORG	82.79	83.06	82.92	22.00	97.39
(Björkelund et al., 2013)	–	–	82.86	–	–
DD POS + MWE + LORG	83.74	83.85	**83.80**	23.81	97.87
compound recognition LORG	78.03	78.63	78.49	–	–
compound recognition best single CRF (partial-internal)	78.27	82.84	80.49	–	–
compound recognition MWE	79.68	83.50	81.54	–	–
compound recognition DD POS + MWE + LORG	80.76	84.19	**82.44**	–	–
compound recognition + POS tagging LORG	75.43	75.71	75.57	–	–
compound recognition + POS tagging MWE	76.49	80.10	78.28	–	–
compound recognition + POS tagging DD POS + MWE + LORG	77.92	81.23	**79.54**	–	–

Table 4: Evaluation on SPMRL 2013 *test* set: parsing (first 3 lines), and compound recognition.

7 Conclusion

We have presented an original architecture for the joint task of syntactic parsing and compound recognition. We first introduced a combination of recognisers based on linear chain CRFs, and a second system that adds in a phrase-structure parser. Our experimental prototype improves state-of-the-art on the French SPMRL corpus.

In order to derive decoding algorithms for these joint systems, we used dual decomposition. This approach, leading to simple and efficient algorithms, can be extended further to incorporate additional components. As opposed to pipeline approaches, a component prediction can be *corrected* if its solution is too far from the general consensus. As opposed to joint systems relying on pure dynamic programming to build a complex single system, the search space does not grow exponentially, so we can avoid using pruning heuristics such as beam search. The price to pay is an iterative algorithm.

Finally, this work paves the way towards component-based NLP software systems that perform complex processing based on consensus between components, as opposed to previous pipelined approaches.

Acknowledgements

We would like to thank Nadi Tomeh and Davide Buscaldi for their feedback on an early draft of this paper, as well as the three anonymous reviewers for their helpful comments. This work is supported by a public grant overseen by the French National Research Agency (ANR) as part of the *Investissements d'Avenir* program (ANR-10-LABX-0083).

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for French. In Anne Abeillé, editor, *Treebanks*. Kluwer, Dordrecht.

Abhishek Arun and Frank Keller. 2005. Lexicalization in crosslinguistic probabilistic parsing: The case of French. In *Proceedings of the Annual Meeting of the Association For Computational Linguistics (ACL'05)*, pages 306–313.

Anders Björkelund, Richárd Farkas, Thomas Müller, and Wolfgang Seeker. 2013. (re) ranking meets morphosyntax: State-of-the-art results from the spmrl 2013 shared task. In *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically Rich Languages: Shared Task*.

Matthieu Constant and Isabelle Tellier. 2012. Evaluating the impact of external lexical resources into a CRF-based multiword segmenter and part-of-speech tagger. In *Proceedings of the 8th conference on Language Resources and Evaluation*.

Matthieu Constant, Anthony Sigogne, and Patrick Watrin. 2012. Discriminative strategies to integrate multiword expression recognition and parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL'12)*, pages 204–212.

Matthieu Constant, Marie Candito, and Djamé Seddah. 2013a. The LIGM-Alpage Architecture for the SPMRL 2013 Shared Task: Multiword Expression Analysis and Dependency Parsing. In *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically Rich Languages: Shared Task*, Seattle, WA.

Matthieu Constant, Joseph Le Roux, and Anthony Sigogne. 2013b. Combining compound recognition and PCFG-LA parsing with word lattices and conditional random fields. *ACM Transaction in Speech and Language Processing*, 10(3).

Blandine Courtois, Mylène Garrigues, Gaston Gross, Maurice Gross, René Jung, Michel Mathieu-Colas, Anne Monceaux, Anne Poncet-Montange, Max Silberztein, and Robert Vivés. 1997. Dictionnaire électronique DELAC : les mots composés binaires. Technical Report 56, University Paris 7, LADL.

Jenny Rose Finkel and Christopher D. Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 326–334, Stroudsburg, PA, USA. Association for Computational Linguistics.

Spence Green, Marie-Catherine de Marneffe, John Bauer, and Christopher D. Manning. 2011. Multiword expression identification with tree substitution grammars: A parsing tour de force with french. In *Proceedings of the conference on Empirical Method for Natural Language Processing (EMNLP'11)*, pages 725–735.

Spence Green, Marie-Catherine de Marneffe, and Christopher D Manning. 2013. Parsing models for identifying multiword expressions. *Computational Linguistics*, 39(1):195–227.

Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. 2007. MRF optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, pages 504–513.

Joseph Le Roux, Antoine Rozenknop, and Jennifer Foster. 2013. Combining PCFG-LA models with dual decomposition: A case study with function labels and binarization. In Association for Computational Linguistics, editor, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, October.

Slav Petrov. 2010. Products of random latent variable grammars. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies*, pages 19–27.

Odile Piton, Denis Maurel, and Claude Belleil. 1999. The Prolex Data Base : Toponyms and gentiles for NLP. In *Proceedings of the Third International Workshop on Applications of Natural Language to Data Bases (NLDB'99)*, pages 233–237.

Alexander Rush, David Sontag, Michael Collins, and Tommi Jaakola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In ACL, editor, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Benoît Sagot. 2010. The lefff, a freely available, accurate and large-coverage lexicon for french. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*.

Djamé Seddah, Reut Tsarfaty, Sandra K$\acute{}$ubler, Marie Candito, Jinho Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiorkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clérgerie. 2013. Overview of the spmrl 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically Rich Languages*, Seattle, WA.

Veronica Vincze, István Nagy, and Gábor Berend. 2011. Multiword expressions and named entities in the wiki50 corpus. In *Proceedings of the conference on Recent Advances in Natural Language Processing (RANLP'11)*, pages 289–295.

Learning the Taxonomy of Function Words for Parsing

Dongchen Li, Xiantao Zhang, Dingsheng Luo and Xihong Wu
Key Laboratory of Machine Perception and Intelligence
Speech and Hearing Research Center
Peking University, Beijing, China
{lidc,zhangxt,dsluo,wxh}@cis.pku.edu.cn

Abstract

Completely data-driven grammar training is prone to over-fitting. Human-defined word class knowledge is useful to address this issue. However, the manual word class taxonomy may be unreliable and irrational for statistical natural language processing, aside from its insufficient linguistic phenomena coverage and domain adaptivity. In this paper, a formalized representation of function word subcategorization is developed for parsing in an automatic manner. The function word classification representing intrinsic features of syntactic usages is used to supervise the grammar induction, and the structure of the taxonomy is learned simultaneously. The grammar learning process is no longer a unilaterally supervised training by hierarchical knowledge, but an interactive process between the knowledge structure learning and the grammar training. The established taxonomy implies the stochastic significance of the diversified syntactic features. The experiments on both Penn Chinese Treebank and Tsinghua Treebank show that the proposed method improves parsing performance by 1.6% and 7.6% respectively over the baseline.

1 Introduction

Probabilistic context-free grammar (PCFG) is widely used in the fields of speech recognition, machine translation, information retrieval, etc. It takes the empirical rules and probabilities from a Treebank. However, due to the context-free assumption, PCFG does not always perform well (Klein and Manning, 2003). For instance, it assumes adverbs, including temporal adverbs, degree adverbs and negation adverbs, to share the same distribution, whereas the distinction would provide useful indication for disambiguating the syntactic structure of the context.

It arose that the manual word classification in linguistic research was used to enrich PCFG and improve the performance. However, from the point of view of statistical natural language processing, there are some drawbacks for manual classification. Firstly, Linguistic phenomena covered by the manual refinement may be limited by the linguistic observations of human. Secondly, the evidence of manual refinement is often based on a particular corpus or specific sources of knowledge acquisition. As a result, its adaptivity to different domains or genres may be insufficient. As for function words, due to the ambiguity and complexity in syntactic grammar, it is more difficult to develop formalized representation than for content words. There are diversified standards for grammar refinement. Consequently, the word classification or category refinement can be conducted in distinct manners, while each of them is reasonable in some sense. A delicate hierarchical classification inevitably involves in multiple dividing standards. However, the word sets under distinct dividing standards may be overlapping. The problems come up that how to choose the set of the multiple standards to cooperate to build the taxonomy, and how to decide the priority of each standard. Regarding that the manual method is hard to overcome critical issues, manual taxonomy for function words may not be reliable for statistical natural language processing.

This article attempts to address these issues in a data-driven manner. we first manually construct a cursory and flat classification of function words. A hierarchical split-merge approach is employed to

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1886–1896, Dublin, Ireland, August 23-29 2014.

introduce our classification, and the PCFG training procedure is supervised to alleviate the over-fitting issue. The priorities of the subcategorization standards are determined by the measurement of effectiveness for parsing in a greedy manner in the hierarchical classification. And the hierarchical structure of the classification is learned by data-driven approach in the course of grammar induction, so as to fit the practical usages in the Treebank. Accordingly, the grammar learning process is no longer a unilaterally supervised training by hierarchical knowledge, but an interactive process between the knowledge representation induction and the grammar training. That is, the grammar induction is supervised by the knowledge and the structure of the taxonomy is learned simultaneously. These two processes are iterated for several rounds and the hierarchical structure of the function word taxonomy is constructed. In each round, the induced grammar could benefit from the optimized taxonomy during the learning process. The category split in the early rounds take more priorities than in the late ones. Thus, the learned taxonomy implies the stochastic significance of the series of the syntactic features.

Experiments on Penn Chinese Treebank Fifth Edition (CTB5.0) (Xue et al., 2002) and Tsinghua Chinese Treebank (TCT) (Zhou, 2004) are performed. The results show that the induced grammars with refined conjunction categories gain parsing performance improvement by 1.6% on CTB and by 7.6% on TCT. During the training process, a taxonomy of function words is learned, which reflects their practical usages in the corpus.

The rest of this paper is organized as follows. We first review related work on category refinement for parsing. Then we describe our manually defined categories of function words in Section 3. The hierarchical state-split approach for introducing the the categories are presented in Section 4, and our taxonomy learning method is described in Section 5. In Section 7, experimental comparison is conducted among various methods on granularity choosing. And conclusions of this research are drawn in last section.

2 Related Work

A variety of techniques have been proposed to enrich PCFG in either manual (Klein and Manning, 2003; Zhang and Clark, 2011) or automatic (Petrov, 2009; Cohen et al., 2012) manner.

2.1 Automatic Refinement of Function Words for Parsing

One way of grammar refinement is data-driven state-split methods (Matsuzaki et al., 2005; Prescher, 2005). The part-of-speech and syntactic tags in the grammar are automatically split to encode the kinds of linguistic distinctions exhibited in the Treebank. The hierarchical state-split approach (Petrov et al., 2006) started from a bare-bones Treebank derived grammar, and iteratively refined it in a split-merge-smooth cycle with the EM-based parameter re-estimation. It achieved state of the art accuracies for many languages including English, Chinese and German.

One tag is usually heterogeneous, in the sense that its word set can be of multiple different types. Nevertheless, the automatic process tries to split the tags through a greedy data-driven manner, where multiple distinctive information is used simultaneously when dividing tags. Thus the refined tags are not intuitively interpretable. Meanwhile, considering that the EM algorithm usually gets stuck at a sub-optimal configuration, this data-driven method suffers from the risk of over-fitting. As shown in their experiments, there is little to be gained from splitting the closed part-of-speech classes (e.g. DT, CC, IN) or the nonterminal ADJP.

To alleviate the risk of over-fitting, we employ the human-defined knowledge to constrain the splitting process in this research. Based on the state-split model, our approach aims to reach a compromise between manual and automatic refinement approaches.

2.2 Manual Refinement of Function Words for Parsing

The other way to refine the annotation for training a parser is incorporating knowledge base. Semantic knowledge of content words has been proved to be effective in alleviate the data sparsity. Some researches utilized semantic knowledge in WordNet (Miller, 1995; Fellbaum, 1999) for English parsing (Fujita et al., 2010; Agirre et al., 2008), and Xiong et al. (2005; Lin et al. (2009) improved Chinese pars-

ing by incorporating semantic knowledge in HowNet (Dong and Dong, 2003; Dong and Dong, 2006). While WordNet and Hownet contain word classification for content words, Li et al. (2014b; Li et al. (2014a) have focused on exploiting manual classification for conjunction in parsing.

Klein and Manning (2003) examined the annotation in Penn English Treebank, manually split the majority of the part-of-speech (POS) tags. For the function words, they split the tag "IN" into subordinating conjunctions, complementizers and prepositions, and appended B̂E to all forms of "be" and ĤAVE to all forms of "have". Conjunction tags are also marked to indicate whether they were "But", "but" or "&". The experimental results showed that the split tags of function words surprisingly make much contribution to the overall improved parsing accuracy. Levy and Manning (2003) transferred this work to Penn Chinese Treebank. They found that, in some cases, certain adverbs such as "however (然而)" and "especially (尤其是)" preferred IP modification and could help disambiguate IP coordination from VP coordination. To capture this point, they marked those adverbs possessing an IP grandparent. However, these manual refinement methods seems to split the tags in a rough way, which might account for a modest accuracy achieved. Some existing work used heuristic rules to simply split the tags of function words (Klein and Manning, 2003; Levy and Manning, 2003). They demonstrated that many function words stood out to be helpful in predicting the syntactic structure and syntactic label.

3 Manual Tabular Subcategories of Function Words

When subcategorizing function words, in this section, we manually list various grammatical distinctions that are commonly made in traditional and generative grammar in a fairly flat taxonomy. The grammar training procedure learns by using our manual taxonomy as a starting point, and constructs a reasonable and subtle hierarchical strucutre based on the distribution of function words usages in the corpus.

Based on some existing knowledge base (Xia, 2000; Xue et al., 2000; Zhu et al., 1995; Wang and Yu, 2003) and previous research work (Li et al., 2014b), we investigate and summarize the usage of function words, and come up with a hierarchical subcategories. The taxonomy of the function words is represented in a tree structure, where each subcategory of a function word corresponds to a node in the taxonomy, the nonterminals are subcategories and the terminals are words.

For the convenience and consistence, our manual classification just gives a rough and broad taxonomy. It is labor-intensive and error-prone of classifying the function words manually to produce a consistent output. Fine-grained hierarchical structure is not obligatory, but would be harmful if inappropriately classified, as it may mislead the learning process. To avoid this kind of risk, the elaboration is saved, rather than introducing unnecessary bias. The learning process would perform the hierarchical classification according to the distribution in the corpus.

For instance, the distinction within conjunctions is intricate. Conjunctions are the words that are called "connective words" in traditional Chinese grammar books. In Penn Chinese Treebank, they are tagged as coordinating conjunctions (CC), subordinating conjunctions (CS), or adverbs (AD) according to their syntactic distribution. CC conjoins two equivalent constituents (noun phrases, clauses, etc.), each of which has approximately the same function as the whole construction. CS precedes a subordinating clause, whereas conjunctive adverbs often appear in the main clause and pair with a subordinating conjunction (e.g., if (如果)/CS ... then (就)/AD). However, in Chinese, it is often hard to tell the subordinating clause from the main clause in the compound statement. As a result, in the prospective of linguistic computing, the confusion is that, CS and conjunctive adverbs both precedes the subordinating clauses or main clauses, while CC connects two phrases or precedes the main clause. In our scheme, we simply conflates the CC, CS and conjunctive adverbs together. This result in a general "conjunction" category, within which we just enumerate all the possible uses of the conjunctions. As a result, the structure of our human-defined taxonomy is fairly flat, as briefly shown in Figure 1 and Figure 2. Our scheme releases our hands from the confusing situations, by leaving them to our data-driven method described in the following section. Figure 1 and Figure 2 abbreviate the manual classification and their corresponding examples.

Many prepositions in Chinese are evolved from verbs, thus the linguistic characteristics of prepositions are somewhat similar to verbs. Therefore, this paper divides the preposition word set according to

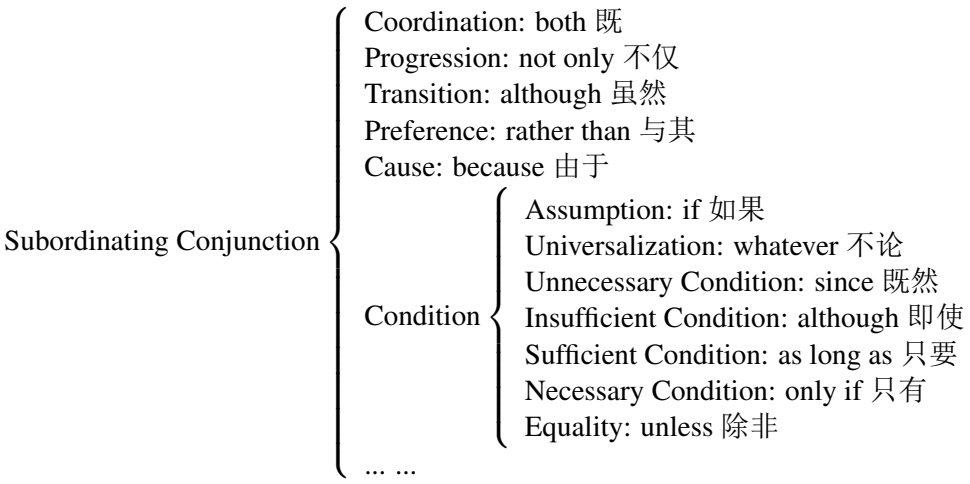


Figure 1: Abbreviated Hierarchical subcategories of subordinating conjunctions with examples.

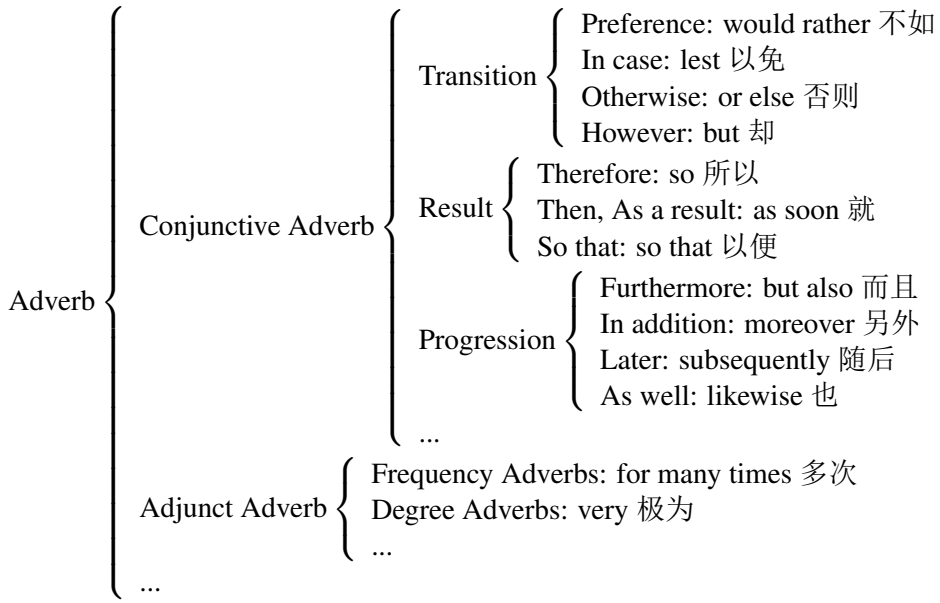


Figure 2: Abbreviated Hierarchical subcategories of adverbs with examples.

the types of their associated arguments: “benefactive”, such as “ 为(for)” and “给(to)”, marks the beneficiary of an action; “locative”, such as “在(in)”, marks adverbials that indicate the place of the event; “direction”, such as “ 向(towards)” and “ 由(from)”, marks adverbials that answer the questions “from where?” and “to where?”; “temporal”, such as “在(on)”, marks temporal or aspectual adverbials that answer the question “when?”, and so on.

4 Refining Grammar with Hierarchical Category Refinement

In this section, we choose the appropriate granularity in a data-driven manner based on the split-merge learning method in Section 2.1. Our approach first initializes the categories with the most general subcategories in the taxonomy and then splits the categories through the hypernym-hyponym relation in the

taxonomy. Data-driven method is used to merge the overly refined subcategories.

The top category in the taxonomy is used as the starting annotations of POS tags. As we cannot predict which layer should be the most adequate one, we try to avoid applying any priori restriction on the refinement granularity, and start with the most general tags.

With the hierarchical knowledge, it turns out to be a critical issue that which granularity should be used to refine the tags for parsing. We intend to take neither too coarse subcategories nor too fine ones in the hierarchical knowledge for parsing. Instead, it would be our advantage to split the tags with the very granularity where needed, rather than splitting them all to one specific granularity in the taxonomy.

For example, "Conjunctive Adverbs" are divided into three subcategories in our taxonomy as shown in Figure 2. The evidence for the refinement may occur in very rare case, and certainly some of the context of the different subcategories are quite the same. Splitting symbols with the same context is not only unnecessary, but potentially harmful, since it unreasonably fragments observations of other symbols' behavior.

In this paper, the hierarchical subcategory knowledge is used to refine grammars by supervising the automatic hierarchical state-split approach. In the split stage in each cycle, the function word subcategory is split along the hierarchy of the knowledge, instead of being randomly split and classified automatically. In this way, we try to alleviate the over-fitting of the greedy data-driven approach, and a new set of knowledge-related tags are generated. In the following step, we retreat some of the split subcategories to their more general layer according to its likelihood loss of merging them. In this way, we try to avoid the excessive refinement in our hierarchical knowledge without sufficient data support.

There are two issues that we have to consider in this process: a) how to deal with the polysemous words, and b) how to deal with the multi-branch situation other than binary branch in the taxonomy. Regarding to the polysemous words, they occur mostly in two situation for function words. Some are the polysemous words which can be taken as conjunctions or auxiliary words, while the others can be taken as preposition or adverbs. Fortunately there is no ambiguity for a word given its POS tag, so we could neglect this situation in the split process when training. We demonstrated the solution for the multiple branches in the Section 5.

5 Learning the Taxonomy of Function Words

There are multiple subcategorization criterions for building function word taxonomy, and it is difficulty for human to rank the ordering in the classification process. This section represents the method of learning the taxonomy of the function words in data-driven manner. Based on the manual tabular classification, the similar word classes are conflated to express the data distribution.

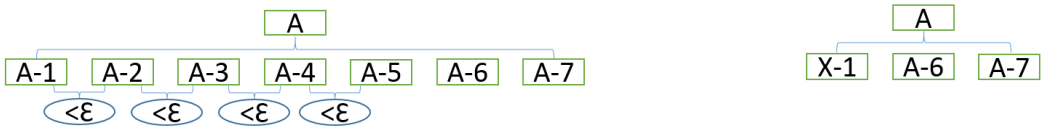
The multiple branches in the taxonomy are intractable for the original split-merge method, because it splits every category into two and merges half of them for efficiency. If we follow this scheme in our training process, it would be difficult to deal with the multi-branch situation in the taxonomy, because how to choose the first two to split among the multiple branches is another challenge. It is an equally difficult problem for us to binarize the taxonomy by hand comparing to directly choosing the granularity.

It would be our advantage to binarize the taxonomy by a data-driven method. For automatic binarization, a straightforward approach is to measure the utility of traversing all the plausible ways of cutting all the branches into two sets individually and use the best one. Then we can deal with the divided two sets in the same manner recursively. However, not only is this impractical, requiring an entire training phase for each possible binarization scheme which is exponentially expensive, but it assumes the contributions of multiple binarizations in different branches are independent. In fact, extra sub-symbols may need to be added to several nonterminals before they can cooperate to pass information along the parse tree.

Therefore, we go in the opposite direction, and propose an extended version of split-merge learning to handle the multiple branches in the taxonomy. That is, we split each state into all the subcategories in the lower layer in the taxonomy even if it has multiple branches, train, and then measure for every two sibling subcategories in the same layer the loss in likelihood incurred when merging them. If this loss is small, the new division of these two subcategories does not carry enough useful information and can be merged back. Contrary to the gain in likelihood for splitting, the loss in likelihood for merging can be

efficiently approximated (Petrov et al., 2006).

More specifically, we assume transitivity in merging multiple subcategories in one layer. Figure 3 gives an illustration. After the split stage, the category A has been split into subcategories A-1, A-2, ... to A-7. Then we compute the loss in likelihood of the training data by merging back each pair of two subcategories through A-1 to A-7. If the loss is lower than a certain threshold [1] set for each round of merge, this pair of newly split subcategories will be merged. We only show the sibling ones for brevity in this example. Assume the losses of merging these pairs (A-1, A-2), (A-2, A-3), (A-3, A-4) and (A-4, A-5) are below the threshold ε. Thus, A-1, A-2, A-3, A-4 and A-5 are merged to X-1 due to the transitivity of the connected points, where X-1 is the automatically generated subcategory which contains the five conflated subcategories as its descendants. At the meantime, A-6 and A-7 still remain. This scheme is an approximation because it merges subcategories that should be merged with the same subcategory. But it will leave the split of this instances to the next round when more evidence on interaction with other more refined subcategories is given.



(a) Refined subcategories before the merge stage

(b) Refined subcategories after the merge stage

Figure 3: Illustration of merging the subcategories for multiple branches in the taxonomy. Where ε is a certain threshold below which this pair of subcategories will be merged, and X is the automatically generated subcategory which contains the conflated subcategories as its descendants.

After merging in each round, the hierarchical knowledge is reshaped to fit the practical usage in the Treebank. The split-merge cycles allow us to progressively increase the complexity of the hierarchical knowledge, and the more useful distinctions are represented as the higher level in the taxonomy, which gives priority to the most useful distinctions in return by supervising the grammar induction. Figure 4 demonstrates the transformation of the hierarchical structure from the tabular classification. Along this road, the training scheme is not a unilateral training, but an interactive process between the knowledge representation learning and the grammar training. Our learning process exerts a mutual effect to both the induced grammar and the optimized structure of the hierarchical knowledge. In this way, the set of dividing standards are chosen iteratively according to their syntactic features. The more effective divisions are conducted in the early stages. In the following stages, the divisions which interact with previous divisions to give the most effective disambiguating information are adopted. The final taxonomy are built based on manual classification in data-driven approach, and the hierarchical structure are optimized and rational in the perspective of actual data distribution. Figure 4 illustrates a concrete instance of the procedure of learning the taxonomy. On one hand, this procedure provides a more rational hierarchical subcategorization structure according to data distribution. On the other hand, the order of the division criterions represents the priorities the grammar induction takes for each criterion. The structure in the higher levels of the taxonomy are determined by the dominant syntactic characteristics. And the division in the later iterations are on the basis of minor distinctive characteristics.

6 Experiments and Results

6.1 Data Set

We present experimental results on both CTB5.0 (All traces and functional tags were stripped.) and TCT.

We ran experiments on CTB5.0 using the standard data allocation: files from CHTB_001.fid to CHTB_270.fid, and files from CHTB_400.fid to CHTB_1151.fid were used as training set. The development set includes files from CHTB_301.fid to CHTB_325.fid, and the test set includes files CHTB_271.fid

[1] In practice, instead of setting a predefined threshold for merging, we merge a specific number of the newly split subcategories.

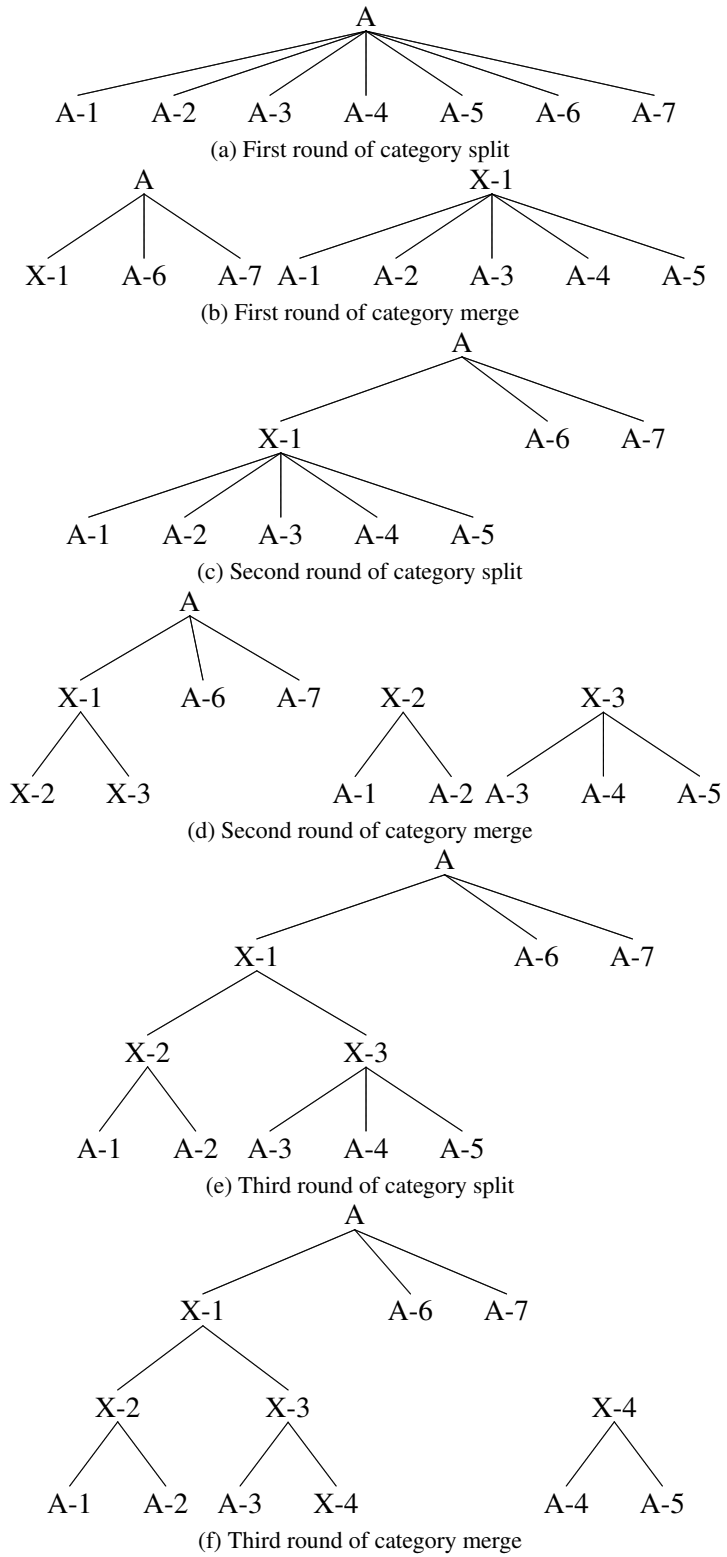


Figure 4: Iteration of grammar induction and taxonomy structure learning

to CHTB_300.fid. Experiments on TCT use the data set as in CIPS-SIGHAN-ParsEval-2012 (Zhou, 2012). We have parsed on the segmented text in the Treebank, namely, no use of gold POS-tags, use of gold segmentations, and full-length sentences. This is the same as for other 5 parsers in Table 1 for comparison. All the experiments were carried out after six cycles of split-merge.

6.2 Final Results

The final results are shown in Table 1. Our final parsing performance is higher than both the manual annotation method (Levy and Manning, 2003) and the data-driven method (Petrov, 2009).

Parser	Precision	Recall	F_1
Levy(2003)	78.40	79.20	78.80
Petrov(2009)	84.82	81.93	83.33
Lin(2009)	86.00	83.10	84.50
Qian(2012)	84.57	**83.68**	84.13
Zhang(2013)	84.42	84.43	84.43
This paper	**86.55**	83.41	**84.95**

Table 1: Our final parsing performance compared with the best previous work on CTB5.0.

On test set TCT, the method achieves the best precision, recall and F-measure in the CIPS-SIGHAN-ParsEval-2012 competition, and table 2 compares our results with the system of Beijing Information Science and Technology University (BISTU) which got the second place in the competition.

Parser	Precision	Recall	F_1
BISTU	70.10	68.08	69.08
This paper	**76.81**	**76.66**	**76.74**

Table 2: Our final parsing performance compared with the best previous works on TCT.

Given the manual labor required for generating the taxonomy (and in languages where there is a taxonomy, determining whether it is suitable), this first study focuses on a language where there is quite a bit of under- and over-specification in the Treebanks' tag sets. So this work is only implemented on Chinese. We regard it as future work to transfer this approach to other languages.

6.3 Analysis

The outline of constructing the taxonomy of function words are as follows. Firstly, the function words are manually subcategorized in a rough and cursory way. When dealing with subcategories hard to resolve their relation of subordination, we simply treat them as siblings in the tree in a rather flat stricture, and leave the elaboration of exquisite clustering to the algorithms. The data-driven approach in Section 4 automatically choose the appropriate granularity of refinement for our grammar. Moreover, the split-merge learning for multiple branches in the hierarchical subcategories in Section 5 exploits the relationship between the sibling nodes in the same layer, making use of the Treebank data to adjust and optimize the hierarchy.

During the split-merge process, the hierarchical subcategories are learned to fit the data, which is a transformation of our manually defined hierarchy. The transformed hierarchy is just the route map of subcategories employed in our model. As abbreviated in Figure 5 and Figure 6, many distinctions between word sets of the subcategories have been exploited by our approach, and the learned taxonomy is interpretable. For instance, It shows that the learned structure of the taxonomy is reasonable.

6.4 Comparison with Previous Work

Although the taxonomy of function words are learned in the grammar training process, the grammar is trained on the Treebank in supervised manner. Thus, this work is not directly relevant with unsupervised grammar induction literature (Headden III et al., 2009; Berant et al., 2007; Mareček and Žabokrtskỳ, 2014).

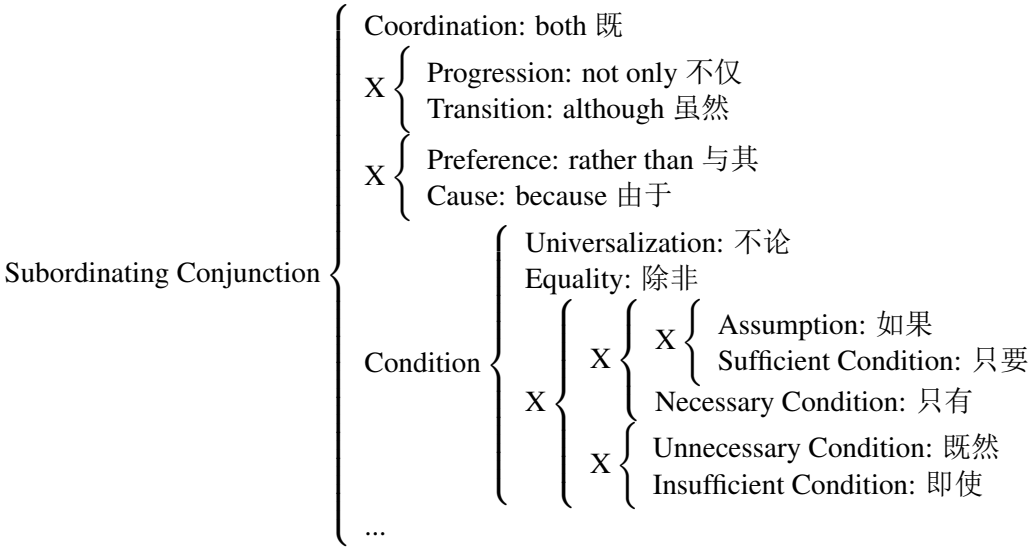


Figure 5: Abbreviated automatically learned hierarchical subcategories of subordinating conjunctions with examples. Where "X" represents the automatically generated subcategory.

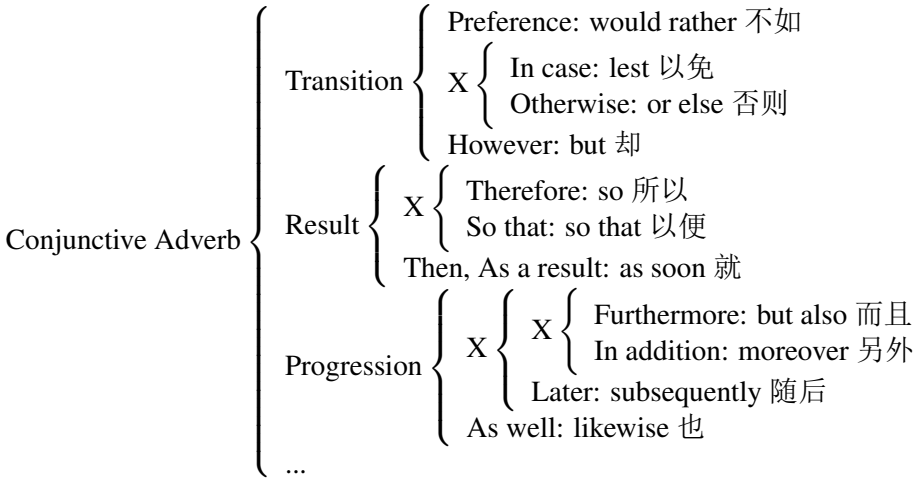


Figure 6: Abbreviated automatically learned hierarchical subcategories of adverbs with examples.

Lin et al. (2009) and Li et al. (2014b) presented ideas of using either hierarchical semantic knowledge from HowNet for content words or grammar knowledge for subordinating conjunctions. They introduced hierarchical subcategory knowledge in a different stage. They split the original Treebank categories in split-merge process according to the data, and then find a method to map the subcategories to the node in the taxonomy, and constrain their further splitting. Comparing to their work, our approach is more delicate, which is splitting the categories according to the knowledge, and learning the knowledge structure according to data during the training course. Lin et al. (2009) incorporated semantic knowledge of content words into the data-driven method. It would be promising if this work stacks with the content word knowledge. However, the work with content word knowledge have to handle the polysemous words in the semantic taxonomy, so they split the categories according to the data, and then find a way to map the subcategories to the node in the taxonomy, and constrain their further splitting. It is our goal to make these two methods compatible with each other.

Incorporating word formation knowledge achieved higher parsing accuracy according to Zhang and

Clark (2011). However, they ran their experiment on gold POS-tags and a different data set split, which is different form the setup of work in Table 1 including this work. They also presented their result on automatically assigned POS-tags and the same data set split as in the work in Table 1 to facilitate the performance comparison. It gave F_1 score of 81.45% for sentences with less than 40 words and 78.3% for all sentences, significantly lower than Petrov and Klein (2007).

Zhang et al. (2013) exhaustively exploited character-level syntactic structures for words, and achieved 84.43% on F_1 measure. They placed more emphasis on the word-formation of content words, which our model highlights the value of the function words. The complementary intuitions make it possible to integrate these approaches together in the future work.

7 Conclusion

This paper presents an approach for inducing finer syntactic categories while learning the taxonomy for function words. It used linguistic insight to guide the state-split process, and the hierarchical structure representing syntactic features of function word usages was established during the grammar training process. Empirical evidence has been provided that automatically subcategorizing function words contributes to high parsing performance. The induced grammar supervised by the taxonomy outperformed pervious approaches, which benefited from both the knowledge and the data-driven method. The proposed approach for learning the structure of the taxonomy could be generalized to construct semantic knowledge base.

Acknowledgments

This work was supported in part by the National Basic Research Program of China (973 Program) under grant 2013CB329304, the Research Special Fund for Public Welfare Industry of Health under grant 201202001, the Key National Social Science Foundation of China under grant 12&ZD119, the National Natural Science Foundation of China under grant 91120001.

References

Eneko Agirre, Timothy Baldwin, and David Martinez. 2008. Improving parsing and pp attachment performance with sense information. *Proceedings of ACL-08: HLT*, pages 317–325.

Jonathan Berant, Yaron Gross, Matan Mussel, Ben Sandbank, Eytan Ruppin, and Shimon Edelman. 2007. Boosting unsupervised grammar induction by splitting complex sentences on function words. In *Proceedings of the Boston University Conference on Language Development*.

Shay B Cohen, Karl Stratos, Michael Collins, Dean P Foster, and Lyle Ungar. 2012. Spectral learning of latent-variable pcfgs. In *Proceedings of the 50th annual meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 223–231. Association for Computational Linguistics.

Zhendong Dong and Qiang Dong. 2003. Hownet-a hybrid language and knowledge resource. In *Proceedings of the international conference on natural language processing and knowledge engineering*, pages 820–824. IEEE.

Zhengdong Dong and Qiang Dong. 2006. *HowNet and the computation of meaning*. World Scientific Publishing Co. Pte. Ltd.

Christiane Fellbaum. 1999. *WordNet*. Wiley Online Library.

Sanae Fujita, Francis Bond, Stephan Oepen, and Takaaki Tanaka. 2010. Exploiting semantic information for hpsg parse selection. *Research on language and computation*, 8(1):1–22.

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

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st annual meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics.

Roger Levy and Christopher D Manning. 2003. Is it harder to parse chinese, or the chinese treebank? In *Proceedings of the 41st annual meeting on Association for Computational Linguistics-Volume 1*, pages 439–446. Association for Computational Linguistics.

Dongchen Li, Xiantao Zhang, and Xihong Wu. 2014a. Improved parsing with taxonomy of conjunctions. In *2014 IEEE China Summit & International Conference on Signal and Information Processing*. IEEE.

Dongchen Li, Xiantao Zhang, and Xihong Wu. 2014b. Learning grammar with explicit annotations for subordinating conjunctions in chinese. In *Proceedings of the 52th annual meeting of the Association for Computational Linguistics Student Research Workshop*. Association for Computational Linguistics.

Xiaojun Lin, Yang Fan, Meng Zhang, Xihong Wu, and Huisheng Chi. 2009. Refining grammars for parsing with hierarchical semantic knowledge. In *Proceedings of the 2009 conference on empirical methods in natural language processing: Volume 3-Volume 3*, pages 1298–1307. Association for Computational Linguistics.

David Mareček and Zdeněk Žabokrtskỳ. 2014. Dealing with function words in unsupervised dependency parsing. In *Computational Linguistics and Intelligent Text Processing*, pages 250–261. Springer.

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic cfg with latent annotations. In *Proceedings of the 43rd annual meeting on Association for Computational Linguistics*, pages 75–82. Association for Computational Linguistics.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human language technologies 2007: the conference of the North American chapter of the Association for Computational Linguistics*, pages 404–411.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st international conference on computational linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics.

Slav Orlinov Petrov. 2009. *Coarse-to-Fine natural language processing*. Ph.D. thesis, University of California.

Detlef Prescher. 2005. Inducing head-driven pcfgs with latent heads: Refining a tree-bank grammar for parsing. In *Machine Learning: ECML 2005*, pages 292–304. Springer.

Hui Wang and Shiwen Yu. 2003. The semantic knowledge-base of contemporary chinese and its applications in wsd. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 112–118. Association for Computational Linguistics.

Fei Xia. 2000. The part-of-speech tagging guidelines for the penn chinese treebank (3.0). *Technical report*.

Deyi Xiong, Shuanglong Li, Qun Liu, Shouxun Lin, and Yueliang Qian. 2005. Parsing the penn chinese treebank with semantic knowledge. In *Natural language processing–IJCNLP 2005*, pages 70–81. Springer.

Nianwen Xue, Fei Xia, Shizhe Huang, and Anthony Kroch. 2000. The bracketing guidelines for the penn chinese treebank (3.0). *Technical report*.

Nianwen Xue, Fu-Dong Chiou, and Martha Palmer. 2002. Building a large-scale annotated chinese corpus. In *Proceedings of the 19th international conference on computational linguistics-Volume 1*, pages 1–8. Association for Computational Linguistics.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational linguistics*, 37(1):105–151.

Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2013. Chinese parsing exploiting characters. *51st annual meeting of the Association for Computational Linguistics*.

Qiang Zhou. 2004. Annotation scheme for chinese treebank. *Journal of Chinese information processing*, 18(4):1–8.

Qiang Zhou. 2012. Evaluation report of the third chinese parsing evaluation: Cips-sighan-parseval-2012. In *Proceedings of the second CIPS-SIGHAN joint conference on Chinese language processing*, pages 159–167.

Xuefeng Zhu, Shiwen Yu, and Hui Wang. 1995. The development of contemporary chinese grammatical knowledge base and its applications. *International journal of asian language processing*, 5(1,2):39–41.

A Neural Reordering Model for Phrase-based Translation

Peng Li[†] **Yang Liu**[†] **Maosong Sun**[†] **Tatsuya Izuha**[‡] **Dakun Zhang**[*]

[†]State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Sci. and Tech., Tsinghua University, Beijing, China
pengli09@gmail.com, {liuyang2011,sms}@tsinghua.edu.cn

[‡]Toshiba Corporation Corporate Research & Development Center
tatsuya.izuha@toshiba.co.jp

[*]Toshiba (China) R&D Center
zhangdakun@toshiba.com.cn

Abstract

While lexicalized reordering models have been widely used in phrase-based translation systems, they suffer from three drawbacks: context insensitivity, ambiguity, and sparsity. We propose a neural reordering model that conditions reordering probabilities on the words of both the current and previous phrase pairs. Including the words of previous phrase pairs significantly improves context sensitivity and reduces reordering ambiguity. To alleviate the data sparsity problem, we build one classifier for all phrase pairs, which are represented as continuous space vectors. Experiments on the NIST Chinese-English datasets show that our neural reordering model achieves significant improvements over state-of-the-art lexicalized reordering models.

1 Introduction

Reordering plays a crucial role in phrase-based translation (Koehn et al., 2003; Och and Ney, 2004). While local reordering can be directly memorized in phrases, modeling reordering at a phrase level still remains a major challenge: it can be cast as a travelling salesman problem and proves to be NP-complete (Knight, 1999; Zaslavskiy et al., 2009).

The past decade has witnessed the rapid development of phrase reordering models (e.g., (Och et al., 2004; Tillman, 2004; Zens et al., 2004; Xiong et al., 2006; Al-Onaizan and Papineni, 2006; Koehn et al., 2007; Galley and Manning, 2008; Feng et al., 2010; Green et al., 2010; Bisazza and Federico, 2012; Cherry, 2013), just to name a few). Among them, *lexicalized reordering models* (Tillman, 2004; Koehn et al., 2007; Galley and Manning, 2008) have been widely used in practical phrase-based systems. Unlike the distance-based reordering model (Koehn et al., 2003) that only penalizes phrase displacements in terms of the degree of nonmonotonicity, lexicalized reordering models introduce reordering probabilities conditioned on the words of each phrase pair. They often distinguish between three orientations with respect to the previous phrase pair: *monotone*, *swap*, and *discontinuous*. As lexicalized reordering models capture the phenomenon that some words are far more likely to be displaced than others, they outperform unlexicalized reordering models substantially.

Despite their apparent success in statistical machine translation, lexicalized reordering models suffer from the following three drawbacks:

1. *Context insensitivity*. Lexicalized reordering models determine the orientations only depending on the words of current phrase pairs. In fact, a phrase pair usually has different orientations in different contexts. It is important to include more contexts to improve the expressive power of reordering models.

2. *Ambiguity*. Short phrase pairs, which are observed in the training data more frequently, usually have multiple orientations. We observe that about 92.4% of one-word Chinese-English phrase pairs are ambiguous. This makes it hard to decide which orientation should be properly used in decoding.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1897–1907, Dublin, Ireland, August 23-29 2014.

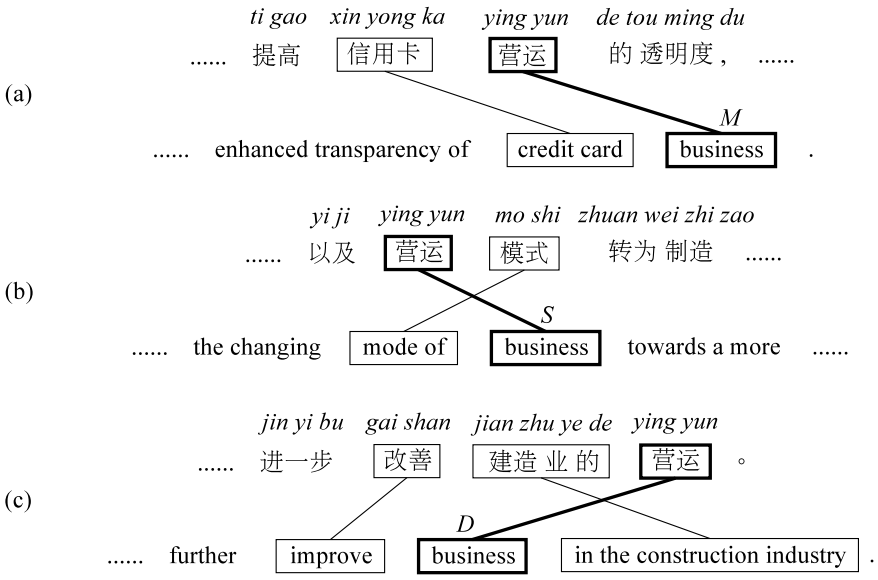


Figure 1: Ambiguity in phrase reordering. The phrase pair ⟨"*yingyun*", "business"⟩ is labeled with different orientations in different contexts: (a) *monotone*, (b) *swap*, and (c) *discontinuous*. Lexicalized reordering models use fixed probability distributions (e.g., 17.50% for M, 1.59% for S, and 80.92% for D) in decoding even though the surrounding contexts keep changing.

3. *Sparsity*. Lexicalized reordering models maintain a reordering probability distribution for each phrase pair. As most long phrase pairs that are capable of memorizing local word selection and reordering only occur once in the training data, maximum likelihood estimation can hardly train the models accurately.

In this work, we propose a neural reordering model for phrase-based translation. The contribution is twofold. Firstly, unlike conventional lexicalized reordering models, the neural reordering model conditions reordering probabilities on the words of both the current and previous phrase pairs. Including the words of previous phrase pairs significantly improves context sensitivity and reduces reordering ambiguity. Secondly, to alleviate the data sparsity problem, we build a neural classifier for all phrase pairs, which are represented as continuous space vectors. Experiments on the NIST Chinese-English datasets show that our neural reordering model achieves significant improvements over state-of-the-art lexicalized models.

2 Lexicalized Reordering Models

The lexicalized reordering models (Tillman, 2004; Koehn et al., 2007; Galley and Manning, 2008) have become the *de facto* standard in modern phrase-based systems. These models are called *lexicalized* because they condition reordering probabilities on the words of each phrase pair. Depending on the relationship between the current and previous phrase pairs, lexicalized reordering models often define *orientations* to classify different reordering patterns.

More formally, we use $\mathbf{f} = \{\tilde{f}_1, \ldots, \tilde{f}_n\}$ to denote a sequence of source phrases, $\mathbf{e} = \{\tilde{e}_1, \ldots, \tilde{e}_n\}$ to denote the phrase sequence on the target side, and $\mathbf{a} = \{a_1, \ldots, a_n\}$ to denote the alignment between source and target phrases. A source phrase $\tilde{f}_{a_i}$ and a target phrase $\tilde{e}_i$ form a phrase pair. Lexicalized reordering models aim to estimate the conditional probability of a sequence of orientations $\mathbf{o} = \{o_1, \ldots, o_n\}$:

$$P(\mathbf{o}|\mathbf{f}, \mathbf{e}, \mathbf{a}) = \prod_{i=1}^{n} P(o_i|\mathbf{f}, \tilde{e}_1, \ldots, \tilde{e}_i, a_1, \ldots, a_i) \quad (1)$$

where each o_i takes values over a set of predefined orientations. For simplicity, current lexicalized

model	source phrase length						
	1	2	3	4	5	6	7
$P(o_i\|\tilde{f}_{a_i}, \tilde{e}_i, a_{i-1}, a_i)$	92.74	54.01	24.09	14.40	10.78	8.47	6.95
$P(o_i\|\tilde{f}_{a_i}, \tilde{e}_i, \tilde{f}_{a_{i-1}}, \tilde{e}_{a_{i-1}}, a_{i-1}, a_i)$	21.72	5.22	2.63	1.48	0.98	0.67	0.54

Table 1: Percentages of phrase pairs that have multiple orientations. Including previous phrase pairs in modeling significantly reduces the reordering ambiguity for the M/S/D orientations. For example, while 92.74% of 1-word Chinese-English phrase pairs have multiple orientations observed in the training data, the ratio dramatically drops to 21.72% if the orientations are conditioned on both the current and previous phrase pairs.

reordering models use orientations conditioned only on a_{i-1} and a_i:

$$P(\mathbf{o}|\mathbf{f}, \mathbf{e}, \mathbf{a}) \approx \prod_{i=1}^{n} P(o_i|\tilde{f}_{a_i}, \tilde{e}_i, a_{i-1}, a_i) \tag{2}$$

The most widely used orientations are *monotone* (M), *swap* (S), and *discontinuous* (D): [1]

$$o_i = \begin{cases} M & \text{if } a_i - a_{i-1} = 1 \\ S & \text{if } a_i - a_{i-1} = -1 \\ D & \text{if } |a_i - a_{i-1}| \neq 1 \end{cases} \tag{3}$$

As lexicalized reordering models maintain a reordering probability distribution for each phrase pair, it is hard to accurately learn reordering probabilities for long phrase pairs that are usually observed only once in the training data. On the contrary, short phrase pairs that occur in the training data for many times tend to be ambiguous. For example, as shown in Figure 1, a Chinese-English phrase pair ⟨"*yingyun*", "business"⟩ is observed to have different orientations in different contexts.

It is unreasonable to use fixed reordering probability distributions in decoding as the surrounding contexts keep changing. Previous study shows that considering more contexts into reordering modeling improves translation performance (Khalilov and Simaan, 2010). Therefore, we need a more powerful mechanism to include more contexts, resolve the reordering ambiguity, and reduce the data sparsity.

3 A Neural Reordering Model

3.1 The Model

Intuitively, conditioning reordering probabilities on the words of both the current and previous phrase pairs will significantly reduce both reordering ambiguity and context insensitivity. The new reordering model is given by

$$P(\mathbf{o}|\mathbf{f}, \mathbf{e}, \mathbf{a}) \approx \prod_{i=1}^{n} P(o_i|\tilde{f}_{a_i}, \tilde{e}_i, \tilde{f}_{a_{i-1}}, \tilde{e}_{i-1}, a_{i-1}, a_i) \tag{4}$$

where $\langle \tilde{f}_{a_{i-1}}, \tilde{e}_{i-1} \rangle$ is the previous phrase pair.

Including the previous phrase pairs improves the context sensitivity. For example, given a phrase pair ⟨"*yingyun*", "business"⟩, its orientation is more likely to be *monotone* if it is preceded by a noun phrase pair such as ⟨"*xinyongka*", "credit card"⟩. On the contrary, the probability of the *discontinuous* orientation is higher if the previous phrase pairs contain verbs such as ⟨"*gaishan*", "improve"⟩. Therefore, the new model is capable of capturing the phenomenon that the orientation of a phrase pair depends on its surrounding contexts.

Another advantage of including previous phrase pairs is the reduction of reordering ambiguity. As shown in Table 1, 92.74% of 1-word Chinese-English phrase pairs have multiple orientations (i.e., M, S,

[1] There are many variants of lexicalized reordering models depending on the model type, orientation, directionality, language, and collapsing. See http://www.statmt.org/moses/?n=FactoredTraining.BuildReorderingModel for more details.

and D) observed in the training data. The ratio decreases with the increase of phrase length. In contrast, the new model is much less ambiguous (e.g., the ratio of ambiguous one-word phrase pairs dramatically drops to 21.72%) as it is conditioned on both the current and previous phrase pairs.

Unfortunately, including more contexts in modeling also increases the data sparsity. We observe that about 90% of reordering examples (i.e., the current and previous phrase pairs) are observed only once in the training data. As a result, it is more difficult to train lexicalized reordering models accurately using maximum likelihood estimation.

To alleviate the data sparsity problem, we use the following two strategies:

1. *Reordering as classification*. Instead of maintaining a reordering probability distribution for each phrase pair, we build a reordering classifier for all phrase pairs (Xiong et al., 2006; Li et al., 2013). This significantly reduces data sparsity by considering all occurrences of extracted phrase pairs as training examples. We find that $500,000$ reordering examples suffice to train a robust classifier (Section 4.5).

2. *Continuous space representation*. Instead of using a symbolic representation of phrases, we use a continuous space representation that treats a phrase as a dense real-valued vector (Socher et al., 2011b; Li et al., 2013). Consider two phrases "in London" and "in Centara Grand". It is usually easy to predict the orientations of "in London" because it might be observed in the training data for many times. This is not the case for "in Centara Grand" as it might occur only once. However, if the two phrases happen to have very similar continuous space representations, "in Centara Grand" is likely to have a similar reordering probability distribution with "in London".

To generate vector space representation for phrases, we follow Socher et al. (2011a) to use recursive autoencoders. Given two words w_1 and w_2, suppose their vector space representations are c_1 and c_2. The vector space representation p of the two-word phrase $\{w_1, w_2\}$ can be computed using a two-layer neural network:

$$p = g^{(1)}(W^{(1)}[c_1; c_2] + b^{(1)}) \tag{5}$$

where $[c_1; c_2] \in \mathbb{R}^{2n}$ is the concatenation of c_1 and c_2, $W^{(1)} \in \mathbb{R}^{n \times 2n}$ is a weight matrix, $b^{(1)}$ is a bias vector, and $g^{(1)}$ is an element-wise activation function.

In order to measure how well p represents c_1 and c_2, they can be reconstructed using another two-layer neural network:

$$[c_1'; c_2'] = g^{(2)}(W^{(2)}p + b^{(2)}) \tag{6}$$

where $c_1' \in \mathbb{R}^n$ and $c_2' \in \mathbb{R}^n$ are reconstructed vectors of c_1 and c_2, $W^{(2)} \in \mathbb{R}^{2n \times n}$ is a weight matrix, $b^{(2)} \in \mathbb{R}^n$ is a bias vector, and $g^{(2)}$ is an element-wise activation function. The reconstruction error can be measured by comparing c_1 and c_2 with c_1' and c_2'. This process runs recursively in a bottom-up style to obtain the vector space representation of a multi-word phrase (Socher et al., 2011a). Socher et al. (2011a) find that minimizing the norms of hidden layers leads to the reduction of reconstruction error in an undesirable way. Therefore, we normalize p such that $||p||_2 = 1$.

Treating phrase reordering as a classification problem, we propose a neural reordering classifier that takes the current and previous phrase pairs as input. The neural network consists of four recursive autoencoders and a softmax layer. The input of the classifier are the previous phrase pair and the current phrase pair. Four recursive autoencoders are used to transform the four phrases (i.e., $\tilde{f}_{a_i}, \tilde{e}_i, \tilde{f}_{a_{i-1}}, \tilde{e}_{i-1}$) into vectors. Then, these vectors are fed to the softmax layer to predict reordering orientations. Note that the recursive autoencoders for the same language share with the same parameters. Our neural network is similar to that of Li et al. (2013). The major difference is that Li et al. (2013) need to compute vector space representation for variable-sized blocks ranging from words to sentences on the fly both in training and decoding. In contrast, we only need to compute vectors for phrases with up to 7 words in the training phase, which makes our approach simpler and more scalable to large data.

Formally, given the previous phrase pair $\langle \tilde{f}_{a_{i-1}}, \tilde{e}_{i-1} \rangle$, the current phrase pair $\langle \tilde{f}_i, \tilde{e}_i \rangle$ and the orientation o_i, the reordering probability is computed as

$$P(o_i | \tilde{f}_{a_i}, \tilde{e}_i, \tilde{f}_{a_{i-1}}, \tilde{e}_{i-1}, a_{i-1}, a_i) = g(W^o c(\tilde{f}_{a_i}, \tilde{e}_i, \tilde{f}_{a_{i-1}}, \tilde{e}_{i-1}) + b^o), \quad (7)$$

where W^o is a weight matrix, b^o is a bias vector, $c(\tilde{f}_{a_i}, \tilde{e}_i, \tilde{f}_{a_{i-1}}, \tilde{e}_{i-1})$ is the concatenation of the vectors of the four phrases. [2]

Following Och (2003), we use a linear model in our decoder with conventional features (e.g., translation probabilities and n-gram language model). The neural reordering model is incorporated into the discriminative framework as an additional feature.

3.2 Training

Training the neural reordering model involves minimizing the following two kinds of errors:

- *Reconstruction error*: It measures how well the computed vector space representations represent the input vectors. It is defined as the average reconstruction error of all the parent nodes in the trees formed during computing the vector space representation for all the phrases in the training data.
- *Classification error*: It measures how well the resulting classifier predicts the reordering orientations. It is defined as the average cross-entropy errors of all the training examples.

In our experiments, the objective function is a linear interpolation of the reconstruction error and the classification error.

Following Socher et al. (2011b), we use L-BFGS (Liu and Nocedal, 1989) to optimize the parameters. At the beginning of each iteration, a binary tree for each phrase is constructed using a greedy algorithm (Socher et al., 2011b). [3] With these trees fixed, the partial derivatives with respect to parameters are computed via the backpropagation through structures algorithm (Goller and Kuchler, 1996).

When optimizing the parameters of the softmax layer, the training procedure keeps the parameters of the recursive autoencoders and word embedding matrices fixed. The corresponding error function is the classification error as described above. We also use L-BFGS to optimize the parameters and the standard error backpropagation algorithm (Rumelhart et al., 1986) to compute the derivatives.

3.3 Decoding

As the vector space representation of a phrase is calculated based on all the words in the phrase, using the neural reordering model complicates the conditions for risk-free hypothesis recombination (Koehn et al., 2003). Therefore, many hypotheses are not likely to be recombined if the neural reordering model is directly integrated in decoding, making the decoder to only explore in a much smaller search space. [4] Therefore, we use Moses to generate search graphs and then use *hypergraph reranking* (Huang and Chiang, 2007; Huang, 2008) to find most probable derivations using the neural reordering model.

4 Experiments

4.1 Data Preparation

We evaluate our reordering model on Chinese-English translation. The training corpus consists of 1.23M sentence pairs with 32.1M Chinese words and 35.4M English words. A 4-gram language model was trained on the Xinhua portion of the English GIGAWORD corpus using KenLM (Heafield, 2011), which contains 398.6M words. We used the NIST 2006 MT Chinese-English dataset as the development set, and NIST 2002-2005, 2008 MT Chinese-English datasets as the test sets. Case-insensitive BLEU is used

[2] In practice, as suggested by Socher et al. (2011b), we feed the four average vectors of the vectors present in each recursive autoencoders to the softmax layer. Taking "resident population" as an example, there are three vectors in the binary tree used by the corresponding recursive autoencoder, denoted as $\hat{x}_1$, $\hat{x}_2$ and $\hat{x}_3$. The average vector is computed as $\bar{x} = \frac{1}{3}\sum_{i=1}^{3} \hat{x}_i$.

[3] As phrases in phrase-based translation are not necessarily syntactic constituents, we do not use parse trees in this work.

[4] Experimental results show that we can only achieve comparable performance with Moses by integrating neural reordering model directly in decoding.

Model	Orientation	MT06	MT02	MT03	MT04	MT05	MT08
distance	N/A	29.56	31.40	31.27	31.34	29.98	23.87
word	M/S/D	30.19	32.03	31.86	*32.09*	30.55	24.20
	left/right	30.17	31.98	31.52	31.98	30.19	24.30
phrase	M/S/D	30.24	32.35	31.85	32.00	*30.78*	24.33
	left/right	29.57	*32.64*	31.53	31.90	30.70	24.28
hierarchical	M/S/D	*30.46*	32.52	*31.89*	*32.09*	30.39	24.11
	left/right	30.03	32.13	31.59	31.91	30.21	*24.41*
neural	M/S/D	30.68	32.19	31.94	32.20	30.81	24.71
	left/right	**31.03****	**33.03****	**32.48****	**32.52****	**31.11***	**25.20****

Table 2: Comparison of distance-based, lexicalized, and neural reordering models in terms of case-insensitive BLEU-4 scores. "distance" denotes the distance-based reordering model (Koehn et al., 2003), "word" denotes the word-based lexicalized model (Tillman, 2004), "phrase" denotes the phrase-based lexicalized model (Koehn et al., 2007), "hierarchical" denotes the hierarchical phrase-based reordering model (Galley and Manning, 2008), and "neural" denotes our model. The "left" and "right" orientations only considers whether the current source phrase is on the left of the previous source phrase or not. We use "*" to highlight the result that is significantly better than the best baseline (highlighted in italic) at $p < 0.05$ level and "**" at $p < 0.01$ level. The neural model does not work well for the M/S/D orientations due to the non-separability problem (Section 4.3).

as the evaluation metric. As a trade-off between expressive power and computational cost, we set the dimension of the word embedding vectors to 25. [5] Both $g^{(1)}$ and $g^{(2)}$ are set to $\tanh(\cdot)$. The other hyperparameters are optimized via random search (Bergstra and Bengio, 2012).

4.2 Comparison of Distance-based, Lexicalized, and Neural Reordering Models

We compare three kinds of reordering models with increasing expressive power:

1. *distance-based model*: penalizing phrase displacements proportionally to the amount of nonmonotonicity (Koehn et al., 2003);

2. *lexicalized models*: conditioning the reordering probabilities on the current phrase pairs. The orientations can be determined with respect to words (Tillman, 2004), phrases (Koehn et al., 2007), or hierarchical phrases (Galley and Manning, 2008);

3. *neural model*: conditioning the reordering probabilities on both the current and previous phrase pairs.

For lexicalized and neural models, we further distinguish between two kinds of orientation sets: {*monotone, swap, discontinuous*} and {*left, right*}. The *left*/*right* orientations only consider whether the current source phrase is on the left of the previous source phrase or not. Therefore, *swap* and *discontinuous-left* are merged into *left* while *monotone* and *discontinuous-right* into *right*.

All these reordering models are tested using Moses (Koehn et al., 2007), except that the neural model needs an additional hypergraph reranking procedure (Section 3.3). Implemented using Java, it takes the reranker 0.748 second to rerank a hypergraph on average.

Table 2 shows the case-insensitive BLEU-scores of distance-based, lexicalized, and neural reordering models on the NIST Chinese-English datasets. "distance" denotes the distance-based reordering model (Koehn et al., 2003), "word" denotes the word-based lexicalized model (Tillman, 2004), "phrase" denotes the phrase-based lexicalized model (Koehn et al., 2007), "hierarchical" denotes the hierarchical phrase-based reordering model (Galley and Manning, 2008), and "neural" denotes our model.

[5] We find that the dimensions of vectors do not have a significant impact on translation performance. For efficiency, we set the dimension to 25.

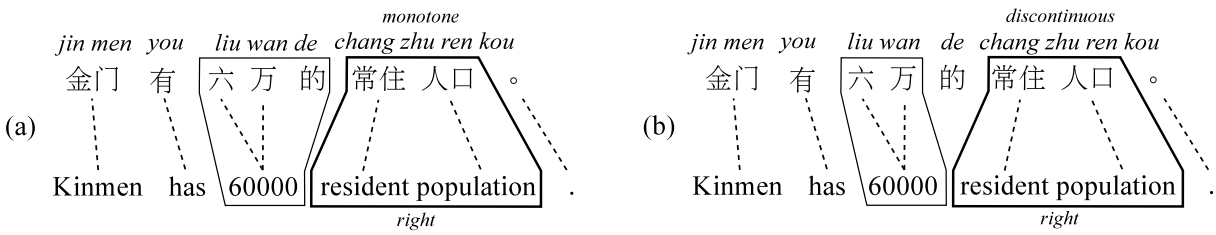


Figure 2: The non-separability problem for the neural reordering model. Given an aligned Chinese-English sentence pair, the unaligned Chinese word "*de*" makes a big difference in determining M/S/D orientations. In (a), "*de*" is included in the previous source phrase and thus the orientation is *monotone*. In (b), however, it is not included in the previous source phrase and the orientation is *discontinuous*. In our neural reordering model, "*liu wan de*" and "*liu wan*" have very similar vector space representations yet different orientations (i.e., M and D). In other words, training examples labeled with M, S, D are prone to be mixed with each other in the vector space. Therefore, it is difficult to find a hyperplane to separate M, S and D examples in the high-dimensional space.

We find that lexicalized reordering models obtain significant improvements over the distance-based model, which indicates that conditioning reordering probabilities on the words of the current phrase pairs does improve the expressive power. Our neural model using *left*/*right* orientations significantly outperforms all variants of lexicalized models. We use "*" to highlight the result that is significantly better than the best baseline (highlighted in italic) at $p < 0.05$ level and "**" at $p < 0.01$ level. This suggests that conditioning reordering probabilities on the words of current and previous phrase pairs is helpful for resolving reordering ambiguities and reducing context insensitivity.

4.3 The Non-Separability Problem

In Table 2, the neural model using the M/S/D orientations fails to outperform lexicalized models significantly. One possible reason is that the neural model suffers from the *non-separability problem* due to the M/S/D orientations.

As shown in Figure 2, given an aligned Chinese-English sentence pair, the unaligned Chinese function word "*de*" makes a big difference in determining M/S/D orientations. In (a), "*de*" is included in the previous source phrase and thus the orientation is *monotone*. In (b), however, "*de*" is not included in the previous source phrase and the orientation is *discontinuous*. In our neural reordering model, "*liu wan de*" and "*liu wan*" have very similar vector space representations yet different orientations (i.e., M and D). In other words, training examples labeled with M, S, D are prone to be mixed with each other in the vector space. Therefore, it is difficult to find a hyperplane to separate M, S and D examples in the high-dimensional space.

Fortunately, we find that using the *left*/*right* orientations can alleviate this problem. As the *left*/*right* orientations only consider whether the current source phrase is on the left of the previous source phrase or not, unaligned source words will not change orientations. For example, both Figure 2(a) and 2(b) are identified as the *right* orientation.

As a result, using *left*/*right* orientations in the neural reordering model not only has a higher classification accuracy (85%) over using the M/S/D orientations (69%), but also achieves higher BLEU scores on all NIST datasets systematically.

4.4 The Effect of Distortion Limit

Figure 3 shows the performance of the lexicalized model and our neural model with various distortion limits. The lexicalized model is the word-based model with M/S/D orientations. The neural model uses *left*/*right* orientations. The neural model consistently outperforms the lexicalized model, especially for large distortion limits. This finding suggests that the neural model is superior to lexicalized models in predicting long-distance reordering.

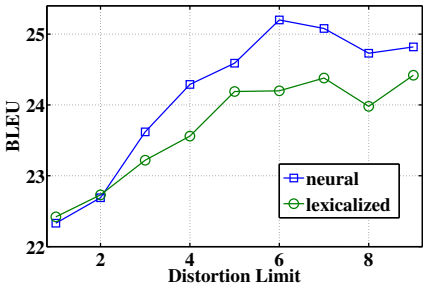


Figure 3: BLEU with various distortion limits.

# examples	Accuracy	BLEU
100,000	83.55	30.92
200,000	84.40	31.03
300,000	84.55	31.01
400,000	84.95	30.93
500,000	85.25	31.27
3,000,000	85.55	31.03

Table 3: Effect of training corpus size.

Vectors	MT06	MT02	MT03	MT04	MT05	MT08
ours	31.03	33.03	32.48	32.52	31.11	25.20
word2vec	30.44	32.28	32.00	32.07	30.24	24.54

Table 4: Comparison of neural reordering models trained based on word vectors produced by our model (ours) and word2vec (Mikolov et al., 2013).

4.5 The Effect of Training Corpus Size

Table 3 shows the classification accuracy and translation performance with various number of randomly sampled reordering examples for training the neural classifier. The classification accuracy and translation performance generally rise as the number of reordering example increases.[6] Surprisingly, both the classification accuracy and translation performance of using 500,000 reordering examples are close to using 3,000,000 reordering examples, suggesting that a relatively small amount of reordering examples are enough for training a robust classifier.

4.6 Learned Vector Space Representations

We randomly sampled 200,000 English phrases and found 999 clusters according to the vector space representations computed by recursive autoencoders using the k-means algorithm (MacQueen, 1967). The distance between two phrases is calculated by the Euclidean distance between their vector space representations.

Figure 4 shows 10 of the 999 clusters. An interesting finding is that phrase pairs that are close in the vector space share with similar reordering patterns rather than semantic similarity. For example, "by june 1" and "within the agencies" have similar distributions on the *left*/*right* orientations but are totally unrelated in terms of meaning. As a result, the vector representations of words trained using unlabeled data hardly helps in training the neural reordering model. Table 4 shows the results when we replace the word vectors of our model with those trained using word2vec (Mikolov et al., 2013). The recursive autoencoders and the classifier are retrained. The performance of the neural reordering model trained in this way drops significantly, which confirms our analysis.

5 Related Work

Reordering as classification is a common way to alleviate the data sparsity problem. Xiong et al. (2006) use a maximum entropy model to predict whether to merge two blocks in a straight or an inverted order in their ITG decoder. Nguyen et al. (2009) build a similar model for hierarchical phrase reordering models (Galley and Manning, 2008). Green et al. (2010) and Yahyaei and Monz (2010) predict finer-grained distance bins instead. Another direction is to learn sparse reordering features and create more flexible distributions (Cherry, 2013). Although these models are effective, feature engineering is a major challenge. In contrast, our neural reordering model is capable of learning features automatically.

[6]The reason why the BLEU scores oscillate slightly on the training set is that classification accuracy is not directly correlated with BLEU scores. Optimizing the neural reordering model directly with respect to BLEU score may further improve the performance. We leave this for future work.

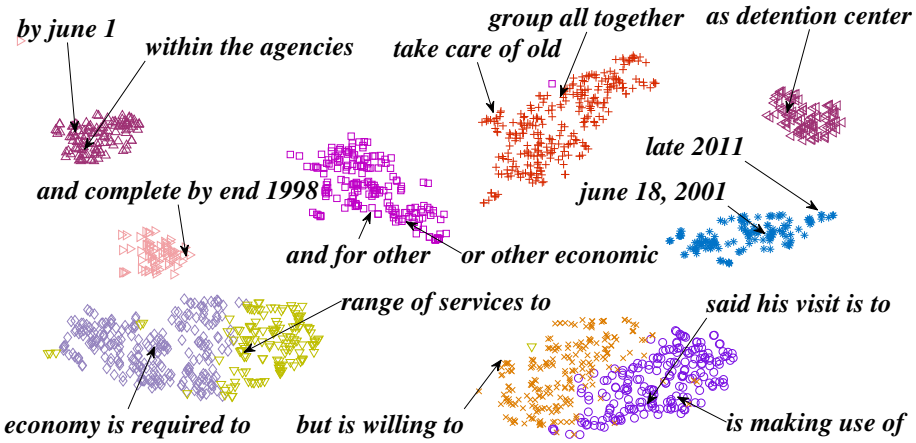


Figure 4: Phrase clusters as calculated by the Euclidean distance in the vector space. English phrases that have similar reordering probability distributions rather than similar semantic similarity fall into one cluster.

Along another line, n-gram-based models (Mariño et al., 2006; Durrani et al., 2011; Durrani et al., 2013) treat translation as Markov chains over minimal translation units (Mariño et al., 2006; Durrani et al., 2013) or operations (Durrani et al., 2011) directly. Although naturally leveraging both the source and target side contexts, these approaches still face the data sparsity problem.

Our work is closely related to Li et al. (2013). The major difference is that Li et al. (2013) need to compute vector space representation for variable-sized blocks ranging from words to sentences on the fly both in training and decoding. In contrast, we only need to compute vectors for phrases with up to 7 words in the training phase, which makes our approach simpler and more scalable to large data.

6 Conclusion

We have shown that surrounding context is effective for resolving reordering ambiguities in phrase-based models. As the data sparseness problem is the major challenge for using context in reordering models, we propose to use a single classifier based on recursive autoencoders to predict reordering orientations. Experimental results show that our neural reordering model outperforms the state-of-the-art lexicalized reordering models significantly and consistently across all the NIST datasets under various settings.

There are a few future directions we plan to explore. First, as the machine translation system and neural classifier are trained separately, the neural network training only has an indirect effect on translation quality. Jointly training the machine translation system and neural classifier is an interesting topic. Second, it is interesting to develop more efficient models to leverage larger contexts to resolve reordering ambiguities. Third, we plan to extend our work to other translation models such as syntax-based and n-gram based models (Mariño et al., 2006; Durrani et al., 2011; Durrani et al., 2013). Finally, as we cast phrase reordering as two-category classification problem (i.e, *left* vs. *right*), it is interesting to intersect structured SVM (Tsochantaridis et al., 2005) with neural networks to develop a large margin training algorithm for our neural reordering model.

Acknowledgements

This research is supported by the 973 Program (No. 2014CB340501), the National Natural Science Foundation of China (No. 61331013), the 863 Program (No. 2012AA011102), Toshiba Corporation Corporate Research & Development Center, and the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme.

References

Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 529–536.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(1):281–305.

Arianna Bisazza and Marcello Federico. 2012. Modified distortion matrices for phrase-based statistical machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 478–487.

Colin Cherry. 2013. Improved reordering for phrase-based translation using sparse features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 22–31.

Nadir Durrani, Helmut Schmid, and Alexander Fraser. 2011. A joint sequence translation model with integrated reordering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1045–1054.

Nadir Durrani, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013. Can Markov models over minimal translation units help phrase-based SMT? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 399–405.

Yang Feng, Haitao Mi, Yang Liu, and Qun Liu. 2010. An efficient shift-reduce decoding algorithm for phrased-based machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 285–293.

Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856.

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of 1996 IEEE International Conference on Neural Networks (Volume:1)*, volume 1, pages 347–352.

Spence Green, Michel Galley, and Christopher D. Manning. 2010. Improved models of distortion cost for statistical machine translation. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 867–875.

Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, pages 187–197.

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 586–594.

Maxim Khalilov and Khalil Simaan. 2010. Source reordering using maxent classifiers and supertags. In *Proceedings of The 14th Annual Conference of the European Association for Machine Translation*, pages 292–299.

Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180.

Peng Li, Yang Liu, and Maosong Sun. 2013. Recursive autoencoders for ITG-based translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 567–577.

DongC. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528.

James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297.

José B. Mariño, Rafael E. Banchs, Josep M. Crego, Adrià de Gispert, Patrik Lambert, José A. R. Fonollosa, and Marta R. Costa-jussà. 2006. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Vinh Van Nguyen, Akira Shimazu, Minh Le Nguyen, and Thai Phuong Nguyen. 2009. Improving a lexicalized hierarchical reordering model using maximum entropy. In *Proceedings of The twelfth Machine Translation Summit (MT Summit XII)*.

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

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004: Main Proceedings*, pages 161–168.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning internal representations by error propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362.

Richard Socher, Eric H. Huang, Jeffrey Pennin, Andrew Y. Ng, and Christopher D. Manning. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proceedings of Advances in Neural Information Processing Systems 24*, pages 801–809.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 151–161.

Christoph Tillman. 2004. A unigram orientation model for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004: Short Papers*, pages 101–104.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528.

Sirvan Yahyaei and Christof Monz. 2010. Dynamic distortion in a discriminative reordering model for statistical machine translation. In *Proceedings of the 7th International Workshop on Spoken Language Translation*, pages 353–360.

Mikhail Zaslavskiy, Marc Dymetman, and Nicola Cancedda. 2009. Phrase-based statistical machine translation as a traveling salesman problem. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 333–341.

Richard Zens, Hermann Ney, Taro Watanabe, and Eiichiro Sumita. 2004. Reordering constraints for phrase-based statistical machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 205–211.

Recurrent Neural Network-based Tuple Sequence Model for Machine Translation

Youzheng Wu, Taro Watanabe, Chiori Hori
National Institute of Information and Communications Technology (NICT), Japan
erzhengcn@gmail.com
{taro.watanabe, chiori.hori}@nict.go.jp

Abstract

In this paper, we propose a recurrent neural network-based tuple sequence model (RNNTSM) that can help phrase-based translation model overcome the phrasal independence assumption. Our RNNTSM can potentially capture arbitrary long contextual information during estimating probabilities of tuples in continuous space. It, however, has severe data sparsity problem due to the large tuple vocabulary coupled with the limited bilingual training data. To tackle this problem, we propose two improvements. The first is to factorize bilingual tuples of RNNTSM into source and target sides, we call factorized RNNTSM. The second is to decompose phrasal bilingual tuples to word bilingual tuples for providing fine-grained tuple model. Our extensive experimental results on the IWSLT2012 test sets[1] showed that the proposed approach essentially improved the translation quality over state-of-the-art phrase-based translation systems (baselines) and recurrent neural network language models (RNNLMs). Compared with the baselines, the BLEU scores on English-French and English-German tasks were greatly enhanced by 2.1%-2.6% and 1.8%-2.1%, respectively.

1 Introduction

The phrase-based translation systems (Koehn et al., 2003) rely on language model and lexicalized reordering model to capture lexical dependencies that span phrase boundaries. Their translation models, however, do not explicitly model context dependencies between translation units. To address this limitation, Marino et al. (2006) and Crego and Yvon (2010) proposed n-gram-based translation systems to capture dependencies across phrasal boundaries. The n-gram translation models have been shown to be effective in helping the phrase-based translation models overcome the phrasal independence assumption (Durrani et al., 2013; Zhang et al., 2013). Most of the n-gram translation models (Marino et al., 2006; Durrani et al., 2013; Zhang et al., 2013) employed Markov (n-gram) model over sequence of bilingual tuples also known as minimal translation units (MTUs).

Recently, some pioneer studies (Schwenk et al., 2007; Son et al., 2012) proposed feed-forward neural networks with factorizations to model bilingual tuples in a continuous space. Although the authors reported some gains over the n-gram model in machine translation tasks, these models can only capture a limited amount of context and remain a kind of n-gram model. In language modeling, experimental results in (Mikolov et al., 2011; Arisoy et al., 2012; Sundermeyer et al., 2013) showed that recurrent neural networks (RNNs) outperform feed-forward neural networks in both perplexity and word error rate in speech recognition even though it is harder to train properly.

Therefore, in this paper we take the advantages of RNN and tuple sequence model and propose recurrent neural network-based tuple sequence models (RNNTSMs) to improve phrase-based translation system. Our RNNTSMs are capable of modeling long-span context and have better generalization. Compared with such related studies as (Schwenk et al., 2006; Son et al., 2012), our main contributions can

[1]The IWSLT workshop aims at translating TED speeches (http://www.ted.com), a collection of public lectures covering a variety of topics.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1908–1917, Dublin, Ireland, August 23-29 2014.

be summarized as: (i) our models can be regarded as deep neural network translation models because they can capture arbitrary-length context potentially, which are proven to estimate more accurate probabilities of bilingual tuples; (ii) we extend the conventional RNNTSM to factorized RNNTSMs that can significantly overcome the data sparseness problem caused by the large vocabularies of bilingual tuples by incorporating the factors from the source and the target sides in addition to bilingual tuples; (iii) we investigate heuristic rules to decompose phrasal bilingual tuples to word bilingual tuples for reducing the out-of-tuple-vocabulary rate and providing fine-grained tuple sequence model; (iv) we integrate the proposed models into the state-of-the-art phrase-based translation system (MOSES) as a supplement of the work in (Son et al., 2012) that is a complete n-gram translation system.

2 Related Work

The n-gram translation model (Marino et al., 2006) is a Markov model over phrasal bilingual tuples and can improve the phrase-based translation system (Koehn et al., 2003) by providing contextual dependencies between phrase pairs. To further improve the n-gram translation model, Crego and Yvon (2010) explored factored bilingual n-gram language models. Durrani et al. (2011) proposed a joint sequence model for the translation and reordering probabilities. Zhang et al. (2013) explored multiple decomposition structures as well as dynamic bidirectional decomposition. Since neural networks advance the state of the art in the fields of image processing, acoustic modeling (Seide et al., 2011), language modeling (Bengio et al., 2003), natural language processing (Collobert et al., 2011; Socher et al., 2013), machine transliteration (Deselaers et al., 2009), etc, some prior studies have been done on neural network-based translation models (NNTMs).

One kind of the NNTMs relies on word-to-word alignment information or phrasal bilingual tuples. For example, Schwenk et al. (2007) investigated feed-forward neural networks to model bilingual tuples in continuous space. Son et al. (2012) improved this idea by decomposing tuple units, i.e., distinguishing the source and target sides of the tuple units, to address data sparsity issues. Although the authors reported some gains over the n-gram model in the BLEU scores on some tasks, these models can only capture a limited amount of context and remain a kind of n-gram model. In addition, a feed-forward neural network independent from bilingual tuples was proposed (Schwenk, 2012), which can infer meaningful translation probabilities for phrase pairs not seen in the training data.

Another kind of the NNTMs do not rely on alignment. Auli et al. (2013) and Kalchbrenner and Blunsom (2013) proposed joint language and translation model with recurrent neural networks, in which latent semantic analysis and convolutional sentence model were used to model source-side sentence. Potentially, they can exploit an unbounded history of both source and target words thanks to recurrent connections. However, they only modestly observed gains over the recurrent neural network language model. Previous studies (Wu and Wang, 2007; Yang et al., 2013) showed that the performance of word alignment (alignment error rate) is nearly 80%. That means explicit word alignment may be more reliable as a way to represent the corresponding bilingual sentences compared with an implicit compressed vector representation (Auli et al., 2013).

Our RNNTSM takes the advantages of the above NNTMs, that is, RNN enables our model to capture long-span contextual information, while tuple sequence model uses word alignment without much information loss. Furthermore, factorized RNN and word bilingual tuples are proposed to address data sparsity issue. To the best of our knowledge, few studies have been done on this aspect.

3 Tuple Sequence Model

In tuple sequence model, bilingual tuples are translation units extracted from word-to-word alignment. They are composed of source phrases and their aligned target phrases that are also known as minimal translation units (MTUs) and thus cannot be broken down any further without violating the constrains of the translation rules. This condition results in a unique segmentation of the bilingual sentence pair given its alignment. In our implementation, GIZA++ with `grow-diag-final-and` setting is used to conduct word-to-word alignments in both directions, source-to-target and target-to-source (Och and

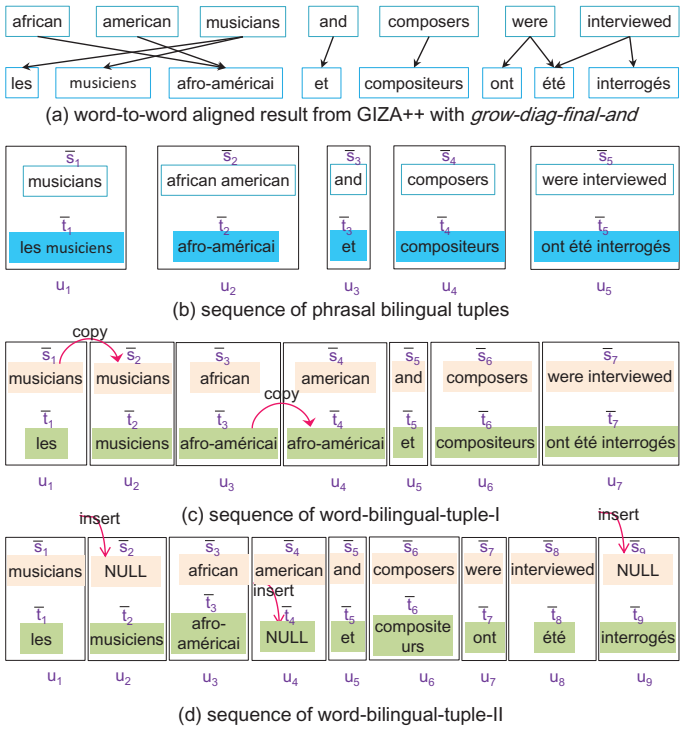


Figure 1: An example of generating basic bilingual tuples from word alignment information.

Ney, 2003). Ncode toolkit[2] is used to generate a unique bilingual segmentation of word-to-word aligned sentence. Figure 1(a)-(b) illustrates the process of generating bilingual tuple. As can be seen in Figure 1, bilingual tuple u_1 is composed of source phrase $\bar{s}_1$ (musicians) and target phrase $\bar{t}_1$ (les musiciens) linked to $\bar{s}_1$. Because this type of bilingual tuples are composed of one or more words from the source side and zero or more words from the target side, we call them phrasal bilingual tuples.

The phrasal bilingual tuple is not able to provide translations for individual words that appear tied to other words unless they occur alone in some other tuple. For example, if target phrase $\bar{t}_k$="les musiciens" is always aligned to source phrase $\bar{s}_k$="musicians" in the training corpus, then no word-to-word translation probability for "musicians:musiciens" will exist. This becomes a serious drawback when a large number of phrasal bilingual tuples are extracted from one-to-many, many-to-one, and many-to-many alignments. To tackle the issue, we propose to decompose phrasal bilingual tuples into word bilingual tuples for providing fine-grained tuple sequence model. Suppose source phrase $\bar{s}_k$, a sequence of source word $s_{k1}, s_{k2}, ..., s_{kI}$, is aligned to target phrase $\bar{t}_k$, a sequence of target word $t_{k1}, t_{k2}, ..., t_{kJ}$, in which I and J refer to the number of words in source phrase and that in target phrase. The following two types of heuristic rules are considered.

(**word-bilingual-tuple-I**): For one-to-many alignments, we copy s_{kI} $J-1$ times to fill the short phrase $\bar{s}_k$. For many-to-one alignments, we copy t_{kJ} $I-1$ times to fill the phrase $\bar{t}_k$. For many-to-many alignment, a maximum phrase length, we set it to 5, is used to avoid vocabulary explosion. That means, if $I > 5$; then $\bar{s}_k$=⟨unk⟩, if $J > 5$; then $\bar{t}_k$=⟨unk⟩.

(**word-bilingual-tuple-II**): For one-to-many, many-to-one, and many-to-many alignments, we insert a special token "NULL" $\mid J - I \mid$ times to fill the short phrase, and map each word in the extended phrase monotonically to generate a word-wise tuple sequence.

The Figure 1(c)-(d) demonstrate the decomposition results. As shown in Figure 1(c), the translation probability of "musicians" being aligned to "musiciens" can be learned in the word bilingual tuples. The word bilingual tuples enable our model use information from source-side of the tuples for computing translation probabilities of some tuples. For example, translating "musicians:musiciens" benefits from its source word "musicians". Table 2 in Section 4 shows the sizes of the tuple vocabularies. We can see

[2] http://ncode.limsi.fr/

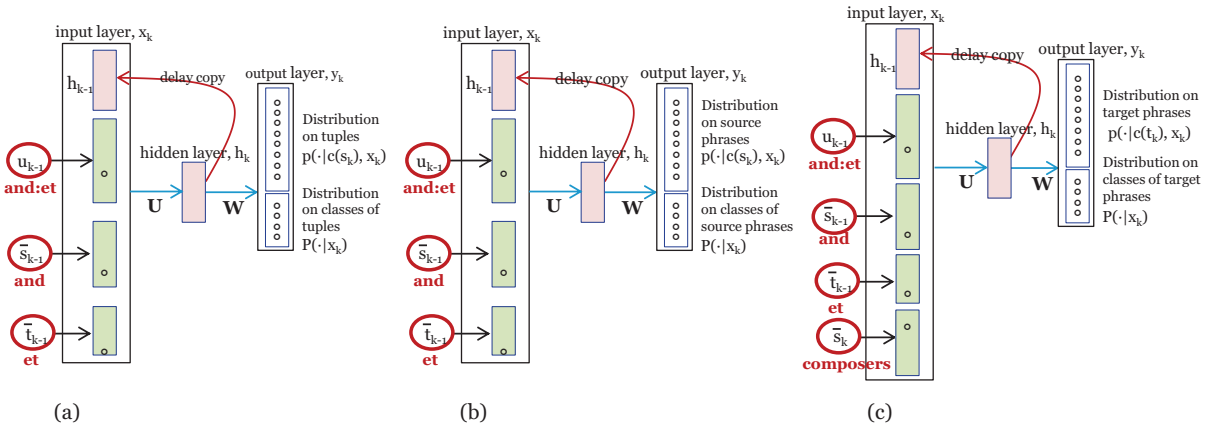


Figure 2: (a): factorized RNNTSM, called fRNNTSM for short, which will go back to the RNNTSM model when $\bar{s}_{k-1}$ and $\bar{t}_{k-1}$ are dropped. (b): fRNNTSM$_{\texttt{source}}$. (c) fRNNTSM$_{\texttt{target}}$.

that the word-bilingual-tuple-I has lower out-of-tuple-vocabulary (OOTV) rate, though it increases the tuple vocabulary. The word-bilingual-tuple-II greatly reduces the tuple vocabulary and the OOTV rate. Note that some words may not be aligned correctly, like "NULL-musiciens". However, generating these tuples can be viewed as a language model process that exploits previous source and target words, and current source word contained in previous tuples like "les-musiciens".

Thus, given a target sentence **t**, a source sentence **s**, and its alignment **a**, the tuple sequence model can be defined over the sequence of bilingual tuples $(u_1, u_2, ..., u_m)$ as follows.

$$p(\mathbf{t}, \mathbf{s}, \mathbf{a}) = \prod_{k=1}^{m} p(u_k|u_{k-1}, u_{k-2}, ..., u_1) = \prod_{k=1}^{m} p(u_k|u_{k-1}, u_{k-2}, ..., u_{k-n+1}) \quad (1)$$

where u_k denotes the k-th bilingual tuple of a given bilingual sentence pair. Each bilingual tuple u_k contains a source phrase $\bar{s}_k$ and its aligned target phrase $\bar{t}_k$[3]. Formally, u_k=$\bar{s}_k$:$\bar{t}_k$. The tuple sequence model does not make any phrasal independence assumption and generates a tuple by looking at a context of previous tuples. The n-gram translation models are Markov models over sequences of tuples, they generate a tuple by looking at previous n-1 tuples.

4 Recurrent Neural Network-based Tuple Sequence Model

In order to use long-span context, this paper presents a recurrent neural network-based tuple sequence model (RNNTSM) to approximate the probability $p(u_i|u_{i-1}, ..., u_1)$. Our RNNTSM can potentially capture arbitrary long context rather than n-1 previous tuples. The input layer encodes bilingual tuples by using 1-of-n coding, and the output layer produces a probability distribution over all bilingual tuples. The hidden layer maintains a representation of the sentence history. This RNNTSM, however, has severe data sparsity problem due to the large tuple vocabulary coupled with the limited bilingual training data.

4.1 Factorized RNNTSM

To solve the problem, we extend the RNNTSM model with factorizing tuples in input layer, as shown in Figure 2(a). Specifically, it consists of an input layer x, a hidden layer h (state layer), and an output layer y. The connection weights among layers are denoted by matrixes **U** and **W**. Unlike the RNNTSM, which predicts probability $p(u_k|u_{k-1}, h_{k-1})$, the factorized RNNTSM predicts probability $p(u_k|u_{k-1}, \bar{s}_{k-1}, \bar{t}_{k-1}, h_{k-1})$ of generating following tuple u_k and is explicitly conditioned on the preceding tuple u_{k-1}, source-side of the tuple $\bar{s}_{k-1}$, and target-side of the tuple $\bar{t}_{k-1}$. It is implicitly conditioned on the entire history by the delay copy of hidden layer h_{k-1}. For those tuples (approximately 20% as shown in Table 2) that are not contained in the training data, i.e., co-occurrence (s_{i-1}, t_{i-1})

[3]Phrases turn to words in the word bilingual tuples. For convenience, we do not distinguish them in our paper.

non-exist while either s_{i-1} or t_{i-1} exists, the factorized RNNTSM backs off to the source- (s_{i-1}) or target-side (t_{i-1}). This process resembles factored n-gram language model (Duh and Kirchhoff, 2004). However, the RNNTSM, computing $p(u_i|u_{i-1}, h_{i-1})$, cannot estimate the probabilities for those tuples. In the special case that $\bar{s}_{k-1}$ and $\bar{t}_{k-1}$ are dropped, the factorized RNNTSM goes back to the RNNTSM. For convenience, u_{k-1}, $\bar{s}_{k-1}$ and $\bar{t}_{k-1}$ are called features. In the input layer, each feature is encoded into a feature vector using the 1-of-n coding. The tuple u_{k-1}, the source phrase $\bar{s}_{k-1}$ and the target phrase $\bar{t}_{k-1}$ are encoded into $|u|$-dimension feature vector v^{u}_{k-1}, $|\bar{s}|$-dimension feature vector $v^{\bar{s}}_{k-1}$ and $|\bar{t}|$-dimension feature vector $v^{\bar{t}}_{k-1}$, respectively. Here, $|u|$, $|\bar{s}|$ and $|\bar{t}|$ stand for the sizes of the tuple, the source phrase, and the target phrase vocabularies. Finally, the input layer x_k is formed by concatenating feature vectors and hidden layer h_{k-1} at the preceding time step, as shown in the following equation.

$$x_k = [v^{u}_{k-1},\ v^{\bar{s}}_{k-1},\ v^{\bar{t}}_{k-1},\ h_{k-1}] \tag{2}$$

The neurons in the hidden and output layers are computed as follows:

$$\begin{aligned} h_k &= f(\mathbf{U} \times x_k), \ \ y_k = g(\mathbf{W} \times h_k) \\ f(z) &= \frac{1}{1+e^{-z}}, \quad g(z) = \frac{e^{z_m}}{\sum_k e^{z_k}} \end{aligned} \tag{3}$$

To speed-up both in the training and testing processes, we map bilingual tuples into classes with frequency binning and divide the output layer into two parts following (Mikolov et al., 2010). The first part estimates the posterior probability distribution over all classes. The second computes the posterior probability distribution over the tuples that belong to class $c(u_k)$, the one that contains predicted tuple u_k. Finally, translation probability $p(u_k|u_{k-1}, \bar{s}_{k-1}, \bar{t}_{k-1}, h_{k-1})$ is calculated by,

$$p(u_k|u_{k-1}, \bar{s}_{k-1}, \bar{t}_{k-1}, h_{k-1}) = p(c(u_k)|x_k) \times p(u_k|c(u_k), x_k) \tag{4}$$

4.2 Factorized RNNTSM on source and target phrases

The above factorized RNNTSM is conditioned on the previous context during computing the probability for tuple u_k. It does not exploit its source side $\bar{s}_k$. For example, tuple "composers:compositeurs" does not benefit from "composers". To address this limitation, we rewrite the probability in Equation 1.

$$\begin{aligned} p(u_k|u_{k-1}, u_{k-2}, ..., u_1) &= p(s_k, t_k|u_{k-1}, u_{k-2}, ..., u_1) \\ &= p(s_k|u_{k-1}, u_{k-2}, ..., u_1) \times p(t_k|s_k, u_{k-1}, u_{k-2}, ..., u_1) \end{aligned} \tag{5}$$

The first sub-model $p(s_k|u_{k-1}, u_{k-2}, ..., u_1)$ computes the probability distribution over source phrases. This model, called fRNNTSM$_{\texttt{source}}$ for short, can be regarded as a reordering model. The second sub-model $p(t_k|s_k, u_{k-1}, u_{k-2}, ..., u_1)$ is a translation model, abbreviated as fRNNTSM$_{\texttt{target}}$, which computes the probability distribution over $\bar{t}_k$ that are translated from $\bar{s}_k$. The two sub-models are computed with the recurrent neural networks shown in Figure 2(b)-(c). Another advantage of using the factorized RNNTSM on source and target phrases separately is that their training become faster because the vocabulary sizes of the source and target phrases are much smaller than that of the tuples.

4.3 Training

Training can be performed by the back-propagation through time (BPTT) algorithm (Boden, 2002) by minimizing an error function defined in the following equations.

$$L = \frac{1}{2} \times \sum_{i=1}^{N} (o_i - p_i)^2 + \gamma \times (\sum_{lk} u^2_{lk} + \sum_{tl} w^2_{tl}) \tag{6}$$

where N is the number of training instances, o_i denotes the desired output; i.e., the probability should be 1.0 for the predicted tuple in the training sentence and 0.0 for all others. γ is the regularization term's weight, which is determined experimentally using a validation set. The training algorithm randomly initializes the matrixes and updates them with Equation 7 over all the training instances in several iterations.

	English-French			English-German		
	tst2010	tst2011	tst2012	tst2010	tst2011	tst2012
Baseline	30.15	35.97	35.48	20.29	21.48	19.30
+RNNTSM	$30.51_{(0.3)}$	$36.11_{(0.1)}$	$36.44_{(0.9)}$	$20.67_{(0.4)}$	$21.85_{(0.4)}$	$19.56_{(0.3)}$
+fRNNTSM (1)	$31.83_{(1.6)}$	$37.58_{(1.6)}$	$37.74_{(2.2)}$	$21.67_{(1.4)}$	$22.89_{(1.4)}$	$20.60_{(1.3)}$
+fRNNTSM$_{\texttt{source}}$(2) +fRNNTSM$_{\texttt{target}}$(3)	$31.89_{(1.7)}$	$38.23_{(2.2)}$	$37.82_{(2.3)}$	$21.49_{(1.2)}$	$22.94_{(1.4)}$	$20.41_{(1.1)}$
+(1) +(2) +(3)	$32.26_{(2.1)}$	$38.36_{(2.4)}$	$38.11_{(2.6)}$	$21.80_{(1.5)}$	$22.88_{(1.4)}$	$20.76_{(1.5)}$

Table 1: BLEU scores of the RNNTSMs, the factorized RNNTSM (fRNNTSM), the fRNNTSM$_{\texttt{source}}$ (sfRNNTSM), the fRNNTSM$_{\texttt{target}}$ with the word-bilingual-tuple-I and their combination. The numbers in the parentheses are the absolute improvements over the Baseline.

In Equation 7, ψ stands for one of the connection weights in the neural networks and η is the learning rate. After each iteration, it uses validation data for stopping and controlling the learning rate. Usually, our RNNs needs 10 to 20 iterations.

$$\psi^{new} = \psi^{previous} - \eta \times \frac{\partial L}{\partial \psi} \tag{7}$$

5 Experiments

We experiment with two language pairs on the IWSLT2012 data sets (Federico et al., 2012), with English as source and French, German as target. The IWSLT data comes from TED speecheds, given by leaders in various fields and covering an open set of topics in technology, entertainment, design, and many others. In the following experiments, the IWSLT dev2010 set is used as the tuning set, the tst2010, tst2011, and tst2012 as the test sets.

Phrase-based translation systems are constructed as baselines using standard settings (GIZA++ alignment, grow-diag-final-and, lexical reordering models, SRILM, and MERT optimizer) in the MOSES toolkit (Koehn et al., 2007). The proposed models are used to re-score n-best lists produced by the baseline systems. The n-best size is set to at most 1000 for each test sentence. During the n-best re-scoring, the weights are re-tuned on the dev2010 data set with MERT optimizer[4]. The proposed RNN-based models are evaluated on a small task and a large task. For the parameters of all the RNN-based models, we set the number of hidden neurons in the hidden layer to 480 and classes in the output layer to 300.

5.1 Small Task

In the small task, the training data only contains the speech-style bi-text, i.e., the human translation of TED speeches. Specially, the corpora for the English-French and English-German pairs contain 139K and 128K parallel sentences. The language model is a standard 4-gram language model with the Kneser-Ney discounting. Both the n-gram LM and the RNNLM are trained on the target side of the bi-text corpus. As the first experiment, we compare the proposed RNNTSMs with the word-bilingual-tuple-I. Table 1 summarizes the results. The main findings from this experiment are: (1) The RNNTSM yields modest improvements of 0.3%-0.4% over the baseline system on most the test sets. (2) The factorized RNNTSMs essentially outperform the baseline and the RNNTSM on all the test sets. Specially, the improvements of the factorized RNNTSM and the combination of the fRNNTSM$_{\texttt{source}}$ and the fRNNTSM$_{\texttt{target}}$ over the baseline for the English-French task range 1.6%-2.2% and 1.7%-2.3%. For the English-German pair, these improvements are between 1.3%-1.4% and 1.1%-1.4%. The results indicate that the factorized RNNTSMs can well address the data sparsity problem of the RNNTSM. (3) The improvements for the English-German pair are comparatively smaller than that for the English-French pair. This is because German is a morphologically rich language (Fraser et al., 2013), its vocabulary is larger and the sparsity

[4]To get statistically reliable comparison (Clark et al., 2011), replication of the MERT optimizer and test set evaluation are performed five times. We finally report the average BLEU scores in the following experiments.

	English-French			English-German		
	#Tuple	#Source/#Target	OOTV	#Tuple	#Source/#Target	OOTV
Phrasal bilingual tuple	308K	130K/175K	26.0%	315K	148K/196K	23.4%
word-bilingual-tuple-I	332K	100K/111K	23.7%	351K	104K/135K	22.2%
word-bilingual-tuple-II	293K	44K/56K	14.9%	327K	43K/86K	14.8%

Table 2: Vocabulary sizes. OOTV refers to the out-of-tuple-vocabulary rate on the dev2010 set. K stands for thousands.

	English-French			English-German		
	tst2010	tst2011	tst2012	tst2010	tst2011	tst2012
+fRNNTSM$_{\mathrm{p}}$	31.44	37.68	37.34	20.76	21.80	19.57
+fRNNTSM$_{\mathrm{I}}$	$31.83_{(0.4)}$	$37.58_{(-0.1)}$	$37.74_{(0.4)}$	$21.67_{(0.9)}$	$22.89_{(1.1)}$	$20.60_{(1.0)}$
+fRNNTSM$_{\mathrm{II}}$	$31.73_{(0.3)}$	37.66	$37.78_{(0.4)}$	$22.00_{(1.2)}$	$23.24_{(1.4)}$	$21.09_{(1.5)}$
+fRNNTSM$_{\mathrm{I}}$ +fRNNTSM$_{\mathrm{II}}$	$31.98_{(0.6)}$	$37.97_{(0.3)}$	$38.14_{(0.6)}$	$22.19_{(1.4)}$	$23.25_{(1.4)}$	$21.17_{(1.7)}$

Table 3: BLEU scores of the factorized RNNTSM with various types of bilingual tuples. fRNNTSM$_{\mathrm{p}}$ refers to the fRNNTSM with phrasal bilingual tuples, fRNNTSM$_{\mathrm{I}}$ to the fRNNTSM with the word-bilingual-tuple-I, etc. + means these models are used with the baseline systems. The numbers in the parentheses are the absolute improvements over the +fRNNTSM$_{\mathrm{p}}$.

problem is more serious. (4) There is no significant difference between the factorized RNNTSM and the combination of the fRNNTSM$_{\mathtt{source}}$ and the fRNNTSM$_{\mathtt{target}}$ on most of the test sets except for the tst2011 set of the English-French task. However, the BLEU scores are modestly improved by combining the three factorized RNNTSMs.

The second experiment is to compare the phrasal bilingual tuples and the word bilingual tuples. Table 2 lists the vocabulary sizes of the tuples, source and target phrases. For the word-bilingual-tuple-I, the bilingual tuple vocabulary size increases by 10% in both the English-French and the English-German pairs. Compared with the phrasal bilingual tuples, the bilingual tuple vocabulary size in the word-bilingual-tuple-II slightly changes. In addition, decomposing the tuples is capable to greatly reduce the out-of-tuple-vocabulary rate by approximately 50% in the word-bilingual-tuple-II. Table 3 compares bilingual tuples in terms of BLEU scores. It can be clearly seen that both the word-bilingual-tuple-I and the word-bilingual-tuple-II achieve better performance than the phrasal bilingual tuple on most of the test sets. The BLEU improvements of the word-bilingual-tuple-II over the phrasal tuple range 1.2-1.5 points on the English-German task. The main reason may be lie in: the decomposition can provide word-to-word translation probabilities (such as "musicians:musiciens" in the example of Section 2) for those non-one-to-one alignments. Thus the translation system will have a translation option for an isolated occurrence of such words. Another important observation is that the decomposition performs differently on the English-French and English-German tasks. For example, there exists slight difference between the word-bilingual-tuple-I and the word-bilingual-tuple-II for the English-French task. However, for the English-German task, the word-bilingual-tuple-II significantly outperforms the word-bilingual-tuple-I by 0.4 BLEU scores. Lastly, we achieve modest improvements by combining the two types of word bilingual tuples.

This paper proposes three factorized RNNTSMs and two types of word bilingual tuples. In this experiment, we combine all of them (+Combination contains 6 models) and compare with RNN-based language model (Mikolov et al., 2010). Table 4 summarizes the results. As shown in Table 4 and Table 1, the combination can further enhance the performance on the English-German task. For example, the combination improves the factorized RNNTSM with the word-bilingual-tuple-I from 20.76 to 21.29 on the tst2012 set of the English-German task. Moreover, the combination significantly outperforms the RNNLM. The improvements over the RNNLMs on all test sets range 0.7-1.2 BLEU scores. The

	English-French			English-German		
	tst2010	tst2010	tst2012	tst2010	tst2010	tst2012
Baseline	$30.15_{(-1.2)}$	$35.97_{(-1.2)}$	$35.48_{(-1.5)}$	$20.29_{(-0.8)}$	$21.48_{(-0.9)}$	$19.30_{(-0.8)}$
+RNNLM	31.43	37.23	37.04	21.14	22.39	20.08
+Combination	$32.10_{(0.7)}$	$38.04_{(0.8)}$	$37.75_{(0.8)}$	$22.13_{(1.0)}$	$23.64_{(1.2)}$	$21.29_{(1.2)}$

Table 4: BLEU scores of the combination of our proposed models and RNNLM in the small task. The numbers in the parentheses are the absolute improvements over the RNNLM.

	English-French			English-German		
	tst2010	tst2010	tst2012	tst2010	tst2010	tst2012
Baseline	$32.92_{(-1.0)}$	$38.67_{(-1.2)}$	$39.41_{(-1.4)}$	$22.29_{(-0.5)}$	$23.67_{(-0.4)}$	$20.83_{(-0.7)}$
+RNNLM	33.93	39.90	40.82	22.80	24.12	21.49
+Combination	$34.24_{(0.3)}$	$40.37_{(0.5)}$	$40.92_{(0.1)}$	$23.61_{(0.8)}$	$25.18_{(1.1)}$	$22.64_{(1.1)}$

Table 5: BLEU scores of the combination of our proposed models and RNNLM. The numbers in the parentheses are the absolute improvements over the RNNLM.

improvement over the baseline are between 1.9-2.3 BLEU points.

5.2 Large Task

In the large task, the training data includes both speech-style and text-style bi-text corpora. The text-style bi-text corpora are collected from the WMT2012 campaign[5], including CommonCrawl, NewsCommentary, and Europarl. Totally, the numbers of the parallel sentences are 4.35M for the English-French task and 3.85M for the English-German task. The language model is obtained by linear interpolation of several 4-gram models trained on the target side of bi-text corpora and the LDC French Gigaword corpus.

Table 5 reports the results. +Combination means the combination of six models, as described in Table 4. We can observe that: (1) The combination of the proposed RNNTSMs only trained on the speech-style data can essentially enhance the baselines by 1.2-1.8 BLEU points. (2) The improvements over the RNNLMs are significant on the English-German task but these improvements are modest on the English-French task. Note that the factorized RNNTSMs and the RNNLMs in the large task are also only trained the speech-style parallel corpus. In future work, we will train them on a bigger corpus, which can be expected to further increase the performance (Auli et al., 2013; Wu et al., 2012).

6 Conclusion

Most prior neural network-based translation models either employ feed-forward neural networks to explicitly integrate source information via word-to-word alignment, or use recurrent neural networks in which source information is implicitly represented with a compressed vector. In this paper, we present recurrent neural network-based tuple sequence models (RNNTSMs) to compute probabilities of bilingual tuples in continuous space. One of major advantages is their potential to capture long-span history compared with feed-forward neural networks. In addition, our models can well address the data sparsity problem thanks to the fine-grained word bilingual tuples and the factorized recurrent neural networks. As can be concluded from the experimental results on the IWSLT2012 test sets, our factorized RNNTSMs with the proposed bilingual tuples can essentially improve the BLEU scores for the English-French and English-German tasks.

We plan to incorporate re-ordering and syntactic features into RNNTSMs and evaluate them on distant language pairs, such as English-Chinese (Japanese) tasks in the future. Moreover, we will prune large tuple vocabulary and speed up the training on bigger data.

[5] http://www.statmt.org/wmt12/translation-task.html

References

Ebru Arisoy, Tara N. Sainath, Brian Kingsbury, and Bhuvana Ramabhadran. 2012. Deep neural network language models. In *Proceedings of NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 20–28.

Michael Auli, Galley Michel, Quirk Chris, and Zweig Geoffrey. 2013. Joint language and translation modeling with recurrent neural networks. In *Proceedings of EMNLP2013*, pages 1044–1054.

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. In *Journal of Machine Learning Research*, pages 1137–1155.

Mikael Boden. 2002. A guide to recurrent neural networks and backpropagation. Technical report.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of ACL 2011*, pages 176–181.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(3):2493–2537.

Josep M. Crego and Francois Yvon. 2010. Factored bilingual n-gram language models for statistical machine translation. *Machine Translation, Special Issue: Pushing the frontiers of SMT*, 24(2):159–175.

Thomas Deselaers, Sasa Hasan, Oliver Bender, and Hermann Ney. 2009. A deep learning approach to machine transliteration. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 233–241.

Kevin Duh and Katrin Kirchhoff. 2004. Automatic learning of language model structure. In *Proceedings of COLING 2004*, pages 148–154.

Nadir Durrani, Helmut Schmid, and Alexander Fraser. 2011. A joint sequence translation model with integrated reordering. *Proceedings of ACL 2011*, pages 1045–1054.

Nadir Durrani, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013. Can markov models over minimal translation units help phrase-based smt? *Proceedings of ACL 2013*, pages 399–405.

Marcello Federico, Mauro Cettolo, Luisa Bentivogli, Michael Paul, and Sebastian Stuker. 2012. Overview of the iwslt 2012 evaluation campaign. In *Proceedings of IWSLT 2012*.

Alexander Fraser, Helmut Schmid, Richard Farkas, Renjing Wang, and Hinrich Schutze. 2013. Knowledge sources for constituent parsing of german, a morphologically rich and less-configurational language. *Computational Linguistics*, 39(1):57–85.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous transaltion models. In *Proceedings of EMNLP2013*.

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

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, and etc. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of ACL2007 on Interactive Poster and Demonstration Sessions*, pages 177–180.

Jose B. Marino, Rafael E. Banchs, Josep M. Crego, Adria de Gispert, Patrik Lambert, Jose A.R. Fonollosa, and Marta R. Costa-jussa. 2006. N-gram-based machine translation. In *Computational Linguistics*, volume Volume 32 Issue 4, pages 527–549.

Tomas Mikolov, Martin Karafiat, Lukas Burget, Jan. H. Cernocky, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of INTERSPEECH 2010*, pages 1045–1048.

Tomas Mikolov, Anoop Deoras, Stefan Kombrink, Lukas Burget, and Jan Honza Cernocky. 2011. Empirical evaluation and combination of advanced language modeling techniques. In *Proceedings of INTERSPEECH 2011*, pages 605–608.

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

Holger Schwenk, Daniel Dchelotte, and Jean-Luc Gauvain. 2006. Continuous space language models for statistical machine translation. In *Proceedings of COLING/ACL 2006*, pages 723–730.

Holger Schwenk, Marta R. Costa-jussa, and Jose A. R. Fonollosa. 2007. Smooth bilingual n-gram translation. In *Proceedings of EMNLP/HLT 2007*, pages 430–438.

Holger Schwenk. 2012. Continuous space translation models for phrase-based statistical machine translation. In *Proceedings of COLING 2012*, pages 1071–1080.

Frank Seide, Gang Li, Xie Chen, and Dong Yu. 2011. Feature engineering in context-dependentdeep neural networks for conversational speech transcription. In *Proceedings of ASRU 2011*, pages 24–29.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Chris Manning, Andrew Ng, and Chris Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP 2013*.

Le Hai Son, Alexandre Allauzen, and Francois Yvon. 2012. Continuous space translation models with neural networks. In *Proceedings of HLT-NAACL 2012*, pages 39–48.

Martin Sundermeyer, Ilya Oparin, Jean-Luc Gauvain, Ben Freiberg, Ralf Schlter, and Hermann Ney. 2013. Comparison of feedforward and recurrent neural network language models. In *Proceedings of ICASSP 2013*, pages 8430–8433.

Hua Wu and Haifeng Wang. 2007. Comparative study of word alignment heuristics and phrase-based smt. In *Proceedings of MT SUMMIT XI*, pages 305–312.

Youzheng Wu, Xugang Lu, Hitoshi Yamamoto, Shigeki Matsuda, Chiori Hori, and Hideki Kashioka. 2012. Factored language model based on recurrent neural network. In *Proceedings of COLING 2012*, pages 2835–2850.

Nan Yang, Shujie Liu, Mu Li, Ming Zhou, and Nenghai Yu. 2013. Word alignment modeling with context dependent deep neural network. In *Proceedings of ACL2013*, pages 166–175.

Hui Zhang, Kristina Toutanova, Chris Quirk, and Jianfeng Gao. 2013. Beyond left-to-right: Multiple decomposition structures for smt. *Proceedings of NAACL-HLT 2013*, pages 12–21.

Class-Based Language Modeling for Translating into Morphologically Rich Languages

Arianna Bisazza and **Christof Monz**
Informatics Institute, University of Amsterdam
Science Park 904, 1098 XH Amsterdam, The Netherlands
{a.bisazza,c.monz}@uva.nl

Abstract

Class-based language modeling (LM) is a long-studied and effective approach to overcome data sparsity in the context of n-gram model training. In statistical machine translation (SMT), different forms of class-based LMs have been shown to improve baseline translation quality when used in combination with standard word-level LMs but no published work has systematically compared different kinds of classes, model forms and LM combination methods in a unified SMT setting. This paper aims to fill these gaps by focusing on the challenging problem of translating into Russian, a language with rich inflectional morphology and complex agreement phenomena. We conduct our evaluation in a large-data scenario and report statistically significant BLEU improvements of up to 0.6 points when using a refined variant of the class-based model originally proposed by Brown et al. (1992).

1 Introduction

Class-based n-gram modeling is an effective approach to overcome data sparsity in language model (LM) training. By grouping words with similar distributional behavior into equivalence classes, class-based LMs have less parameters to train and can make predictions based on longer histories. This makes them particularly attractive in situations where n-gram coverage is low due to shortage of training data or to specific properties of the language at hand.

While translation into English has drawn most of the research effort in statistical machine translation (SMT) so far, there is now a growing interest in translating into languages that are more challenging for standard n-gram modeling techniques. Notably, morphologically rich languages are characterized by high type/token ratios (T/T) that reflect in high out-of-vocabulary word rates and frequent backing-off to low order n-gram estimates, even when large amounts of training data are used. These problems have been long studied in the field of speech recognition but much less in SMT, although the target LM is a core component of all state-of-the-art SMT frameworks.

Partly inspired by successful research in the field of speech recognition, various forms of class-based LMs have been shown to improve the quality of SMT when used in combination with standard word-level LMs. These approaches, however, have mostly focused on English (Uszkoreit and Brants, 2008; Dyer et al., 2011; Monz, 2011; Hassan et al., 2007; Birch et al., 2007) with only recent exceptions (Green and DeNero, 2012; Ammar et al., 2013; Wuebker et al., 2013; Durrani et al., 2014). Moreover, there is no published work that systematically evaluates different kinds of classes, model forms and LM combination methods in a unified SMT setting. On the contrary, most of the existing literature on LM combination uses mixtures of multiple *word*-level LMs for domain adaptation purposes.

This paper aims to fill these gaps by applying various class-based LM techniques to the challenging problem of translating *into* a morphologically rich language. In particular we focus on English-Russian, a language pair for which a fair amount of both parallel data and monolingual data has been provided by the Workshop of Machine Translation (Bojar et al., 2013). Russian is characterized by a rich inflectional morphology, with a particularly complex nominal declension (six core cases, three genders and two

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1918–1927, Dublin, Ireland, August 23-29 2014.

number categories). This results in complex agreement phenomena and an extremely rich vocabulary. Indeed, by examining our training data (see Section 4), we find the Russian T/T ratio to be almost two times higher than the English one.

Given this task, we make a number of contributions leading to a better understanding of ways to utilize class-based language models for translating into morphologically rich languages. We conduct a comparative evaluation of different target LMs along the following axes: (1) Classes: data-driven versus shallow morphology-based; (2) Model forms: simple class sequence (stream-based) versus original class-based (Brown et al., 1992); and (3) Combination frameworks: model-level log-linear combination versus word-level linear interpolation. When comparing the different model forms we pay particular attention to the role word emission probabilities play in class-based models, which turns out to be a significant factor for translating into morphologically rich languages. In this context we also evaluate for the first time a specific form of class-based LM called fullibm (Goodman, 2001) within statistical MT.

2 Class-based language models

As introduced by (Brown et al., 1992), the idea of class-based n-gram language modeling is to group words with similar distributional behavior into equivalence classes. The word transition probability is then decomposed into a class transition probability and a word emission probability:

$$P_{\text{class}}(w_i|w_{i-n+1}^{i-1}) = p_0(C(w_i)|C(w_{i-n+1}^{i-1})) \cdot p_1(w_i|C(w_i)) \quad (1)$$

This results in models that are more compact and more robust to data sparsity. Often, in the context of SMT, the word emission probability is dropped and only the class sequence is modeled. In this work, we refer to this model form as *stream-based* n-gram LM:[1]

$$P_{\text{stream}}(w_i|w_{i-n+1}^{i-1}) = p_0(C(w_i)|C(w_{i-n+1}^{i-1})) \quad (2)$$

Stream-based LMs are used, for instance, in factored SMT (Koehn et al., 2007), and in general many of the 'class-based LMs' mentioned in the SMT literature are actually of the latter form (2) (Dyer et al., 2011; Green and DeNero, 2012; Ammar et al., 2013; Chahuneau et al., 2013; Wuebker et al., 2013; Durrani et al., 2014). One exception is the work of Uszkoreit and Brants (2008), who incorporate word emission probabilities in their class-based LM used as an additional feature function in the log-linear combination (cf. Section 3.1). Interestingly, we are not aware of work that compares actual class-based LMs and stream-based LMs with respect to SMT quality.

While class-based LMs are known to be effective at counteracting data sparsity issues due to rich vocabularies, it is worth noting that they adhere to the fundamental constraints of n-gram modeling. Thus, grammatical agreement may be improved by a class-based LM approach only within a limited context window. Previous work that attempted to overcome this limitation includes (i) syntactic LMs for n-best reranking (Hasan et al., 2006; Carter and Monz, 2011) or integrated into decoding with significant engineering challenges (Galley and Manning, 2009; Schwartz et al., 2011) and (ii) unification-based constraints applied to a syntax-based SMT framework (Williams and Koehn, 2011).

We will now describe different kinds of word-to-class mapping functions used by class-based LMs. These can be completely data-driven or based on different sorts of linguistic or orthographic features.

2.1 Data-driven classes

The most popular form of class-based LMs was introduced by (Brown et al., 1992). In this approach, the corpus vocabulary is partitioned into a preset number of clusters by directly maximizing the likelihood of a training corpus. No linguistic or orthographic features are taken into account while training the classes.[2] Later work has focused on decreasing the large computational cost of the exchange algorithm proposed by Brown et al. (1992), either with a distributed algorithm (Uszkoreit and Brants, 2008) or by using a whole-context distributional vector space model (Schütze and Walsh, 2011). In this paper we use the standard SRILM implementation of Brown clustering.

[1] Not to be confused with the incrementally trainable stream-based LMs of Levenberg and Osborne (2009).

[2] Och (1999) extends a similar approach to bilingual clustering with the aim of generalizing the applicability of translation rules in an alignment template SMT framework.

2.2 Linguistic classes

Linguistic knowledge is another way to establish word equivalence classes. Common examples include lemma, part of speech and morphology-based classes, each of which can capture different aspects of the word sequence, such as the relative order of syntactic constituents or grammatical agreement. Hassan et al. (2007) and Birch et al. (2007) went as far as scoring n-grams of Combinatorial Categorial Grammar supertags. When using linguistic classes, one has to deal with the fact that the same word can belong to different classes when used in different contexts. Solutions to this problem include tagging the target word sequence as it is generated (Koehn et al., 2007; Birch et al., 2007; Green and DeNero, 2012), choosing the most probable class sequence for each phrase pair (Monz, 2011) or—even more lightweight—choosing the most probable class for each word (Bisazza and Federico, 2012).

Alternatively, simpler deterministic class mappings can be derived by using shallow linguistic knowledge, such as suffixes or orthographic features. The former can be obtained with a rule-based stemmer (as in this work), or, even more simply, by selecting the ϕ most common word suffixes in a training corpus and then mapping each word to its longest matching suffix (Müller et al., 2012). Orthographic features may include capitalization information or the presence of digits, punctuation or other special characters (Müller et al., 2012).

2.3 Hybrid surface/class models

Müller et al. (2012) obtain the best perplexity reduction when excluding frequent words from the class mapping. That is, each word with more than θ occurrences in the training corpus is assigned to a singleton class with word emission probability equal to 1. The frequency threshold θ is determined with a grid search on a monolingual held-out set. Optimal values for perplexities are shown to vary considerably among languages. In this work we follow this setup closely.

It is worth noting that Bisazza and Federico (2012) have applied a similar idea to the problem of style adaptation: they train a hybrid POS/word n-gram LM on an in-domain corpus and use it as an additional SMT feature function with the goal of counterbalancing the bias towards the style of the large out-of-domain data. The idea of modeling sequences of mixed granularity (word/subword) was earlier introduced to speech recognition by Yazgan and Saraçlar (2004).

The most extensive comparison of distributional, morphological and hybrid classes that we are aware of is the work by Müller et al. (2012), but that does not include any SMT evaluation. Looking at perplexity results over a large number of European language pairs (not including Russian), Müller et al. (2012) conclude that a hybrid suffix/word class-based LM simply built on frequency-based suffixes performs as well as a model trained on much more expensive distributional classes. Motivated by this finding, we evaluate these two kinds of classes in the context of SMT into a morphologically rich language.

2.4 Fullibm language model

As outlined above, the class-based LMs generally used in SMT are in fact stream-based models in the sense that they only estimate the probability of the class sequence (see Equation 2). However, the classic form of class-based LM (Brown et al., 1992) also includes a class-to-word emission probability $p_1(w_i|C(w_i))$ whose utility has not been properly assessed in the context of SMT.

Besides, we observe that a variety of class-based LM variants have been studied in speech recognition but not in SMT. In particular, Goodman (2001) presents a generalization of the standard class-based form where the word emission is also conditioned on the class history rather than on the current class alone. The resulting model is called *fullibm*:

$$P_{\text{fullibm}}(w_i|w_{i-n+1}^{i-1}) = p_0(C(w_i)|C(w_{i-n+1}^{i-1})) \cdot p_1(w_i|C(w_{i-n+1}^{i})) \tag{3}$$

We expect this model to yield more refined, context-sensitive word emission distributions which may result in better target LM probabilities for our SMT system.

3 SMT combining framework

Class-based LMs are rarely used in isolation, but are rather combined with standard word-level models. There exist at least two ways to combine multiple LMs into a log-linear SMT decoder: (i) as separate feature functions in the global log-linear combination or (ii) as components of a linear mixture counting as a single feature function in the global combination.

3.1 Log-linear combination

The standard log-linear approach to SMT allows for the combination of m arbitrary model components (or feature functions), each weighted by a corresponding weight α_m:

$$p(x|h) = \prod_m p_m(x|h)^{\alpha_m} \tag{4}$$

In typical SMT settings, $p_m(x|h)$ are phrase- or word-level translation probabilities, reordering probabilities, and so on. Treating the new LM as an additional feature function has the advantage that its weight can be directly optimized for SMT quality together with all other feature weights, using standard parameter tuning techniques (Och, 2003; Hopkins and May, 2011).

3.2 Linear interpolation

The other widely used combining framework is linear interpolation or mixture model:

$$p(x|h) = \sum_q \lambda_q p_q(x|h) \tag{5}$$

More specifically, word LMs are usually interpolated as a word-level weighted average of the n-gram probabilities:

$$p_{\text{mixLM}}(\mathbf{e}) = \prod_{i=1}^{n} \left(\sum_q \lambda_q p_q(e_i|h_i) \right) \tag{6}$$

The drawback of this approach is that the linear interpolation weights, or *lambdas*, cannot be set with standard SMT tuning techniques. Instead, interpolation weights are typically determined by maximizing the likelihood of a held-out monolingual data set, but this does not always outperform simple uniform weighting in terms of translation quality.[3]

Despite the lambda optimization issue, linear interpolation with uniform or maximum-likelihood weights has been shown to work better for SMT than log-linear combination when combining regular word n-gram LMs (Foster and Kuhn, 2007). However, to the best of our knowledge, the linear interpolation of word- and class-based LMs has never been tested in SMT.

In their intrinsic evaluation, Müller et al. (2012) show that linear mixing with hybrid class/surface models of various kinds consistently decrease the perplexity of a Kneser-Ney smoothed word-level LM, with relative improvements ranging between 3% (English) and 11% (Finnish). All their models are interpolated with class-specific lambda weights, according to the following formula:

$$P_{\text{mix}}(w_i|w_{i-n+1}^{i-1}) \;=\; \lambda_{C(w_{i-1})} \cdot P_{\text{class}}(w_i|w_{i-n+1}^{i-1}) \;+\; (1 - \lambda_{C(w_{i-1})}) \cdot P_{\text{word}}(w_i|w_{i-n+1}^{i-1}) \tag{7}$$

where P_{word} corresponds to the standard n-gram model using the lexical forms. Equation 7 can be seen as a generalization of the simple interpolation $\lambda P_{\text{class}} + (1 - \lambda) P_{\text{word}}$ used by Brown et al. (1992). The class-specific lambdas are estimated by a deleted interpolation algorithm (Bahl et al., 1991). In our experiments, we test both generic and class-specific lambda interpolation for SMT.

[3] Foster and Kuhn (2007) also tried more sophisticated techniques to set interpolation weights but did not obtain significant improvements.

Corpus	Lang.	#Sent.	#Tok.	T/T
paral.train	EN	1.9M	48.9M	.0107
	RU		45.9M	.0204
Wiki dict.	EN/RU	508K	–	–
mono.train	RU	21.0M	390M	.0068
newstest12	EN	3K	64K	–
newstest13		3K	56K	–

Table 1: Training and test data statistics: number of sentences, number of tokens and type/token ratio (T/T). All numbers refer to tokenized, lowercased data.

4 Evaluation

We perform a series of experiments to compare the effectiveness for SMT of various class mapping functions, different model forms, and different LM combining frameworks.

The task, organized by the Workshop of Machine Translation (WMT, Bojar et al. (2013)), consists of translating a set of news stories from English to Russian. As shown in Table 1, the available data includes a fairly large parallel training corpus (1.9M sentences) from various sources, a set of Wikipedia parallel headlines shared by CMU,[4] and a larger monolingual corpus for model training (21M sentences). By measuring the type/token ratios of the two sides of a parallel corpus, we can estimate the difference in morphological complexity between two languages: as shown in Table 1, the Russian T/T is almost two times higher than the English one (.0204 vs .0107) in the WMT13 parallel training data. As is usually the case, much more data is available for LM training. Nevertheless we report a rather high out-of-vocabulary word rate on the devsets' reference translations (2.28%).

4.1 Baseline

Our baseline is an in-house phrase-based (Koehn et al., 2003) statistical machine translation system very similar to Moses (Koehn et al., 2007). All system runs use hierarchical lexicalized reordering (Galley and Manning, 2008; Cherry et al., 2012), distinguishing between monotone, swap, and discontinuous reordering, all with respect to left-to-right and right-to-left decoding. Other features include linear distortion, bidirectional lexical weighting (Koehn et al., 2003), word and phrase penalties, and finally a word-level 5-gram target language model trained on all available monolingual data with modified Kneser-Ney smoothing (Chen and Goodman, 1999). The distortion limit is set to 6 and for each source phrase the top 30 translation candidates are considered.

The feature weights for all approaches were tuned by using pairwise ranking optimization (Hopkins and May, 2011) on newstest12. During tuning, 14 PRO parameter estimation runs are performed in parallel on different samples of the n-best list after each decoder iteration. The weights of the individual PRO runs are then averaged and passed on to the next decoding iteration. Performing weight estimation independently for a number of samples corrects for some of the instability that can be caused by individual samples.

4.2 Language models

The additional LMs are trained with Witten-Bell smoothing (Witten and Bell, 1991), which is a common choice for class-based LM training as Kneser-Ney smoothing cannot be used for computing discount factors when the count-of-counts are zero. The main series of experiments employ 5-gram models, but we also evaluate the usefulness of increasing the order to 7-gram (see Table 3).[5]

Data-driven clusters are learned with the standard Brown clustering algorithm, which greedily maximizes the log likelihood of a class bigram model on the training data. Following Ammar et al. (2013), we set the number of data-driven clusters to 600. In preliminary experiments we also tested a 256-cluster setting, but 600 yielded better BLEU scores. For time reasons, we train the clusters on a subset of the

[4] http://www.statmt.org/wmt13/wiki-titles.ru-en.tar.gz

[5] For this second series of experiments we use the feature weights tuned for the corresponding 5-gram LMs.

LM type	smoothing	vocab.	PP
words	Kneser-Ney	2.7M	270
Brown clusters	Witten-Bell	600	588
suffixes	Witten-Bell	968	2455
suffix/word hybrid (θ=5000)	Witten-Bell	8530	460

Linear interp.	PP	
	generic λ	class-spec.λ's
words + clusters	225	224
words + suffixes	266	265
words + hybrid	243	247

Table 2: Intrinsic evaluation of various types of LMs and their linear interpolations. Perplexity (PP) is computed on a separate held-out set of 5K Russian sentences. All models are 5-grams.

monolingual data including all the parallel data (news commentary) and the large commoncrawl corpus for a total of 1M sentences (22M tokens). We then map all monolingual data to the learned clusters and use that to train all our cluster-based LMs.

For the suffix-based class LMs we closely follow the setup of Müller et al. (2012) with the only difference that we use the Russian Snowball stemmer[6] to segment the vocabulary instead of frequency-based suffixes. The suffix threshold θ (see Section 2.3) is determined by minimizing perplexity on a separate held-out set (5K sentences): θ=5000 is the optimal setting among $\{2000, 5000, 10000, 20000\}$.[7] The same held-out set is used to estimate both the generic and the class-specific lambdas for the linear interpolation experiments.

Table 2 presents an overview of the LMs used in our experiments. We can see on the left side that all class-based LMs have notably higher perplexities compared to the word-level, with the fully suffix-based LM performing worst by far. Nevertheless, all class-based models yield a decrease in perplexity when they are interpolated with the word-level model (right side). The best improvement is achieved by the data-driven classes (225 versus 270, that is -17%), but the result of the hybrid LM is also quite successful (-10%) and much in line with the improvements reported by Müller et al. (2012) on other Slavic languages. Because the fully suffix-based LM yields only a modest reduction, we do not to include it in the SMT evaluation. The right side of Table 2 also shows that using class-specific interpolation weights is not significantly better, and sometimes is even worse than using only one generic λ, at least from the point of view of perplexity. Since weight estimation for linear interpolation is still an open problem for SMT, we decide nevertheless to compare these two interpolation methods in our translation experiments (see Table 4).

4.3 SMT results

Table 3 shows the results for English to Russian translation using log-linear combination with Brown clusters and the hybrid suffix/word classes. Translation quality is measured by case-insensitive BLEU (Papineni et al., 2002) on newstest13 using one reference translation. The relative improvements of the different class-based LM runs are with respect to the baseline which uses a word-based LM only and achieves comparable results to the state-of-the-art. We use approximate randomization (Noreen, 1989) to test for statistically significant differences between runs (Riezler and Maxwell, 2005).

We can see from Table 2(a) that using a stream-based LM as an additional feature, which is log-linearly interpolated with the other decoder features during parameter estimation, leads to small but statistically significant improvements. The results also indicate that using a higher n-gram class model (7-gram) does not yield additional improvements over a 5-gram class model, which is in contrast with the results reported by Wuebker et al. (2013) on a French-German task.

Since the stream-based models ignore word emission probabilities, one would expect further improvements from the theoretically more correct class-based model which include word emission probabilities (see Equation 1). Somewhat surprisingly, this is not the case. On the contrary, both 5- and 7-gram class-based models perform slightly worse than the stream-based models. We suspect that this is due to the limited context used to estimate the emission probabilities in the original Brown class-based models. To verify this we compared this to the fullibm model (Equation 3) which conditions word emission

[6] http://snowball.tartarus.org/algorithms/russian/stemmer.html

[7] Our training corpus is considerably larger than those used by Müller et al. (2012), therefore we search among higher values.

(a) Brown clusters (600)

Additional LM	surface BLEU	surface Δ	stem BLEU	stem Δ
⋆ none [baseline]	18.8	—	24.7	—
⋆ 5g stream-based	19.1	$+0.3^{\bullet}$	24.8	$+0.1$
7g stream-based	19.1	$+0.3^{\bullet}$	24.9	$+0.2$
⋆ 5g class-based	18.9	$+0.1$	24.6	-0.1
7g class-based	18.8	± 0.0	24.7	± 0.0
5g fullibm	19.4	$+0.6^{\bullet}$	25.0	$+0.3^{\bullet}$
7g fullibm	19.3	$+0.5^{\bullet}$	25.0	$+0.3^{\bullet}$

(b) Suffixes/words, $\theta = 5000$

Additional LM	surface BLEU	surface Δ	stem BLEU	stem Δ
⋆ none [baseline]	18.8	—	24.7	—
⋆ 5g stream-based	18.9	$+0.1$	24.6	-0.1
7g stream-based	18.9	$+0.1$	24.6	-0.1
⋆ 5g class-based	19.0	$+0.2^{\circ}$	24.8	$+0.1$
7g class-based	19.1	$+0.3^{\circ}$	24.7	± 0.0
5g fullibm	19.1	$+0.3^{\bullet}$	24.8	$+0.1$
7g fullibm	19.2	$+0.4^{\bullet}$	24.9	$+0.2^{\circ}$

Table 3: SMT translation quality on newstest13 when using different kinds of class-based language models as additional features in the log-linear combination. The settings used for weight tuning are marked with ⋆. Statistically significant differences wrt the baseline are marked with $^{\bullet}$ at the $p \leq .01$ level and $^{\circ}$ at the $p \leq .05$ level.

probabilities on the entire n-gram class history of length $n - 1$. The fullibm class-based models yield the biggest statistically significant improvements over the baseline and also compare favorably to the stream-based and original class-based models. Similarly to stream- and class-based models we do not observe a difference in performance between 5- and 7-gram models for fullibm.

Table 2(b) shows the results obtained by the shallow morphology-based classes inspired by Müller et al. (2012). This form of classes is easy to implement in many languages and computationally much cheaper than the Brown clusters. Although less than the data-driven class models, the hybrid suffix/word models also appear to improve translation quality. We can see that fullibm again yields the highest improvements, but we can also observe more consistent trends where longer n-grams help and class-based models are preferable to stream-based models without emission probabilities.

When translating into a morphologically rich language, such as Russian, the role of the target language model is two-fold. On the one hand, it helps choose the correct meaning from the available phrase translation candidates, on the other hand, it helps choose the correct surface realization of the translation candidate that agrees grammatically with the previous target context. For morphologically rich languages the second aspect plays a considerably larger role than for morphologically poor languages. To disentangle these two roles of the language model we also evaluated the different language models with respect to stem-based information only, stripping off any inflectional information using the Snowball stemmer. These results are also reported in Table 3 and in general exhibit the same trend as the surface-based BLEU scores. Again, fullibm performs best, and the original class-based LMs do not lead to any improvements over the baseline. As a general observation, we find that the surface-level gains are most of the time larger than the stem-level ones, which suggests that the additional LMs are mainly improving the choice of word inflections.

All systems compared in Table 3 use a class language model as an additional feature, which is log-linearly interpolated with the other decoder features. Alternatively, the word- and the class-based lan-

(a) Brown clusters (600)

Additional LM	surface BLEU	surface Δ	stem BLEU	stem Δ
⋆ none [baseline]	18.8	—	24.7	—
⋆ 5g class, log-linear comb.	18.9	$+0.1$	24.6	-0.1
⋆ 5g class, linear (global λ)	18.5	–0.3	24.4	-0.3
5g class, linear (class λ's)	18.6	–0.2	24.5	-0.2

(b) Suffixes/words, $\theta = 5000$

Additional LM	surface BLEU	surface Δ	stem BLEU	stem Δ
⋆ none [baseline]	18.8	—	24.7	—
⋆ 5g class, log-linear comb.	19.0	$+0.2^{\circ}$	24.8	$+0.1$
⋆ 5g class, linear (global λ)	18.9	$+0.1$	24.8	$+0.1$
5g class, linear (class λ's)	18.6	-0.1	24.6	-0.1

Table 4: SMT translation quality on newstest13 when using different LM combining frameworks: additional feature in the log-linear combination or linear interpolation with perplexity-tuned weights (one global lambda or class-specific lambdas).

guage models may be linearly interpolated with weights determined by maximizing the likelihood of a held-out monolingual data set (see Section 3.2). While linear interpolation often outperforms log-linear interpolation for combining language models for domain adaptation (Foster and Kuhn, 2007), this does not seem to be the case for language models for morphologically rich target languages. The results presented in Table 4 consistently show that linear interpolation under-performs log-linear combination under all conditions. Even using class-specific interpolation weights as suggested by Müller et al. (2012) did not lead to any further improvements.

5 Conclusion

We have presented the first systematic comparison of different forms of class-based LMs and different class LM combination methods in the context of SMT into a morphologically rich language.

First of all, our results have shown that careful modeling of class-to-word emission probabilities—often omitted from the models used in SMT—is actually important for improving translation quality. In particular, we have achieved best results when using a refined variant of the original class-based LM, called fullibm, which had never been tested for SMT but only for speech recognition (Goodman, 2001). Secondly, we have found that a rather simple LM based on shallow morphology-based classes can get close, in terms of BLEU, to the performance of more computationally expensive data-driven classes. Although the reported improvements are modest, they are statistically significant and obtained in a competitive large-data scenario against a state-of-the-art baseline.

On the downside, and somewhat in contrast with previous findings in domain adaptation, we have observed that linear interpolation of word- and class-based LMs with perplexity-tuned weights performs worse than the log-linear combination of models with model-level weights globally tuned for translation quality. This result was confirmed also when using class-specific lambdas as suggested by Müller et al. (2012).

Indeed, modeling morphologically rich languages remains a challenging problem for SMT but, with our evaluation, we have contributed to assess how far existing language modeling techniques may go in this direction. Natural extensions of this work include combining multiple LMs based on different, and possibly complementary, kinds of classes such as data-driven and suffix-based, or using supervised morphological analyzers instead of a simple stemmer. In a broader perspective, we believe that future research should question the fundamental constraints of n-gram modeling and develop innovative modeling techniques that conform to the specific requirements of translating into morphologically rich languages.

Acknowledgments

This research was funded in part by the Netherlands Organisation for Scientific Research (NWO) under project numbers 639.022.213 and 612.001.218. We kindly thank Thomas Müller for providing code and support for the weight optimization of linearly interpolated models.

References

Waleed Ammar, Victor Chahuneau, Michael Denkowski, Greg Hanneman, Wang Ling, Austin Matthews, Kenton Murray, Nicola Segall, Alon Lavie, and Chris Dyer. 2013. The CMU machine translation systems at WMT 2013: Syntax, synthetic translation options, and pseudo-references. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 70–77, Sofia, Bulgaria, August. Association for Computational Linguistics.

Lalit R. Bahl, Peter F. Brown, Peter V. de Souza, Robert L. Mercer, and David Nahamoo. 1991. A fast algorithm for deleted interpolation. In *Eurospeech*. ISCA.

Alexandra Birch, Miles Osborne, and Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 9–16, Prague, Czech Republic, June. Association for Computational Linguistics.

Arianna Bisazza and Marcello Federico. 2012. Cutting the long tail: Hybrid language models for translation style adaptation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 439–448, Avignon, France, April. Association for Computational Linguistics.

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria, August. Association for Computational Linguistics.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Simon Carter and Christof Monz. 2011. Syntactic discriminative language model rerankers for statistical machine translation. *Machine Translation*, 25(4):317–339.

Victor Chahuneau, Eva Schlinger, Noah A. Smith, and Chris Dyer. 2013. Translating into morphologically rich languages with synthetic phrases. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1677–1687, Seattle, Washington, USA, October. Association for Computational Linguistics.

Stanley F. Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 4(13):359–393.

Colin Cherry, Robert C. Moore, and Chris Quirk. 2012. On hierarchical re-ordering and permutation parsing for phrase-based decoding. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 200–209, Montréal, Canada, June. Association for Computational Linguistics.

Nadir Durrani, Philipp Koehn, Helmut Schmid, and Alexander Fraser. 2014. Investigating the usefulness of generalized word representations in SMT. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*, Dublin, Ireland, August.

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

George Foster and Roland Kuhn. 2007. Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 128–135, Prague, Czech Republic, June. Association for Computational Linguistics.

Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Morristown, NJ, USA. Association for Computational Linguistics.

Michel Galley and Christopher D. Manning. 2009. Quadratic-time dependency parsing for machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 773–781, Suntec, Singapore, August. Association for Computational Linguistics.

Joshua T. Goodman. 2001. A bit of progress in language modeling. *Computer Speech & Language*, 15(4):403–434.

Spence Green and John DeNero. 2012. A class-based agreement model for generating accurately inflected translations. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, ACL '12, pages 146–155, Stroudsburg, PA, USA. Association for Computational Linguistics.

Saša Hasan, Oliver Bender, and Hermann Ney. 2006. Reranking translation hypotheses using structural properties. In *Proceedings of the EACL'06 Workshop on Learning Structured Information in Natural Language Applications*, pages 41–48, Trento, Italy, April.

Hany Hassan, Khalil Sima'an, and Andy Way. 2007. Supertagged phrase-based statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 288–295, Prague, Czech Republic, June. Association for Computational Linguistics.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP'11)*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL 2003*, pages 127–133, Edmonton, Canada.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Abby Levenberg and Miles Osborne. 2009. Stream-based randomised language models for smt. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 756–764, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christof Monz. 2011. Statistical Machine Translation with Local Language Models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 869–879, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Thomas Müller, Hinrich Schütze, and Helmut Schmid. 2012. A comparative investigation of morphological language modeling for the languages of the European Union. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 386–395, Montréal, Canada, June. Association for Computational Linguistics.

Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley-Interscience.

Franz Josef Och. 1999. An efficient method for determining bilingual word classes. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 71–76.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

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

Stefan Riezler and John T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Hinrich Schütze and Michael Walsh. 2011. Half-context language models. *Comput. Linguist.*, 37(4):843–865, December.

Lane Schwartz, Chris Callison-Burch, William Schuler, and Stephen Wu. 2011. Incremental syntactic language models for phrase-based translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 620–631, Portland, Oregon, USA, June. Association for Computational Linguistics.

Jakob Uszkoreit and Thorsten Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *Proceedings of ACL-08: HLT*, pages 755–762, Columbus, Ohio, June. Association for Computational Linguistics.

Philip Williams and Philipp Koehn. 2011. Agreement constraints for statistical machine translation into german. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 217–226, Edinburgh, Scotland, July. Association for Computational Linguistics.

Ian H. Witten and Timothy C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Trans. Inform. Theory*, IT-37(4):1085–1094.

Joern Wuebker, Stephan Peitz, Felix Rietig, and Hermann Ney. 2013. Improving statistical machine translation with word class models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1377–1381, Seattle, Washington, USA, October. Association for Computational Linguistics.

Ali Yazgan and Murat Saraçlar. 2004. Hybrid language models for out of vocabulary word detection in large vocabulary conversational speech recognition. In *Proceedings of ICASSP*, volume 1, pages I – 745–8 vol.1, may.

Latent Domain Translation Models in Mix-of-Domains Haystack

Hoang Cuong and **Khalil Sima'an**
Institute for Logic, Language and Computation
University of Amsterdam
Science Park 107, 1098 XG Amsterdam, The Netherlands

Abstract

This paper addresses the problem of selecting adequate training sentence pairs from a mix-of-domains parallel corpus for a translation task represented by a small in-domain parallel corpus. We propose a novel latent domain translation model which includes domain priors, domain-dependent translation models and language models. The goal of learning is to estimate the probability of a sentence pair in mix-domain corpus to be in- or out-domain using in-domain corpus statistics as prior. We derive an EM training algorithm and provide solutions for estimating out-domain models (given only in- and mix-domain data). We report on experiments in data selection (intrinsic) and machine translation (extrinsic) on a large parallel corpus consisting of a *mix of a rather diverse set of domains*. Our results show that our latent domain invitation approach outperforms the existing baselines significantly. We also provide analysis of the merits of our approach relative to existing approaches.

Large parallel corpora are important for training statistical MT systems. Besides size, the *relevance* of a parallel training corpus to the translation task at hand can be decisive for system performance, cf. (Axelrod et al., 2011; Koehn and Haddow, 2012). In this paper we look at data selection where we have access to a large parallel data repository $\mathcal{C}_{mix}$, representing a rather varied mix of domains, and we are given a sample of in-domain parallel data $\mathcal{C}_{in}$, exemplifying a target translation task. Simply concatenating $\mathcal{C}_{in}$ with $\mathcal{C}_{mix}$ does not always deliver best performance, because including irrelevant sentences might be more harmful than beneficial, cf. (Axelrod et al., 2011). To make the best of available data, we must select sentences from $\mathcal{C}_{mix}$ for their relevance to translating sentences from $\mathcal{C}_{in}$.

Axelrod et al. (2011) and follow-up work, e.g., (Haddow and Koehn, 2012; Koehn and Haddow, 2012), select sentence pairs in $\mathcal{C}_{mix}$ using the cross-entropy difference between in- and mix-domain *language models*, both source and target sides, a modification of the Moore and Lewis method (Moore and Lewis, 2010). In the translation context, however, often a source phrase has different senses/translations in different domains, which cannot be distinguished with monolingual language models. The dependence of translation choice on domain suggests that the word alignments themselves can better be conditioned on domain information. However, in the data selection setting, corpus $\mathcal{C}_{mix}$ often does not contain useful domain markers, and $\mathcal{C}_{in}$ contains only a small sample of in-domain sentence pairs.

In this paper we present a *latent domain translation model* which weights every sentence pair $\langle \mathbf{f}, \mathbf{e} \rangle \in C_{mix}$ with a probability $P(D \mid \mathbf{f}, \mathbf{e})$ for being in-domain (D_1) or out-domain (D_0). Our model defines $P(\mathbf{e}, \mathbf{f}) = \sum_{D \in \{D_1, D_0\}} P(D) P(\mathbf{e}, \mathbf{f} \mid D)$, using a latent domain variable $D \in \{D_0, D_1\}$. Using bi-directional translation models, this leads to a domain prior $P(D)$, domain-dependent translation models $P_t(\cdot \mid \cdot, D)$ and language models $P_{lm}(\cdot \mid D)$ as in Equation 1:

$$P(\mathbf{e}, \mathbf{f} \mid D) \;=\; \frac{1}{2} \times \{P_{lm}(\mathbf{e} \mid D) P_t(\mathbf{f} \mid \mathbf{e}, D) \;+\; P_{lm}(\mathbf{f} \mid D) P_t(\mathbf{e} \mid \mathbf{f}, D)\} \tag{1}$$

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1928–1939, Dublin, Ireland, August 23-29 2014.

For efficiency we assume IBM Model I alignments $\mathbf{a}$ and translation tables $t(\cdot)$, e.g., $P_t(\mathbf{e} \mid \mathbf{f}, D) \propto \sum_{\mathbf{a}} \prod_i t(e_i \mid f_{\mathbf{a}_i}, D)$. Language models (LMs) P_{lm} are trained separately, albeit one problem not addressed by earlier work is how to train out-domain LMs given only in- and mix-domain data?

In our model, initially both the translation and LM probabilities estimated from $\mathcal{C}_{in}$ serve as *priors* for weighting sentence pairs in $\mathcal{C}_{mix}$ as being *more* relevant for in-domain translation *than not*. This initial weighting reveals *pseudo out-domain* data in $\mathcal{C}_{mix}$, which we use to train out-domain language models as well as initialize out-domain word alignment tables.[1] With these sharpened translation and language models, training commences using a version of EM (Dempster et al., 1977). Because the *potentially relevant data* in $\mathcal{C}_{mix}$ might be a superset of any in-domain data, the estimates from $\mathcal{C}_{in}$ serve merely as initial model estimates. Metaphorically, iterative EM training resembles `party invitations` on social networks (hence, the *Invitation model*): if initially in/out-domain sentence pairs (the `hosts`) `invite` some sentence pairs from $\mathcal{C}_{mix}$, in the next iteration the new `pseudo` in/out-domain sentences help invite more sentence pairs. In EM, sentence pairs receive weighted, rather than absolute, invitations from in- and out-domain models.

We present extensive experiments on a rather difficult selection task exploiting a large mix-domain corpus of 4.61M sentence pairs. Initially we conduct intrinsic evaluation on the mix-domain corpus where we also hide in-domain data and seek to retrieve it. Subsequently we conduct full MT experiments over the task. The results show that our Invitation model gives far better selections as well as translation performance than the baseline trained on the large data C_{mix}.

1 Invitation models of weighting and selection

By now training data selection from large mix-domain data is an accepted necessity, e.g., (Axelrod et al., 2011; Gascó et al., 2012; Haddow and Koehn, 2012; Banerjee et al., 2012; Irvine et al., 2013). Data selection has a different (but complementary) goal than domain adaptation, which aims at adapting an *existing out-domain system* by focusing on, e.g., translation model (Koehn and Schroeder, 2007; Foster and Kuhn, 2007; Sennrich, 2012), reordering model (Chen et al., 2013) and/or language model adaptation (Eidelman et al., 2012). Our setting is in line with data selection approaches (Moore and Lewis, 2010; Axelrod et al., 2011; Duh et al., 2013), and is somewhat related to phrase pair weighting (Matsoukas et al., 2009; Foster et al., 2010). In this paper we explicitly draw attention to the special case of a mix-domain parallel corpus consisting of a large and rather diverse set of domains.

Our model assigns to every sentence pair $\langle \mathbf{f}, \mathbf{e} \rangle \in \mathcal{C}_{mix}$ a probability as in Equation 2:

$$\begin{aligned} P(D \mid \mathbf{f}, \mathbf{e}) &= \frac{P(\mathbf{f}, \mathbf{e}, D)}{\sum_{D \in \{D_1, D_0\}} P(\mathbf{f}, \mathbf{e}, D)} \\ P(\mathbf{f}, \mathbf{e}, D) &= \frac{1}{2} \times P(D) \times \{P_{lm}(\mathbf{e} \mid D) P_t(\mathbf{f} \mid \mathbf{e}, D) + P_{lm}(\mathbf{f} \mid D) P_t(\mathbf{e} \mid \mathbf{f}, D)\} \end{aligned} \tag{2}$$

Viewed as learning two latent corpora $\mathcal{C}_1$ and $\mathcal{C}_0$, the task is to assign every $\langle \mathbf{f}, \mathbf{e} \rangle \in C_{mix}$ an expected count $P(D_x \mid \mathbf{f}, \mathbf{e})$ that it is in $\mathcal{C}_x \in \{C_0, C_1\}$. Next we discuss the model components each in turn.

The domain-dependent translation models $P_t(\cdot \mid D)$ can be viewed as modeling the probability that $\mathbf{e}$ translates as $\mathbf{f}$ in domain $D \in \{D_0, D_1\}$. Given $\mathbf{f} = f_1, f_2, \ldots, f_m$ and $\mathbf{e} = e_1, e_2, \ldots, e_l$, we assume (hidden) alignments $\mathbf{a} = a_1, a_2, \ldots, a_m$ akin to IBM Model I (Brown et al., 1993):

$$P_t(\mathbf{f}, \mathbf{a} \mid \mathbf{e}, D) = \frac{\epsilon}{(l+1)^m} \prod_{j=1}^{m} t(f_j | e_{a_j}, D) \tag{3}$$

$$P_t(\mathbf{f} \mid \mathbf{e}, D) = \sum_{\mathbf{a}} P_t(\mathbf{f}, \mathbf{a} | \mathbf{e}, D) = \frac{\epsilon}{(l+1)^m} \prod_{j=1}^{m} \sum_{i=0}^{l} t(f_j | e_i, D). \tag{4}$$

[1] Earlier work on data selection exploits the contrast between in-domain and mix-domain instead of (pseudo) out-domain language models. However, the mix-domain language models trained on a mix of rather diverse set of domains could be considered kind of wide-coverage, which makes for a rather weak contrast with the in-domain language models.

where $t(f_j|e_{a_j}, D)$ is the domain-dependent lexical probability of f_j given e_{a_j} with respect to D. One crucial aspect about model inspired by IBM-Model-I is that $P_t(\mathbf{f} \mid \mathbf{e}, D)$ can be estimated efficiently, as in Equation 4. This makes the training particularly efficient as detailed in Section 2

The in-/out-domain source and target language models are not the same as in previous work, e.g., (Axelrod et al., 2011), which employ in-/mix-domain language models. This makes explicit the difficulty in finding data to train out-domain language models, and we present a solution in Section 2.

The domain priors $P(D_1)$ and $P(D_0)$ represent the percentage of the pairs that are in-/ and out domain respectively in $\mathcal{C}_{mix}$ learned by our model. Their estimate during training might be a reasonable selection cut-off threshold. However, we found that it is not entirely clear whether these cut-off criteria might exclude other relevant/irrelevant pairs that are not exactly in-domain. We leave this extension for future work.[2]

Finally, it should be noted that the domain-dependent word alignment model, $t(f|e, D)$ is a generalization of the standard (domain-independent) word alignment model, $t(f|e)$, in which, $t(f|e, D) = \frac{t(f|e)t(D|f,e)}{\sum_f t(f|e)t(D|f,e)}$. Here, $t(D|f, e)$ can be thought of as the *latent word-relevance models*, i.e., the probability that a word pair is relevant for in- (D_1) or out-domain (D_0). Empirical results (beyond the scope of this work) show that training the latent in-domain alignment model, $t(f|e, D_1)$ often gives better translation systems than training the standard (domain-independent) alignment model, $t(f|e)$.

2 Training

With all language models trained separately, our selection model can be viewed to have two sets of domain-dependent parameters $\Theta = \{\Theta_{D_0}, \Theta_{D_1}\}$. The parameters Θ_D consist of the domain-dependent lexical parameters (e.g., $t_{\Theta_D}(f|e, D)$, $t_{\Theta_D}(e|f, D)$) and the domain prior parameter (e.g., $P_{\Theta_D}(D)$). Our training procedure seeks the parameters Θ that maximize the log-likelihood of $\mathcal{C}_{mix}$:

$$\mathcal{L} = \sum_{\mathbf{f},\mathbf{e}} \log P_\Theta(\mathbf{f}, \mathbf{e}) \quad = \quad \sum_{\mathbf{f},\mathbf{e}} \log \sum_D \sum_{\mathbf{a}} P_{\Theta_D}(\mathbf{a}, D, \mathbf{f}, \mathbf{e}) \tag{5}$$

Because of the latent variables $\mathbf{a}$ and D, there is no closed form solution and the model is fit using the EM algorithm (Dempster et al., 1977). EM can be seen to maximize $\mathcal{L}$ via block-coordinate ascent on a lower bound $\mathcal{F}(q, \Theta)$ using an auxiliary distribution over the latent variables $q(\mathbf{a}, D|\mathbf{f}, \mathbf{e})$

$$\mathcal{L} \quad \geq \quad \mathcal{F}(q, \Theta) = \sum_{\mathbf{f},\mathbf{e}} \sum_D \sum_{\mathbf{a}} q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e}) \log \frac{P_{\Theta_D}(\mathbf{a}, D, \mathbf{f}, \mathbf{e})}{q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e})} \tag{6}$$

where the inequality results from log being concave and Jensen's inequality. We rewrite the Free energy $\mathcal{F}(q, \Theta)$ (Neal and Hinton, 1999) as follows:

$$\begin{aligned}
\mathcal{F}(q, \Theta) \quad &= \quad \sum_{\mathbf{f},\mathbf{e}} \sum_D \sum_{\mathbf{a}} q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e}) \log \frac{P_{\Theta_D}(\mathbf{a}, D, \mathbf{f}, \mathbf{e})}{q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e})} \\
&= \sum_{\mathbf{f},\mathbf{e}} \sum_{D,\mathbf{a}} q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e}) \log \frac{P_{\Theta_D}(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e})}{q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e})} \; + \; \sum_{\mathbf{f},\mathbf{e}} \sum_{D,\mathbf{a}} q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e}) \log P_\Theta(\mathbf{f}, \mathbf{e}) \\
&= \sum_{\mathbf{f},\mathbf{e}} \log P_\Theta(\mathbf{f}, \mathbf{e}) - KL[q(\mathbf{a}, D \mid \mathbf{f}, \mathbf{e}) \;||\; P_{\Theta_D}(\mathbf{a}, D|\mathbf{f}, \mathbf{e})]
\end{aligned} \tag{7}$$

where $KL[\cdot||\cdot]$ is the KL-divergence. To find $q^*(\mathbf{a}, D|\mathbf{f}, \mathbf{e})$ that maximizes $\mathcal{F}(q, \Theta)$:

$$\begin{aligned}
q^*(\mathbf{a}, D|\mathbf{f}, \mathbf{e}) \quad &= \quad \operatorname*{argmax}_{q(\mathbf{a},D|\mathbf{f},\mathbf{e})} \mathcal{F}(q, \Theta) = \operatorname*{argmin}_{q(\mathbf{a},D|\mathbf{f},\mathbf{e})} KL[q(\mathbf{a}, D|\mathbf{f}, \mathbf{e})||P_{\Theta_D}(\mathbf{a}, D|\mathbf{f}, \mathbf{e})] \\
&= \quad P_{\Theta_D}(\mathbf{a}, D|\mathbf{f}, \mathbf{e}) = P_{\Theta_D}(D|\mathbf{f}, \mathbf{e}) P_{\Theta_D}(\mathbf{a}|\mathbf{f}, \mathbf{e}, D).
\end{aligned} \tag{8}$$

[2]We especially thank an anonymous reviewer who gave valuable comments related to this point.

Here

$$P_{\Theta_D}(\mathbf{a}|\mathbf{f},\mathbf{e},D) \;=\; \frac{P_{\Theta_D}(\mathbf{f},\mathbf{a}|\mathbf{e},D)}{P_{\Theta_D}(\mathbf{f}|\mathbf{e},D)} \;=\; \frac{\prod_{j=1}^{m} t(f_j|e_{a_j},D)}{\prod_{j=1}^{m}\sum_{i=0}^{l} t(f_j|e_i,D)} \tag{9}$$

The distribution $q^*(\mathbf{a},D|\mathbf{f},\mathbf{e})$ together with $q^*(D|\mathbf{f},\mathbf{e}) = \sum_{\mathbf{a}} q^*(\mathbf{a},D|\mathbf{f},\mathbf{e}) = P_{\Theta_D}(D|\mathbf{f},\mathbf{e})$ can be used to softly fill in the values of $\mathbf{a}$ and D respectively to estimate model parameters.

We now state our derived EM update formulas. We use the notation $P^{(c)}$ and $t^{(c)}$ for current iteration estimates, and $P^{(+)}$ and $t^{(+)}$ for the re-estimates. We denote the expected counts that e aligns to f in the translation $(\mathbf{f}|\mathbf{e})$ with respect to a domain D with $c(f|e;\mathbf{f},\mathbf{e},D)$. Similarly, we denote the expected count of $(\mathbf{f}|\mathbf{e})$ with respect to a domain D by $c(D;\mathbf{f},\mathbf{e})$.

E-step $\forall D \in \{D_0, D_1\}$ do

$$\begin{aligned} c(D;\mathbf{f},\mathbf{e}) &= P^{(c)}(D \mid \mathbf{f},\mathbf{e}) \\ c(f|e;\mathbf{f},\mathbf{e},D) &= P^{(c)}(D \mid \mathbf{f},\mathbf{e}) \frac{t^{(c)}(f \mid e,D)}{\sum_{i=0}^{l} t^{(c)}(f \mid e_i,D)} \sum_{j=1}^{m} \delta(f,f_j) \sum_{i=0}^{l} \delta(e,e_i) \end{aligned}$$

M-step $\forall D \in \{D_0, D_1\}$ do

$$t^{(+)}(f|e,D) = \frac{\sum_{\mathbf{f},\mathbf{e}} c(f|e;\mathbf{f},\mathbf{e},D)}{\sum_{f}\sum_{\mathbf{f},\mathbf{e}} c(f|e;\mathbf{f},\mathbf{e},D)} \qquad P^{(+)}(D) = \frac{\sum_{\mathbf{f},\mathbf{e}} c(D;\mathbf{f},\mathbf{e})}{\sum_{D}\sum_{\mathbf{f},\mathbf{e}} c(D;\mathbf{f},\mathbf{e})}$$

To re-estimate $P(D \mid \mathbf{f},\mathbf{e})$ we substitute the M-step estimates into Equations 3, 4 and 2. We initialize translation tables $t(f|e,D_1)$ and $t(e|f,D_1)$ with non-zero estimates obtained from applying IBM model I to in-domain corpus $\mathcal{C}_{in}$.[3] Before EM training starts we must train the LMs. The in-domain LMs $P_{lm}(\mathbf{e}|D_1)$ and $P_{lm}(\mathbf{f}|D_1)$ are trained on the source and target sides of $\mathcal{C}_{in}$ respectively. For the out-domain LMs $P_{lm}(\mathbf{e}|D_0)$ and $P_{lm}(\mathbf{f}|D_0)$ we need an *out-domain* data set to train them. It would also be reasonable to use the set to train the out-domain tables, $t(\cdot \mid \cdot, D_0)$. This raises an hitherto unattended question regarding how to construct such an out-domain data set.

Inspired by burn-in in sampling, initially we isolate all LMs from our model to train the translation models for a single EM iteration; we initialize the model with a translation table constructed on $\mathcal{C}_{in}$ and uniform otherwise. Using the re-estimates, we score sentence pairs in $\mathcal{C}_{mix}$ with $P(D_1|\mathbf{f},\mathbf{e})$ and select a *burn-in subset* of smallest scoring pairs as *pseudo out-domain data* which can be used to train $P_{lm}(\mathbf{e}|D_0)$ and $P_{lm}(\mathbf{f}|D_0)$. Choosing the optimal size of this subset is difficult, but in practice, we usually choose a subset that has similar size (number of words) to the given in-domain corpus. The rationale behind this choice is to avoid the risk that pseudo out-domain models would dominate the in-domain models during further training. We observe that choosing the same size for a pseudo out-domain corpus is not guaranteed to always give optimal performance, and this point deserves further study.

Finally, once the domain-dependent LMs have been trained, the domain-dependent LM probabilities stay *fixed* during EM. Crucially, it is important to scale the probabilities of the four LMs to make them comparable: we normalize the probability that a LM assigns to a sentence by the total probability this LM assigns to all sentences in $\mathcal{C}_{mix}$.

3 Experimental setting

We carry out experiments in data selection (intrinsic) as well as in machine translation (extrinsic). We build an English-Spanish mix-domain corpus consisting of a large and rather varied set of domains (a

[3]Note that in practice, we usually use only one iteration to train IBM Model I. To simplify the implementation, we ignore factor $\frac{\epsilon}{(l+1)^m}$ in the model (Equation 3), which serves a minor role. It should be also noted that we set a (small) threshold, e.g., $t(\cdot|\cdot,\cdot) = 0.0001$ for all word pairs that do not occur in the in-domain corpus to avoid over-fitting.

haystack) in a way that allows us to directly measure selection quality. Starting out from a general-domain corpus $\mathcal{C}_g$ consisting of $4.51M$ sentence pairs, collected from multiple resources including EuroParl (Koehn, 2005), Common Crawl Corpus, UN Corpus, News Commentary, TAUS Software, TAUS Hardware, and TAUS Pharmacy, and a 177K in-domain (TAUS Legal) sentence pairs.

We create $\mathcal{C}_{mix}$ by selecting an arbitrary 100K pairs of in-domain set and adding them to $\mathcal{C}_g$; the remaining 77K in-domain pairs constitute $\mathcal{C}_{in}$. We think of this as `hiding` in-domain data in $\mathcal{C}_{mix}$ so we can evaluate our ability to retrieve it; in this setting we can evaluate selection directly using pseudo-precision/recall defined as the percentage of selected in-domain pairs to the total selected or to the hidden 100K pairs respectively.

Table 1 summarizes the data and the translation task. It should be noted that a mix-domain corpus, that contains a large and rather varied set of domains, frequently contains subsets with a vocabulary that is close to the in-domain adaptation task; in this case, e.g., Europarl and TAUS Legal share big portions of their source vocabulary, whereas their translations could differ. This makes the selection task far more difficult than assumed by previous approaches as we will show next.

Task	**Corpora**	**English**	**Spanish**
	Mix-Domain Corpus (4.51M sents)	$125,339,057$	$139,655,311$
TAUS Legal	In-Domain Corpus (77K sents)	$1,555,342$	$1,733,370$
	Dev (2K sents)	$27,983$	$30,501$
	Test (2K sents)	$45,736$	$48,999$

Table 1: The data preparation - training, dev and testing corpora (size in words). Note that the dev set contains sentences of 10-25 words, while the test set contains sentences that vary substantially in length, from 5-10 words up to 45-50 words.

Our Invitation model takes 3 EM-iterations to train.[4] We then weigh sentence pairs under our model with $P(D_1 \mid \mathbf{e}, \mathbf{f})$. We test various baseline models, including the bilingual cross-entropy difference model, and the two cross-entropy difference models (on the source language and on the target language).[5] We report *pseudo*-precision/recall at the sentence-level using a range of cut-off criteria for selecting the top scoring instances in the mix-domain corpus.

We use Moses (Koehn et al., 2007) with GIZA++ (Och and Ney, 2003) and k-best batch MIRA (Cherry and Foster, 2012). Final MT systems use the same *non-adapted language models* trained on 2.2M English Europarl sentences plus 248.8K sentences from News Commentary Corpus (WMT 2013).

We report BLEU (Papineni et al., 2002), METEOR 1.4 (Denkowski and Lavie, 2011) and TER (Snover et al., 2006). Statistical significance uses 95% confidence intervals using paired bootstrap re-sampling (Press et al., 1992; Koehn, 2004). The k-best batch MIRA optimizer (Cherry and Foster, 2012) was run at least three times to optimize any SMT system to avoid instability (Clark et al., 2011).[6]

4 Results

Table 2 presents the results showing substantial improvement in selection performance compared to all the baselines. Subsequently we build SMT systems over the selected subsets. We report the translation yielded by these systems over the task in Table 2 as well. It can be easily seen that the baseline approaches that simply train on in- and mix-domain data do not work that well for a difficult selection task from a mix-domain corpus consisting of a large and rather diverse set of domains. The SMT sys-

[4]To train the LM probs, we construct interpolated 4-gram Kneser-Ney language models using BerkeleyLM (Pauls and Klein, 2011). This setting for training language models is used for all experiments in this work.

[5]The script we use to train these models is developed by Luke Orland and available at: `https://github.com/lukeorland/moore\_and\_lewis\_data\_selection`.

[6]Note that metric scores for the systems are averages over multiple runs.

Cut-off	Model	In-domain Pairs	pseudo-Precision	pseudo-Recall	BLEU	METEOR	TER
50K	CE Difference (source side)	370	0.74	0.37	20.5	28.0	62.3
	CE Difference (target side)	375	0.75	0.38	19.3	26.8	63.3
	Bilingual CE Difference	413	0.83	0.41	18.7	26.3	64.3
	Invitation	19156	38.31	19.16	36.5	36.4	47.1
100K	CE Difference (source side)	592	0.59	0.59	24.8	30.8	57.8
	CE Difference (target side)	572	0.57	0.57	22.1	29.7	60.1
	Bilingual CE Difference	649	0.65	0.65	23.1	30.0	58.9
	Invitation	30474	30.47	30.47	37.1	36.9	47.0
150K	CE Difference (source side)	753	0.50	0.75	26.4	32.0	56.2
	CE Difference (target side)	742	0.49	0.74	23.9	31.2	58.8
	Bilingual CE Difference	793	0.53	0.79	24.4	30.9	58.1
	Invitation	38424	25.62	38.42	37.1	37.0	46.7
200K	CE Difference (source side)	874	0.44	0.87	26.6	32.4	56.0
	CE Difference (target side)	888	0.44	0.88	25.8	32.1	57.2
	Bilingual CE Difference	932	0.93	0.65	25.7	32.0	57.0
	Invitation	44392	22.17	44.39	37.5	37.4	46.2
250K	CE Difference (source side)	994	0.40	0.99	27.3	32.8	55.4
	CE Difference (target side)	997	0.40	0.10	26.3	32.4	56.3
	Bilingual CE Difference	1062	0.42	1.06	26.6	32.7	55.6
	Invitation	49419	19.77	49.42	37.3	37.3	46.1
300K	CE Difference (source side)	1122	0.37	1.12	28.2	33.4	54.5
	CE Difference (target side)	1093	0.36	1.09	26.4	32.7	56.0
	Bilingual CE Difference	1169	0.39	1.17	27.8	33.3	54.9
	Invitation	53892	17.96	53.89	37.7	37.5	46.0

Table 2: Systematic comparison between selection models.

tems trained on the selection of our model perform significantly and consistently better (with p-value $= 0.0001$ for all cases) than the others trained on the selection of the baselines.

	Sentences
	Bilingual CE Difference
1	*by assisting in the placement and financing of used and end-of-lease aircraft , atr asset management has helped broaden atr 's customer base , notably in emerging markets , by providing quality reconditioned aircraft at attractive prices and has helped maintain residual values of used aircraft .* *al participar en la colocación y en la financiación de los aviones usados al final del período de arrendamiento , atr gestión de activos ha podido ampliar la base de su clientela , en particular en los países de economías emergentes , al proporcionar aparatos entregados en buen estado a precios interesantes y ha contribuido a mantener el valor residual de los aviones usados .*
2	*in contrast , recent improvements in western europe are not expected to be reversed significantly .* *en cambio no se espera que las recientes mejoras en europa occidental se inviertan significativamente .*
3	*creating xml file ...* *creando el archivo xml ...*
	Invitation Model
1	*as she has said , the harmonisation of the requirements for information to appear on the invoice will mean that traders operating within the single market will be subject to a single legislation , while until now they have had to know , comply with and apply fifteen different legislations .* *como ella ha dicho , la armonización de los requisitos de información que deben constar en la factura permitirá a los comerciantes que operen en el mercado interior sujetarse a una sola legislación , mientras que hasta ahora tenían que conocer , sujetarse y aplicar quince legislaciones diferentes .*
2	*the solicitation documents shall specify the estimated period of time following dispatch of the notice of acceptance that will be required to obtain the approval .* *en el pliego de condiciones se indicará el plazo de tiempo previsto , a partir de la expedición del aviso de aceptación , que será requerido para obtener la aprobación .*
3	*there is no doubt that disadvantages will result for the consumer and for the manufacturer of branded goods , for example with regard to consumer health protection .* *ello generará , sin duda alguna , desventajas para el consumidor y el productor de artículos de marca , entre otros aspectos también en lo que se refiere a la protección de la salud del consumidor .*

Table 3: Top pairs from mix-domain corpus with highest scores according to models.

Table 3 presents some random top ranked sentence pairs from the bilingual cross-entropy difference

Model	Cut-off: 50K English	Cut-off: 50K Spanish	Cut-off: 100K English	Cut-off: 100K Spanish	Cut-off: 200K English	Cut-off: 200K Spanish
CE Difference (source side)	8.65	8.70	11.92	12.21	15.50	16.22
CE Difference (target side)	8.14	10.09	11.61	14.13	15.45	18.50
Bilingual CE Difference	7.03	8.16	10.38	11.96	14.34	16.43
Invitation	40.16	44.70	37.30	41.59	34.32	38.32

Table 4: Average words in selected sentences.

model against our Invitation model for the task. This shows clearly more relevant pairs for our selection model than for the baselines. It should be noted that the baseline models tend to prefer shorter sentences, while our model suffers less from this kind of bias. Table 4 presents the average length (in words) of selected sentences selected by different models over various cut-offs.

Cut-off	Model	In-domain Pairs	pseudo-Precision	pseudo-Recall	BLEU	METEOR	TER
300K	**Without Translation Model**	34156	11.39	34.16	35.8	36.6	47.3
	Without Language Model	51991	17.33	51.99	37.4	37.4	46.6
	Full model	53892	17.96	53.89	37.7	37.5	46.0

Table 5: Experiments exploring the roles of individual components in our model.

Which component type (language or translation models) contributes more to performance? We neutralize each component in turn and build a selection system with the remaining model parameters. Table 5 shows translation models are crucial for performance, while domain-dependent LMs make a small, yet noteworthy contribution. It should also be noted that using the LMs derived separately from in- and out-domain data yields far better performance than the LMs derived from in- and mix-domain data for this task.

System	Phrases	BLEU	METEOR	TER
Large data C_{mix}	236.74M	36.8	37.2	47.1
Subset of 300K	22.47M	37.7	37.5	46.0

Table 6: Translation accuracy comparison.

Finally, we compare a system trained on a selection of the top scored 300K sentences to a baseline large-scale SMT system trained on C_{mix} (4.61M sentences). The baseline trained on C_{mix} works with 236.74M phrase pairs, whereas the Invitation trained system employs a small table of 22.47M phrases. Tabel 6 shows the results. It is interesting that the small MT system trained by Invitation performs significantly better (with p-value $= 0.0001$ for all metrics) than the large-scale system baseline trained on all of C_{mix}.

Input	*cada estado miembro **supervisará** la categoría científica de la **evaluación** y las actividades de los miembros de los comités y de los expertos que haya designado, pero se **abstendrá** de darles instrucciones incompatibles con las funciones que les competen.*
Reference	*each member state shall **monitor** the scientific level of the **evaluation** carried out and supervise the activities of members of the committees and the experts it nominates, but shall **refrain** from giving them any instruction which is incompatible with the tasks incumbent upon them.*
Large C_{mix}	*each member state will **oversee** the category scientific **assessment** and the activities of members of the committees and experts which designated, but **abstain** of instruct incompatible with their regulatory functions.*
Subset 300K	*each member state will **monitor** the scientific category of the **evaluation** and the activities of the members of the committees and of experts who has designated, but **refrain** from giving them instructions incompatible with the required functions assumed.*

Table 7: Translation example yielded by systems.

To give a sense of the improvement in translation, we present an example in Table 7. The example is indeed illuminating because it shows the difference in choice between the mix-domain system and

our selection-trained system. The example shows different translation pairs: ⟨supervisará-*monitor*⟩ vs. ⟨supervisará-*oversee*⟩, ⟨evaluación-*evaluation*⟩ vs. ⟨evaluación-*assessment*⟩, and ⟨abstendrá de-*refrain from*⟩ vs. ⟨abstendrá de-*abstain*⟩. Table 8 presents phrase table entries, i.e., $p(e \mid f)$ and $p(f \mid e)$, for the pairs of words in each system.

		supervisará		**evaluación**		**abstendrá de**	
System	**Entry**	**monitor**	**oversee**	**evaluation**	**assessment**	**refrain from**	**abstain**
Large data C_{mix}	$\phi(e\|f)$	0.002	0.020	0.579	0.429	0.002	0.013
	$\phi(f\|e)$	0.119	0.081	0.391	0.403	0.014	0.060
Subset of 300**K**	$\phi(e\|f)$	0.012	0.024	0.487	0.357	0.015	–
	$\phi(f\|e)$	0.203	0.072	0.338	0.417	0.143	–

Table 8: Phrase entry examples. Note that the system trained on the subset of top 300K pairs of sentences does not contain the phrase pair ⟨*refrain from-abstain*⟩.

5 Final Machine Translation experiments: Putting all data together

For final adaptation evaluations we follow (Koehn and Schroeder, 2007; Nakov, 2008) and (Axelrod et al., 2011; Sennrich, 2012), by passing multiple phrase tables directly to the Moses decoder and tuning a system using these different tables together. Table 9 presents the result, showing the consistent improvement of adaptation with Invitation model compared to the baselines (with p-value $= 0.0001$ for all cases) over the mixture data C_{mix}.

Data	**System**	**BLEU**	**METEOR**	**TER**
	In-domain	36.66	37.19	44.76
50K	+ CE Difference (source side)	37.1	36.7	48.1
	+ CE Difference (target side)	37.1	36.6	48.2
	+ Bilingual CE Difference	37.1	36.6	48.2
	+ **Invitation**	38.0	37.2	47.3
100K	+ CE Difference (source side)	37.3	36.8	47.9
	+ CE Difference (target side)	37.2	36.8	48.0
	+ Bilingual CE Difference	37.2	36.8	48.0
	+ **Invitation**	38.4	37.4	46.9
150K	+ CE Difference (source side)	37.1	36.9	48.2
	+ CE Difference (target side)	37.3	36.9	47.9
	+ Bilingual CE Difference	37.0	36.8	48.1
	+ **Invitation**	38.6	37.5	46.6
200K	+ CE Difference (source side)	37.3	36.9	47.7
	+ CE Difference (target side)	37.3	36.9	47.9
	+ Bilingual CE Difference	37.3	36.9	47.8
	+ **Invitation**	38.4	37.6	46.7
250K	+ CE Difference (source side)	37.4	36.9	47.7
	+ CE Difference (target side)	37.3	37.0	47.7
	+ Bilingual CE Difference	37.3	37.0	47.8
	+ **Invitation**	38.6	37.7	46.5
300K	+ CE Difference (source side)	37.3	37.0	47.8
	+ CE Difference (target side)	37.1	37.0	48.0
	+ Bilingual CE Difference	37.3	36.9	47.8
	+**Invitation**	38.9	37.9	46.3

Table 9: Translation results from our domain-adapted SMT systems.

Finally, we also test the adaptation evaluations between the system trained on the small selection of top 300K sentences against the large-scale SMT system trained on C_{mix} when combined with the in-domain trained system. Table 10 presents the results, revealing comparable translation performance, although they are trained on data sets that are significantly different in size.

System	BLEU	METEOR	TER
In-domain + Large data C_{mix}	39.0	38.0	46.3
In-domain + Subset of 300**K**	38.9	37.9	46.3

Table 10: Translation results from our domain-adapted SMT system and the large-scale SMT system. Note that the baseline is slightly better than our domain-adapted SMT system under BLEU and METEOR, however, not statistically significant.

6 Final notes on mix-domain data selection

The specific data selection scenario studied in this paper brings up different aspects that did not receive (sufficient) attention in earlier work on data selection and domain adaptation:

- The mix-domain parallel corpus C_{mix} contains a large variety of domains that overlap and but also differ in lexical choice and translation. This is radically different from the in-/out-domain setting usually assumed in adaptation and constitutes a major challenge for existing selection approaches.
- The way the small in-domain corpus relates to the large mix-domain corpus is also challenging because translation performance often depends on selecting *relevant* sentence pairs, aside from those that are clearly in-domain.
- The lack of out-domain data in a realistic mix-domain scenario, suggests that efforts are needed at finding data that contrasts enough with the in-domain data. In this work we propose an initial training period (burn-in) for isolating pseudo out-domain data. But it might be that relevance-related approaches could also turn out more effective for this.

In our current model we implement the $P(\mathbf{e} \mid D)$ and $P(\mathbf{f} \mid D)$ as language models, inspired by the approaches based on the contrast between the cross-entropies of in- and mix-domain language models (Moore and Lewis, 2010; Axelrod et al., 2011). However, $P(\mathbf{e} \mid D)$ and $P(\mathbf{f} \mid D)$ should work with *relevance models*, i.e., assessing the relevance of sentences to domain D. *Relevance* is a different concept than fluency as embodied by language models, and this aspects demands special attention in future work.[7]

In ongoing large-scale experiments, we now explore the behavior of our Invitation model on a variety of different data settings and compare that to a range of alternative existing approaches. We are also exploring new variations of our Invitation model to find out what the optimal settings might be for different mixes of domains. So far we find that the burn-in and size of pseudo out-domain selection after burn-in can be important in certain situations. We also observe that estimating the suitable size of the selection set is also a topic that demands more attention because the estimate of $P(D_1)$ with the interpretation *percentage of relevant data in* C_{mix} like likely to demand suitable relevance models instead of language models.

We observe that the present Invitation model could be approached from a discriminative perspective, which could be effective for specific data settings. Finally, it is theoretically not clear whether a single approach will be most effective for all practical data scenarios.

7 Conclusions

This work looks at modeling the relevance of sentence pairs from the mix-domain corpus to a task represented by an in-domain sample. In contrast with previous work we cast this as a translation problem with a latent domain variable. Our *Invitation model* based on iterative weighted Invitations using EM, offers a new view on data selection for MT. Our model also offers principled cut-off points for selecting in-domain and other relevant subsets. Experiments on the in-domain task shows our approach outperforms the existing data selection for such a very complex mixture training data.

[7] We thank Amir Kamran for bringing this difference to our attention through ongoing joint experimental work.

The high accuracy in our experiments in this kind of data compared to the baseline suggests that our model might also offer good estimates that can be used for data weighting. In future work we aim to test the Invitation model for instance weighting and explore avenues for using it for selecting and weighting sub-sentential translation pairs (e.g., phrase pairs) that can be used directly for building SMT systems. A further issue is to improve the quality of word alignments induced for mix-domain corpora. We also aim at exploring a discriminative learning approach in conjunction with our model.

Acknowledgements

The first author is supported by the EXPERT (EXPloiting Empirical appRoaches to Translation) Initial Training Network (ITN) of the European Union's Seventh Framework Programme. We thank Translation Automation Society (TAUS.com) for providing us with suitable data for the mix-domain scenario. We also thank Amir Kamran and Bart Mellebeek for help and collaboration on experiments related to data selection and domain adaptation. We thank Miloš Stanojević and three anonymous reviewers for their valuable comments on earlier versions.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 355–362, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pratyush Banerjee, Sudip Kumar Naskar, Johann Roturier, Andy Way, and Josef van Genabith. 2012. Translation quality-based supplementary data selection by incremental update of translation models. In Martin Kay and Christian Boitet, editors, *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India*, pages 149–166. Indian Institute of Technology Bombay.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19:263–311, June.

Boxing Chen, George Foster, and Roland Kuhn. 2013. Adaptation of reordering models for statistical machine translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 938–946, Atlanta, Georgia, June. Association for Computational Linguistics.

Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 427–436, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 176–181, Stroudsburg, PA, USA. Association for Computational Linguistics.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 85–91, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 678–683, Sofia, Bulgaria, August. Association for Computational Linguistics.

Vladimir Eidelman, Jordan Boyd-Graber, and Philip Resnik. 2012. Topic models for dynamic translation model adaptation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 115–119, Stroudsburg, PA, USA. Association for Computational Linguistics.

George Foster and Roland Kuhn. 2007. Mixture-model adaptation for smt. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 128–135, Stroudsburg, PA, USA. Association for Computational Linguistics.

George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 451–459, Stroudsburg, PA, USA. Association for Computational Linguistics.

Guillem Gascó, Martha-Alicia Rocha, Germán Sanchis-Trilles, Jesús Andrés-Ferrer, and Francisco Casacuberta. 2012. Does more data always yield better translations? In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 152–161, Stroudsburg, PA, USA. Association for Computational Linguistics.

Barry Haddow and Philipp Koehn. 2012. Analysing the effect of out-of-domain data on smt systems. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 422–432, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ann Irvine, John Morgan, Marine Carpuat, Hal Daume III, and Dragos Munteanu. 2013. Measuring machine translation errors in new domains. *Transactions of the Association for Computational Linguistics (TACL)*.

Philipp Koehn and Barry Haddow. 2012. Towards effective use of training data in statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, Montreal, Canada, June. Association for Computational Linguistics.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 224–227, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT, AAMT.

Spyros Matsoukas, Antti-Veikko I. Rosti, and Bing Zhang. 2009. Discriminative corpus weight estimation for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 708–717, Singapore, August. Association for Computational Linguistics.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 220–224, Stroudsburg, PA, USA. Association for Computational Linguistics.

Preslav Nakov. 2008. Improving english-spanish statistical machine translation: Experiments in domain adaptation, sentence paraphrasing, tokenization, and recasing. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 147–150, Stroudsburg, PA, USA. Association for Computational Linguistics.

Radford M. Neal and Geoffrey E. Hinton. 1999. A view of the em algorithm that justifies incremental, sparse, and other variants. In Michael I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, Cambridge, MA, USA.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Adam Pauls and Dan Klein. 2011. Faster and smaller n-gram language models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 258–267, Stroudsburg, PA, USA. Association for Computational Linguistics.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1992. *Numerical Recipes in C (2Nd Ed.): The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA.

Rico Sennrich. 2012. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 539–549, Stroudsburg, PA, USA. Association for Computational Linguistics.

Matthew Snover, Bonnie Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Language Family Relationship Preserved in Non-native English

Ryo Nagata
Konan University
8-9-1 Okamoto, Higashinada, Kobe, Hyogo 658-8501, Japan
nagata-coling@hyogo-u.ac.jp

Abstract

Mother tongue interference is the phenomenon where linguistic systems of a mother tongue are transferred to another language. Recently, Nagata and Whittaker (2013) have shown that language family relationship among mother tongues is preserved in English written by Indo-European language speakers because of mother tongue interference. At the same time, their findings further introduce the following two research questions: (1) Does the preservation universally hold in non-native English other than in English of Indo-European language speakers? (2) Is the preservation independent of proficiency in English? In this paper, we address these research questions. We first explore the two research questions empirically by reconstructing language family trees from English texts written by speakers of Asian languages. We then discuss theoretical reasons for the empirical results. We finally introduce another hypothesis called *the existence of a probabilistic module* to explain why the preservation does or does not hold in particular situations.

1 Introduction

Transfer of linguistic systems of a mother tongue to another language, namely *mother tongue interference*, is often observable in the writing of non-native speakers. The reader may be able to determine the mother tongue of the writer of the following sentence from the underlined article error: *The alien wouldn't use my spaceship but the hers*. The answer would probably be French or Spanish; the definite article is allowed to modify possessive pronouns in these languages, and the usage is sometimes negatively transferred to English writing.

Researchers in corpus linguistics including Swan and Smith (2001), Aarts and Granger (1998), and Altenberg and Tapper (1998) have been working on mother tongue interference to reveal overused/underused words, part of speech (POS), or grammatical items. Recently, Nagata and Whittaker (2013) have shown that language family relationship between mother tongues is preserved in English written by Indo-European language speakers; because of the preservation, one can reconstruct a language family tree similar to the canonical Indo-European family tree (Beekes, 2011; Ramat and Ramat, 2006) from their English writings. They have further revealed factors contributing to the preservation of the language family relationship, which they show is useful for related natural language processing (NLP) tasks such as grammatical error detection/correction and native language identification (Wong and Dras, 2009).

At the same time, their findings further introduce the following two research questions: (1) Does the preservation universally hold in non-native English? (2) Is the preservation independent of proficiency in English? The results (Nagata and Whittaker, 2013) for English written by Indo-European language speakers suggest that the answer to question (1) is *yes*. Based on this, we hypothesize that:

Hypothesis I: The preservation of language family relationship universally holds in non-native English.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1940–1949, Dublin, Ireland, August 23-29 2014.

However, one can counter **Hypothesis I**, arguing that the preservation holds only in English written by Indo-European language speakers because Indo-European languages share large part of linguistic properties with English which is a member of the Indo-European languages, which contributes to the preservation. Apparently, this is not the case in languages in other language families. In these languages, other properties than language family relationship may be more dominant. Furthermore, Kachru's Three Circles of English (Kachru, 1992) raises a question. In Kachru's model, world Englishes are classified into the inner, outer, and expanding circles. The inner circle roughly corresponds to the traditional native speakers of English. The outer circle refers to the non-native varieties in the regions where English serves as a useful lingua franca. The expanding circle roughly corresponds to the other non-native speakers of English. Then, it would be difficult to answer question (1) for the outer circle of English (e.g., English in Hong Kong). For example, on one hand, English in Hong Kong is expected to have mother tongue interference from Chinese language. From this point of view, it is expected to have the family relationship with the Sino-Tibetan language family. On the other hand, one can point out that the outer circle of English should be closer to native English than the expanding circle of English (e.g., English in China) is. This implies that English in Hong Kong might have some other relationship with the members in the outer circle. For question (2), the answer is likely *no* considering that theoretically, the higher one's proficiency is, the closer to native English his or her English becomes; it would be difficult to distinguish between native English and English of non-native speakers whose proficiency is very high. With this reason, we hypothesize that:

Hypothesis II: The preservation of language family relationship is dependent on proficiency in English.

In view of this background, we address these research questions in this paper. We first examine the two hypotheses empirically by reconstructing language family trees from English texts written by speakers of Asian languages, including the outer and expanding circles of English. If we can reconstruct language family trees similar to their canonical family trees from these English texts, it will be a good piece of evidence for **Hypothesis I**. Similarly, to examine **Hypothesis II**, we reconstruct a language family tree from the English texts using the information about their proficiency levels. If we cannot reconstruct language family trees similar to the canonical trees, **Hypothesis II** will be accepted. We then explore theoretical reasons for the empirical results. We finally introduce another hypothesis called *the existence of a probabilistic module* to explain why the preservation does or does not hold in particular situations.

The rest of this paper is structured as follows. Sect. 2 introduces the basic approach of this work. Sect. 3 and Sect. 4 examine **Hypothesis I** and **Hypothesis II**, respectively. Sect. 5 describes theoretical reasons for the experimental results.

2 Approach

2.1 Data Set

Through this paper, we use the International Corpus Network of Asian Learners of English (ICNALE) (Ishikawa, 2011) as the target data to examine the two hypotheses. ICNALE consists of English essays of the outer and expanding circles of English in Asia together with those of native speakers of English. Table 1 (a) shows the statistics on ICNALE.

In ICNALE, each essay, except native essays, is annotated with a proficiency level of the writer, ranging from A_2 (lowest) B1_1, B1_2, to B2+ (highest); Table 1 (b) shows the correspondence between these four proficiency levels and TOEIC scores. We use this information to examine **Hypothesis II**.

2.2 Method for Reconstructing Language Family Trees

We employ the method proposed by Nagata and Whittaker (2013) for reconstructing language family trees, which in turn is based on the method proposed by Kita (1999). In this method, each group of the essays in ICNALE is modeled by an n-gram language model. Then, agglomerative hierarchical clustering (Han and Kamber, 2006) is applied to the language models to reconstruct a language family tree. The distance used for clustering is based on a divergence-like distance between two language models that was originally proposed by Juang and Rabiner (1985).

Category	# of essays	# of tokens
Native	400	88,792
Outer Circle		
Hong Kong	200	46,111
Pakistan	400	93,100
Philippines	400	96,586
Singapore	400	96,733
Expanding Circle		
China	800	194,613
Indonesia	400	92,316
Japan	800	176,537
Korea	600	130,626
Thailand	800	176,936
Taiwan	400	89,736

(a) Statistics on ICNALE

Level	A2	B1_1	B1_2	B2+
Score	225-549	550-669	670-784	785+

(b) Correspondence between the Proficiency Levels and TOEIC Score

Table 1: Summary of ICNALE.

To explain the method in more detail, let us define the following symbols used in the method. Let D_i be a set of English texts where i denotes a mother tongue i. Similarly, let M_i be a language model trained using D_i.

To reduce the influences from the topics of the data set, we use an n-gram language model based on a mixture of word and POS tokens. In this language model, content words in n-grams are replaced with their corresponding POS tags. This greatly decreases the influence of the topics of texts. It also decreases the number of parameters in the language model.

To build the language model, the following three preprocessing steps are applied to D_i. First, texts in D_i are split into sentences. Second, each sentence is tokenized, POS-tagged, and mapped entirely to lowercase. For instance, the example sentence in Sect. 1 would give:

> the/DT alien/NN would/MD not/RB use/VB my/PRP$ spaceship/NN but/CC the/DT hers/PRP ./.

Finally, words are replaced with their corresponding POS tags; for the following words, word tokens are used as their corresponding POS tags: coordinating conjunctions, determiners, prepositions, modals, predeterminers, possessives, pronouns, question adverbs. Also, proper nouns are treated as common nouns. At this point, the special POS tags *BOS* and *EOS* are added at the beginning and end of each sentence, respectively. For instance, the above example would result in the following word/POS sequence:

> BOS the NN would RB VB my NN but the hers . EOS.

Note that the content of the original sentence is far from clear while reflecting mother tongue interference, especially in *the hers*.

Now, the language model M_i can be built from D_i. We set $n = 3$ (i.e., trigram language model) and use Kneser-Ney (KN) smoothing (Kneser and Ney, 1995) to estimate its conditional probabilities.

The clustering algorithm used is agglomerative hierarchical clustering with the average linkage method. The distance[1] between two language models is measured as follows. The probability that M_i generates D_i is calculated by $\Pr(D_i|M_i)$. Note that

$$\Pr(D_i|M_i) \approx \Pr(w_{1,i}) \Pr(w_{2,i}|w_{1,i}) \prod_{t=3}^{|D_i|} \Pr(w_{t,i}|w_{t-2,i}, w_{t-1,i}) \quad (1)$$

[1]It is not a distance in a mathematical sense. However, we will use the term *distance* following the convention in the literature.

where $w_{t,i}$ and $|D_i|$ denote the tth token in D_i and the number of tokens in D_i, respectively, since we use the trigram language model. Then, the distance from M_i to M_j is defined by

$$d(M_i \to M_j) \equiv \frac{1}{|D_j|} \log \frac{\Pr(D_j|M_j)}{\Pr(D_j|M_i)}. \quad (2)$$

In other words, the distance is determined based on the ratio of the probabilities that each language model generates the language data. Because $d(M_i \to M_j)$ and $d(M_j \to M_i)$ are not symmetrical, we define the distance between M_i and M_j to be their average:

$$d(M_i, M_j) \equiv \frac{d(M_i \to M_j) + d(M_j \to M_i)}{2}. \quad (3)$$

Equation (3) is used to calculate the distance between two language models for clustering.

To sum up, the procedure of the language family tree construction method is as follows: (i) Preprocess each D_i; (ii) Build M_i from D_i; (iii) Calculate the distances between the language models; (iv) Cluster the language data using the distances; (v) Output the result as a language family tree.

3 Reconstructing Language Family Trees from Asian English

We used the whole ICNALE as the target data. We used a POS-tagger with the Penn Treebank Tagset (Santorini, 1990), which we had specially developed for analyzing non-native English; we trained it on native and non-native corpora we had manually annotated with POS tags, part of which is available to the public as the Konan-JIEM (KJ) learner corpus (Nagata et al., 2011). Then, we generated a cluster tree from the corpus data using the method described in Subsect. 2.2. We used the Kyoto Language Modeling toolkit[2] to build language models from the corpus data. We removed n-grams that appeared less than five times[3] in each subcorpus in the language models.

Fig. 1 shows the resulting cluster tree. The number at each branching node denotes in which step the two clusters were merged.

The cluster tree supports **Hypothesis I** that the preservation of language family relationship universally holds in non-native English. Although the detailed language family relationship is less well-known in these Asian languages than in the Indo-European languages, still the cluster tree shown in Fig. 1 reflects a rational interpretation of their language family relationship. In the cluster tree, Taiwanese and Chinese Englishes are first merged into a cluster. This perfectly agrees with the fact that their mother tongues are primarily Chinese and thus both should belong to the Sino-Tibetan language family. In turn, Japanese and Korean Englishes are merged into a cluster. Their mother tongues are said to be a member of the Altaic language family. Admittedly, it is still controversial whether the two languages belong to the Altaic language family or not. However, the current research often treats them as a member of the Altaic language family (Crystal, 1997). After Japanese and Korean Englishes, Thai and Indonesian Englishes are merged in to a cluster of which mother tongues belong to different language families; the former belong to the Thai language family while the latter mostly belong to the Austronesian language family. Having said that, it has been pointed out that Thai has some language family relationship with the Austronesian language family (Crystal, 1997). All these observations support **Hypothesis I**.

Interestingly, the cluster tree shown in Fig. 1 preserves, together with language family relationship, the three circles of English, namely, the inner (native), outer, and expanding circles of English with an exception of Pakistani English. This can be interpreted as that some other properties are more dominant than language family relationship in the outer circle of English. An implication from this is that we should not treat the outer and expanding circles as a group of non-native speakers of English but separately as different groups in the related NLP tasks such as grammatical error correction. For example, a method performing well on the outer circle of English (e.g., the NUS corpus (Dahlmeier et al., 2013)) does not necessarily perform equally well on the expanding circle of English (e.g., the CLC corpus) and vice versa. Similarly, a model trained on English written by Indo-European language speakers may perform

[2]The Kyoto Language Modeling toolkit: `http://www.phontron.com/kylm/`

[3]We found that the results were not sensitive to the value of frequency cutoff so long as we set it to a small number.

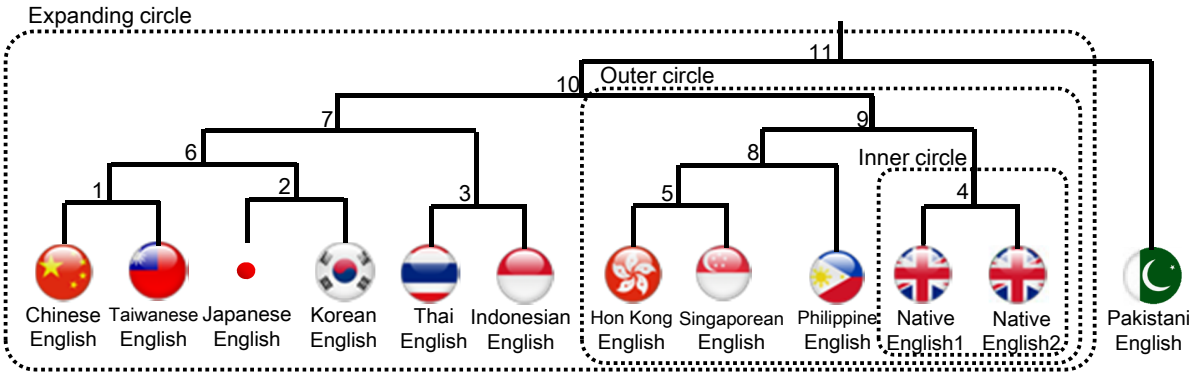


Figure 1: Cluster Tree Reconstructed from Asian Englishes (ICNALE).

better on Chinese English than a model trained on Hon Kong English does. Above all, the subtree for the outer circle of English is a piece of evidence that partly denies **Hypothesis I**.

We further reconstructed a clustering tree from the same data set using 5-gram language models so that the resulting clustering reflects longer-distance syntactic relations. Fig. 2 shows the resulting cluster tree, which reveals that the tree is almost the same as in Fig. 1 with an exception of the Philippine English.

After having observed all these, it would be rational to partly accept **Hypothesis I** and to modify it as follows:

Hypothesis I′: The preservation of language family relationship universally holds in the expanding circle of English.

4 Exploring Correlation between the Preservation and Proficiency

The simplest way to examine **Hypothesis II** would be clustering that uses only either high-proficiency or low-proficiency essays. However, it is not so straightforward because the distribution of each proficiency level varies depending on the English groups. Particularly, some of the 10 non-native Englishes contains no or very few low-proficiency essays[4].

As a simple solution, we first generated a clustering tree from only the high-proficiency essays (B1_2 and B2+) with the same conditions as in Sect. 3. As a more sophisticated solution, we created a new data set from ICNALE so that one of the two Englishes merged into a cluster in Fig. 1 consists of only low-proficiency essays and the other of only high-proficiency essays. For instance, we used only low-proficiency essays (A_2 and B1_1) for Chinese English and only high-proficiency essays (B1_2 and B2+) for Taiwanese English. Then, we generated another cluster tree from the new data set again with the same conditions as in Sect. 3. In addition, as a reference, we generated a cluster tree only using the information about the proficiency levels. In this clustering, we created a vector for each English whose elements and values corresponded to the four proficiency levels and the relative frequencies of the essays falling into the corresponding proficiency level[5]. In this method, we defined the distance for clustering by the Euclidean distance between two vectors.

The idea behind this experiment is as follows. If the preservation is completely independent of proficiency, we will obtain the exact same tree as in Fig.1 both from the only-high-proficiency data set and the high-low proficiency-paired data set. Otherwise, the cluster tree will result in a different form, similar to the one obtained by the vector-based method solely relying on the information about proficiency.

Fig. 1 and Fig. 3 show the cluster trees obtained from the only-high-proficiency data set and the high-low proficiency-paired data set, respectively. In the case of the only-high-proficiency data set, the resulting tree is the exact same as in the one generated from the original data set. Fig. 3 also shows that the cluster tree is very similar to that in Fig. 1. Besides, both tree are far from the cluster tree obtained by the

[4]For instance, Singapore English contains no low-proficiency essays (A2 and B1_1), and Philippine English 26 essays out of 400. See `http://language.sakura.ne.jp/icnale/` for the complete list of the distribution.

[5]We create vectors for the native English essays by setting 1.0 to the element corresponding to B2+ and 0.0 to the others because proficiency levels are not available for the native English essays in ICNALE.

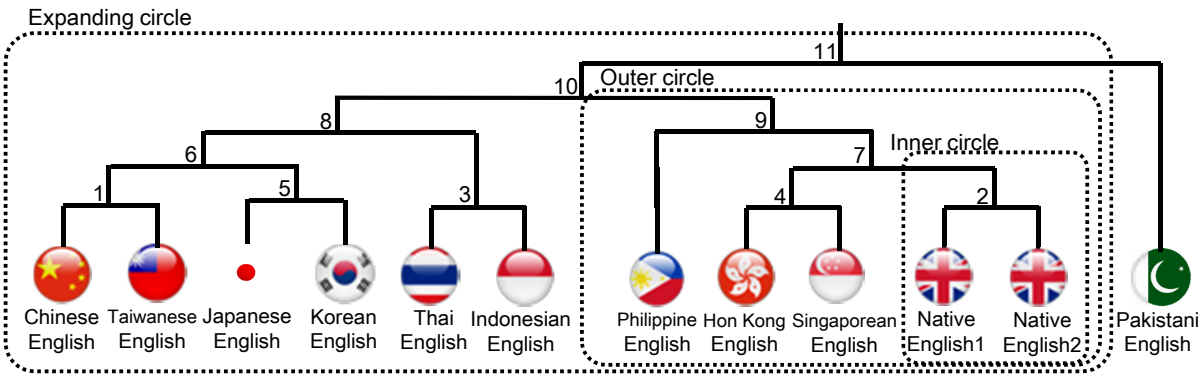


Figure 2: Cluster Tree Reconstructed from Asian Englishes (ICNALE) using 5-gram language models.

vector-based method solely relying on the information about proficiency as shown in Fig. 4. In summary, Fig. 1 to 4 show that the preservation of language family relationship holds in the expanding circle of English regardless of proficiency in English.

These results deny **Hypothesis II** that the preservation of language family relationship is dependent on proficiency in English. Contrary to our expectation, they support[6]:

Hypothesis II′: The preservation of language family relationship is independent of proficiency in English.

5 Discussion

The experiments show that the tree generation method relying on the distributions of word/POS sequences reconstructs from Asian Englishes cluster trees reflecting the family relationship in the Asian languages. These empirical findings, together with those about English written by Indo-European language speakers (Nagata and Whittaker, 2013), support **Hypothesis I′**.

In order to explain theoretically **Hypothesis I′**, we introduce another hypothesis called *the existence of a probabilistic module*, that is, that a probabilistic module that stores the distributional information exists in the human brain. We hypothesize that the probabilistic module consists of sets of probabilities where each set corresponds to a linguistic item which has arbitrariness in its use; the arbitrariness is expressed by means of the probabilities that one of the candidates allowed in the linguistic item is chosen in one's mother tongue. An example of such a linguistic item would be the position of adverb in English where the probabilities in this case represent how likely adverbs appear in certain positions (e.g., the beginning, middle, and end of a sentence). The probabilistic module is equipped with the values of the probabilities which are set according to one's mother tongue. To be precise, in our hypothesis, the probabilities are adapted as follows: (1) proto-languages had developed their values of the probabilities and handed them down to their descendants; (2) over the time, some of the values changed and the others remained unchanged; (3) in turn, the decedent languages handed their values of the probabilities to their descendants with the changes. An example of this would be as follows. The proto-Indo-European language handed down its values of the probabilities to, for example, the Proto-Germanic language and the Proto-Italic language with some changes in the values. Then the Proto-German language handed them down to the Germanic languages such as German and Dutch, again with some changes. So did the Proto-Italic language to the romance languages such as French and Italian. Therefore, the values of the probabilities in German should be more similar to those in Dutch than to those in French or Italian.

With this probabilistic module in the human brain, we can naturally explain the preservation of language family relationship. When non-native speakers use English, the candidates of the arbitrary linguistic items in English are chosen according to the probabilistic module adapted to their mother tongue.

[6]It would be worth while to see if **Hypothesis II** holds in the case of Indo-European Englishes. The difficult part is that there are only a few data annotated with proficiency levels.

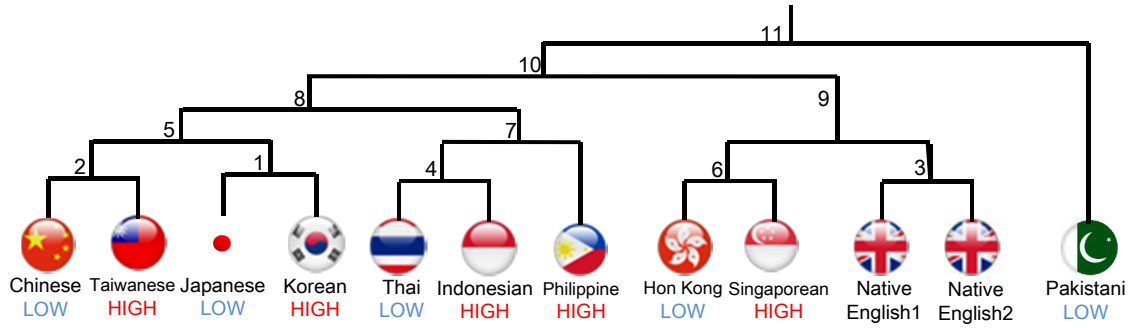


Figure 3: Cluster Tree Reconstructed from the High-low Proficiency-paired ICNALE Data Set (HIGH: high proficiency; LOW: low proficiency).

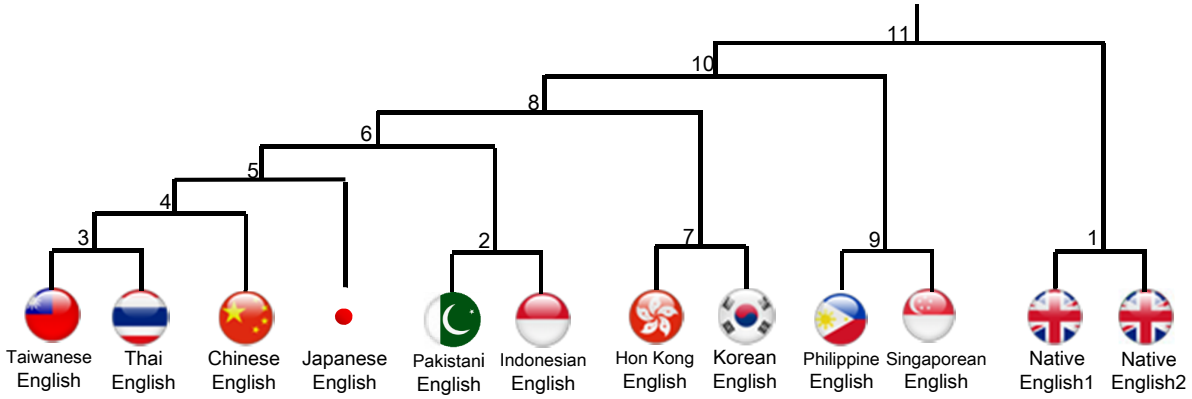


Figure 4: Cluster Tree Generated Based on Only Proficiency Levels.

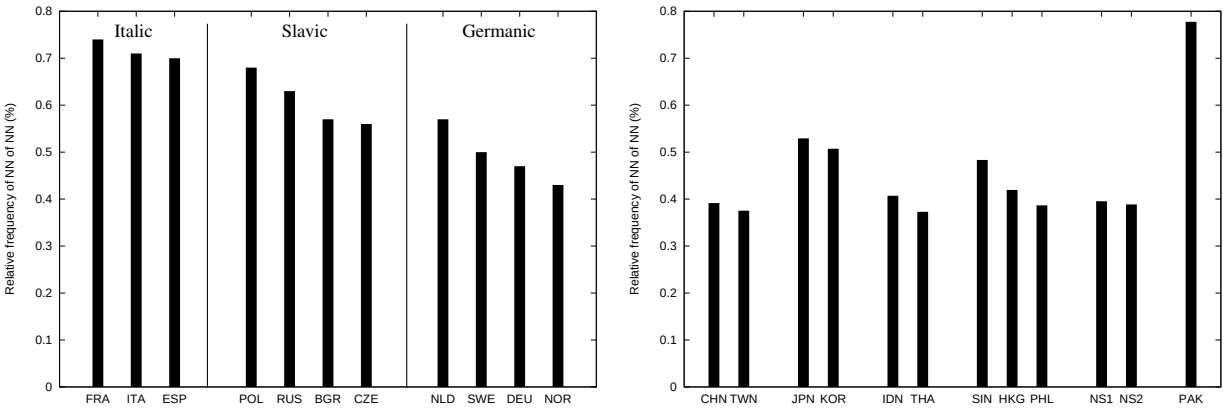
For example, speakers of languages which have a preference for sentence-beginning adverbs would also prefer sentence-beginning adverbs in English writing. Accordingly, the values of the probabilities are implicitly encoded in word/POS sequences such as *BOS RB ,* and *NN RB .*[7] in their English writings, and thus the tree generation method can recognize language family relationship as language family trees via the trigram language model. Provided that the probabilistic module exists in the human brain, this argument can be made about any mother tongues and the target language (not only English) as long as they have arbitrary linguistic items in their language systems, which should be the case in most languages.

This is of course another hypothesis and we need more data and evidence to examine the hypothesis. Nagata and Whittaker (2013) show some evidence that implies the existence of a probabilistic module. They reveal that Englishes written by Indo-European language speakers exhibit certain probabilistic patterns at least in the way of constructing noun phrases (NPs), adverb positions, and article use, reflecting the Italic, Germanic, and Slavic branches of the Indo-European family. Take as an example Fig. 5 (i) which shows frequencies of the trigram *NN of NN* in English written by Indo-European language speakers[8]. Here, note that English language has arbitrariness between the noun-noun compound and the *NN of NN* construction to form an NP (e.g., *education system* vs. *system of education*). Fig. 5 (i) reveals that speakers of the Italic languages (French, Italian, and Spanish) which have a preference for the *NN of NN* construction over the noun-noun compound exhibit relatively high frequencies of the trigram *NN of NN* in English writing. Conversely, speakers of the Germanic languages (Dutch, Swedish, German, and Norwegian) have a preference for the noun-noun compound over the *NN of NN* construction accordingly exhibit lower frequencies of the trigram *NN of NN*. In total, the frequencies roughly classify the 11 Englishes into three groups corresponding to the Italic, Slavic, and Germanic branches of the

[7]These two trigrams roughly correspond to adverbs at the beginning and end of a sentence, respectively.

[8]The ICLE corpus (Granger et al., 2009) was used to calculate the frequencies. The three letters such as FRA in Fig. 5 and Fig. 6 denote the ISO 31661 alpha-3 codes except NS1 (Native Speaker 1) and NS2 (Native Speaker 2).

Indo-European language family.



(i) Indo-European language speakers (ICLE) (ii) Asian language speakers (ICNALE)

Figure 5: Relative Frequency of *NN of NN* in English Texts Written by Non-native Speakers of English.

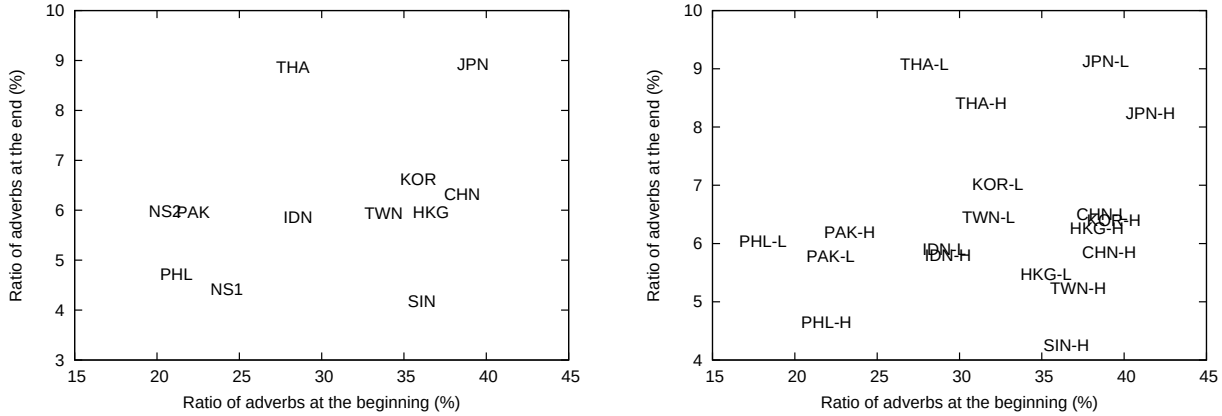
The data of Asian Englishes we used in the experiments exhibit similar tendencies. Fig. 5 (ii) shows frequencies of the trigram *NN of NN* for the Asian Englishes together with the native Englishes (denoted as NS1 and NS2). Fig. 5 (ii) reveals that the pairs of Englishes which share language family relationship each other exhibit similar frequencies of the trigram *NN of NN* as in Fig. 5. Furthermore, Fig. 6 (i) shows a similar tendency in the distribution of adverb positions. The horizontal and vertical axes of Fig. 6 correspond to the ratios of adverbs at the beginning and the end of sentences, respectively, in the Asian and native Englishes. It turns out that the pairs again tend to be located in near positions in the distribution. All of these imply the existence of the probabilistic module.

The probabilistic module also explains why the preservation is independent of proficiency. It is because the values of the probabilities in the probabilistic module will change quite slowly as one improves his or her proficiency. First of all, unlike grammatical errors, explicit feedback such as correction by teachers is not normally given to language learners in the case of the use of the arbitrary linguistic items since any choice among the candidates allowed in a linguistic item is normally correct, as in the adverb positions in English: *Already, I have done it.*, *I have already done it.*, and *I have done it already*, although each of which might have a slightly difference in meaning. Therefore, language learners have little opportunity to adapt the values of the probabilities in their probabilistic module to those in the target language in the first place. Even if feedback is given, it would still be difficult to do so considering that learners scarcely observe the values of the probabilities directly. This is why the values of the probabilities in the probabilistic module tend to be similar within a mother tongue regardless of one's proficiency in English. We can actually see this in Fig. 6 (ii). Fig. 6 (ii) shows the distribution of the ratios of adverbs at the beginning and the end of sentences in the high/low-proficiency essays in ICNALE where X-H and X-L denote high-proficiency and low-proficiency essays of X English, respectively (e.g., THA-H denotes the high-proficiency essays of Thai English). Fig. 6 (ii) reveals that Englishes of the same language speakers tend to remain in near positions regardless of the difference in proficiency.

All these observations would be a good place to start to explore the existence of the probabilistic module. The next step would be to name other arbitrary linguistic items concerning the probabilistic module, one of which for example might be the order of the main and subordinate clauses (e.g., *Because I did it, I did it.* vs *I did it because I did it.*), and then one can reveal their values (probabilities) depending on mother tongues.

6 Conclusions

In this paper, we examined the following two hypotheses: **Hypothesis I**: The preservation of language family relationship universally holds in non-native English; **Hypothesis II**: The preservation of language family relationship is dependent on proficiency in English. The experimental results partly accepted **Hy-**



(i) All levels (ii) High and low levels

Figure 6: Distribution of Adverb Position in Asian Englishes (ICNALE).

pothesis I and revealed that the following hypothesis fitted the data better: **Hypothesis I′**: The preservation of language family relationship universally holds in the expanding circle of English. By contrast, the experimental results denied **Hypothesis II**, supporting the counter hypothesis: **Hypothesis II′**: The preservation of language family relationship is independent of proficiency in English. We then proposed another hypothesis that a probabilistic module exists in the human brain to explain why **Hypothesis I′** and **Hypothesis II′** hold. We further introduced empirical data implying the existence of the probabilistic module.

For future work, we will examine **Hypothesis I′** and **II′** using English texts written by speakers of languages in other families to see if the preservation really universally holds. Also, we will explore the existence of the probabilistic module.

Acknowledgments

The author would like to thank the anonymous reviewers for their thoughtful comments and suggestions on this paper.

References

Jan Aarts and Sylviane Granger, 1998. *Tag sequences in learner corpora: a key to interlanguage grammar and discourse*, pages 132–141. Longman, New York.

Bengt Altenberg and Marie Tapper, 1998. *The use of adverbial connectors in advanced Swedish learners' written English*, pages 80–93. Longman, New York.

Robert S.P. Beekes. 2011. *Comparative Indo-European Linguistics: An Introduction (2nd ed.)*. John Benjamins Publishing Company, Amsterdam.

David Crystal. 1997. *The Cambridge Encyclopedia of Language (2nd ed.)*. Cambridge University Press, Cambridge.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proc. of 8th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31.

Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. 2009. *International Corpus of Learner English v2*. Presses universitaires de Louvain, Louvain.

Jiawei Han and Micheline Kamber. 2006. *Data Mining: Concepts and Techniques (2nd Ed.)*. Morgan Kaufmann Publishers, San Francisco.

Shinichiro Ishikawa, 2011. *A new horizon in learner corpus studies: The aim of the ICNALE project*, pages 3–11. University of Strathclyde Publishing, Glasgow.

Bing-Hwang Juang and Lawrence R. Rabiner. 1985. A probabilistic distance measure for hidden Markov models. *AT&T Technical Journal*, 64(2):391–408.

Braj B. Kachru, 1992. *Teaching World Englishes*, pages 355–365. University of Illinois Press, Urbana and Chicago.

Kenji Kita. 1999. Automatic clustering of languages based on probabilistic models. *Journal of Quantitative Linguistics*, 6(2):167–171.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184.

Ryo Nagata and Edward Whittaker. 2013. Reconstructing an Indo-European family tree from non-native English texts. In *Proc. of 51st Annual Meeting of the Association for Computational Linguistics*, pages 1137–1147.

Ryo Nagata, Edward Whittaker, and Vera Sheinman. 2011. Creating a manually error-tagged and shallow-parsed learner corpus. In *Proc. of 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1210–1219.

Anna Giacalone Ramat and Paolo Ramat. 2006. *The Indo-European Languages*. Routledge, New York.

Beatrice Santorini. 1990. *Part-of-speech tagging guidelines for the Penn Treebank Project*. University of Pennsylvania.

Michael Swan and Bernard Smith. 2001. *Learner English (2nd Ed.)*. Cambridge University Press, Cambridge.

Sze-Meng J. Wong and Mark Dras. 2009. Contrastive analysis and native language identification. In *Proc. Australasian Language Technology Workshop*, pages 53–61.

Why Gender and Age Prediction from Tweets is Hard: Lessons from a Crowdsourcing Experiment

Dong Nguyen[14*] **Dolf Trieschnigg**[14] **A. Seza Doğruöz**[23] **Rilana Gravel**[4]
Mariët Theune[1] **Theo Meder**[4] **Franciska de Jong**[1]
(1) Human Media Interaction, University of Twente, Enschede, The Netherlands
(2) Netherlands Institute for Advanced Studies, Wassenaar, NL
(3) Tilburg School of Humanities, Tilburg University, Tilburg, NL
(4) Meertens Institute, Amsterdam, The Netherlands
*Corresponding author: d.nguyen@utwente.nl

Abstract

There is a growing interest in automatically predicting the gender and age of authors from texts. However, most research so far ignores that language use is related to the social identity of speakers, which may be different from their biological identity. In this paper, we combine insights from sociolinguistics with data collected through an online game, to underline the importance of approaching age and gender as social variables rather than static biological variables. In our game, thousands of players guessed the gender and age of Twitter users based on tweets alone. We show that more than 10% of the Twitter users do not employ language that the crowd associates with their biological sex. It is also shown that older Twitter users are often perceived to be younger. Our findings highlight the limitations of current approaches to gender and age prediction from texts.

1 Introduction

A major thrust of research in sociolinguistics aims to uncover the relationship between social variables such as age and gender, and language use (Holmes and Meyerhoff, 2003; Eckert and McConnell-Ginet, 2013; Eckert, 1997; Wagner, 2012). In line with scholars from a variety of disciplines, including the social sciences and philosophy, sociolinguists consider age and gender as social and fluid variables (Eckert, 2012). Gender and age are shaped depending on the societal context, the culture of the speakers involved in a conversation, the individual experiences and the multitude of social roles: a female teenager might also be a high school student, a piano player, a swimmer, etc. (Eckert, 2008).

Speakers use language as a resource to construct their identity (Bucholtz and Hall, 2005). For example, a person's gender identity is constructed through language by using linguistic features associated with male or female speech. These features gain social meaning in a cultural and societal context. On Twitter, users construct their identity through interacting with other users (Marwick and boyd, 2011). Depending on the context, they may emphasize specific aspects of their identity, which leads to linguistic variation both within and between speakers. We illustrate this with the following three tweets:

Tweet 1: *I'm walking on sunshine <3 #and don't you feel good*
Tweet 2: *lalaloveya <3*
Tweet 3: *@USER loveyou ;D*

In these tweets, we find linguistic markers usually associated with females (e.g. a heart represented as <3). Indeed, 77% of the 181 players guessed that a female wrote these tweets in our online game. However, this is a 16-year old biological male, whose Twitter account reveals that he mostly engages with female friends. Therefore, he may have accommodated his style to them (Danescu-Niculescu-Mizil et al., 2011) and as a result he employs linguistic markers associated with the opposite biological sex.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1950–1961, Dublin, Ireland, August 23-29 2014.

Most of the NLP research focusing on predicting gender and age has approached these variables as *biological* and *static*, rather than *social* and *fluid*. For example, current approaches use supervised machine learning models trained on tweets from males and females. However, the resulting stereotypical models are ineffective for Twitter users who tweet differently from what is to be expected from their biological sex.

As explained above, language use is based on social gender and age identity, and not on biological sex and chronological age. In other words, treating gender and age as fixed biological variables in analyzing language use is too simplistic. By comparing the *biological sex and chronological age* of Twitter users with how they are perceived by the crowd (as an indication of socially constructed identities), we shed light on the *difficulty* of predicting gender and age from language use and draw attention to the *inherent limitations* of current approaches.

As has been demonstrated in several studies, the crowd can be used for experimentation (e.g., Munro et al. (2010)). Our study illustrates the value of the crowd for the study of human behavior, in particular for the experimental study of the social dimension of language use. To collect data, we created an online game (an example of *gamification* (Deterding et al., 2011)) in which thousands of players (the crowd) guessed the biological sex and age of Twitter users based on only the users' tweets. While variance between annotators has traditionally been treated as noise, more recently variation is being treated as a *signal* rather than noise (Aroyo and Welty, 2013). For example, Makatchev and Simmons (2011) analyze how English utterances are perceived differently across language communities.

This paper follows this trend, treating variation as meaningful information. We assume that the crowd's perception (based on the distribution of the players' guesses) is an indication of to what extent Twitter users emphasize their gender and age identity in their tweets. For example, when a large proportion of the players guess the same gender for a particular user, the user is assumed to employ linguistic markers that the crowd associates with gender-specific speech (e.g. iconic hearts used by females).

Our contributions are as follows:

- We demonstrate the use of gamification to study sociolinguistic research problems (Section 3).
- We study the difficulty of predicting an author's gender (Section 4) and age (Section 5) from text alone by analyzing prediction performance by the crowd. We relate our results to sociolinguistic theories and show that approaching gender and age as fixed biological variables is too simplistic.
- Based on our findings, we reflect on current approaches to predicting age and gender from text, and draw attention to the limitations of these approaches (Section 6).

2 Related Work

Gender Within sociolinguistics, studies on gender and language have a long history (Eckert and McConnell-Ginet, 2013). More recently, the NLP community has become increasingly interested in this topic. Most of the work aims at predicting the gender of authors based on their text, thereby focusing more on prediction performance than sociolinguistic insights.

A variety of datasets have been used, including Twitter (Rao et al., 2010; Bamman et al., 2014; Fink et al., 2012; Bergsma and Van Durme, 2013; Burger et al., 2011), blogs (Mukherjee and Liu, 2010; Schler et al., 2005), telephone conversations (Garera and Yarowsky, 2009), YouTube (Filippova, 2012) and chats in social networks (Peersman et al., 2011). Females tend to use more pronouns, emoticons, emotion words, and blog words (*lol, omg, etc.*), while males tend to use more numbers, technology words, and links (Rao et al., 2010; Bamman et al., 2014; Nguyen et al., 2013). These differences have also been exploited to improve sentiment classification (Volkova et al., 2013) and cyberbullying detection (Dadvar et al., 2012).

To the best of our knowledge, the study by Bamman et al. (2014) is the only computational study that approaches gender as a social variable. By clustering Twitter users based on their tweets, they show that multiple gendered styles exist. Unlike their study, we use the crowd and focus on implications for gender and age prediction.

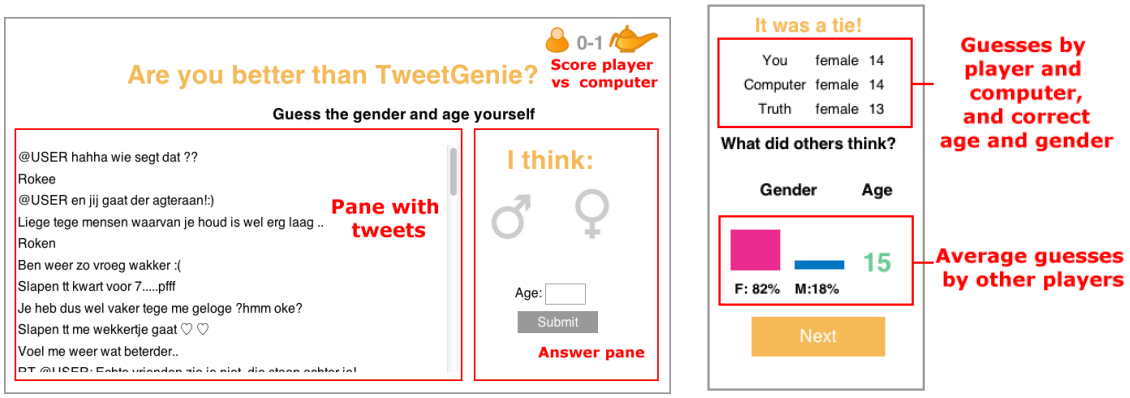


Figure 1: Screenshot of the game. Text is translated into English (originally in Dutch). Left shows the interface when the user needs to make a guess. Right shows the feedback interface.

Age Eckert (1997) makes a distinction between chronological (number of years since birth), biological (physical maturity), and social age (based on life events). Most of the studies on language and age focus on chronological age. However, speakers with the same chronological age can have very different positions in society, resulting in variation in language use. Computational studies on language use and age usually focus on automatic (chronological) age prediction. This has typically been modeled as a classification problem, although this approach often suffers from ad hoc and dataset dependent age boundaries (Rosenthal and McKeown, 2011). In contrast, recent works also explored predicting age as a continuous variable and predicting lifestages (Nguyen et al., 2013; Nguyen et al., 2011) .

Similar to studies on gender prediction, a variety of resources have been used for age prediction, including Twitter (Rao et al., 2010; Nguyen et al., 2013), blogs (Rosenthal and McKeown, 2011; Goswami et al., 2009), chats in social networks (Peersman et al., 2011) and telephone conversations (Garera and Yarowsky, 2009). Younger people use more alphabetical lengthening, more capitalization of words, shorter words and sentences, more self-references, more slang words, and more Internet acronyms (Rosenthal and McKeown, 2011; Nguyen et al., 2013; Rao et al., 2010; Goswami et al., 2009; Pennebaker and Stone, 2003; Barbieri, 2008).

3 Data

To study how people perceive the gender and age identity of Twitter users based on their tweets, we created an online game. Players were asked to guess the gender and age of Twitter users from tweets. The game was part of a website (TweetGenie, www.tweetgenie.nl) that also hosted an automatic system that predicts the gender and age of Twitter users based on their tweets (Nguyen et al., 2014). To attract players, a link to the game was displayed on the page with the results of the automatic prediction, and visitors were challenged to test if they were better than the automatic system (TweetGenie).

3.1 Twitter Data

We sampled Dutch Twitter users in the fall of 2012. We employed external annotators to annotate the biological sex and chronological age (in years) using all information available through tweets, the Twitter profile and external social media profiles such as Facebook and Linkedin. In total over 3000 Twitter users were annotated. For more details regarding the collection of the dataset we refer to Nguyen et al. (2013).

We divided the data into train and test sets. 200 Twitter users were randomly selected from the test set to be included in the online game (statistics are shown in Table 1). Named entities were manually anonymized to conceal the user's identity. Names in tweets were replaced by 'similar' names (e.g. a first name common in a certain region in the Netherlands was replaced with another common name in that region). This was done without knowing the actual gender and age of the Twitter users. Links were replaced with a general [LINK] token and user mentions with @USER.

Gender and age	F, <20	M, <20	F, [20-40)	M, [20-40)	F, ≥40	M, ≥40
Frequency	61	60	24	23	17	15

Table 1: Statistics Twitter users in our game

3.2 Online Game

Game Setup The interface of the game is shown in Figure 1. Players guessed the biological sex (male or female) and age (years) of a Twitter user based on only the tweets. For each user, {20, 25, 30, 35, 40} tweets were randomly selected. For a particular Twitter user, the same tweets were displayed to all players. Twitter users were randomly selected to be displayed to the players.

To include an entertainment element, players received feedback after each guess. They were shown the correct age and gender, the age and gender guessed by the computer, and the average guessed age and gender distribution by the other players. In addition, a score was shown of the player versus the computer.

Collection In May 2013, the game was launched. Media attention resulted in a large number of visitors (Nguyen et al., 2014). We use the data collected from May 13, 2013 to August 21, 2013, resulting in a total of 46,903 manual guesses. Players tweeted positively about the game, such as '*@USER Do you know what is really addictive? "Are you better than Tweetgenie" ...*' and '*@USER Their game is quite fun!*' (tweets translated to English).

We filter sessions that do not seem to contain genuine guesses: when the entered age is 80 years or above, or 8 or below. These thresholds were based on manual inspection, and chosen because it is unlikely that the shown tweets are from users of such ages. For each guess, we registered a session ID and an IP address. A new session started after 2 hours of inactivity. To study player performance more robustly, we excluded multiple sessions of the same player. After three or more guesses had been made in a session, all next sessions from the same IP address were discarded.

Statistics Statistics of the data are shown in Table 2. Figure 2 shows the distribution of the number of guesses per session. The longest sessions consisted of 18 guesses. Some of our analyses require multiple guesses per player. In that case, we only include players having made at least 7 guesses.

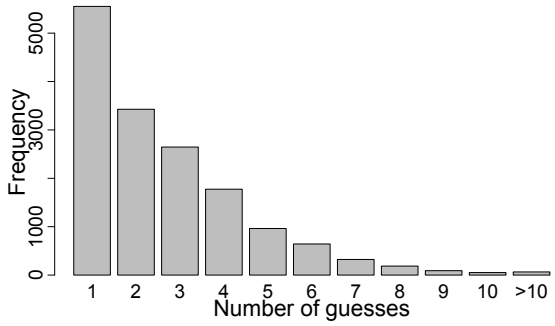


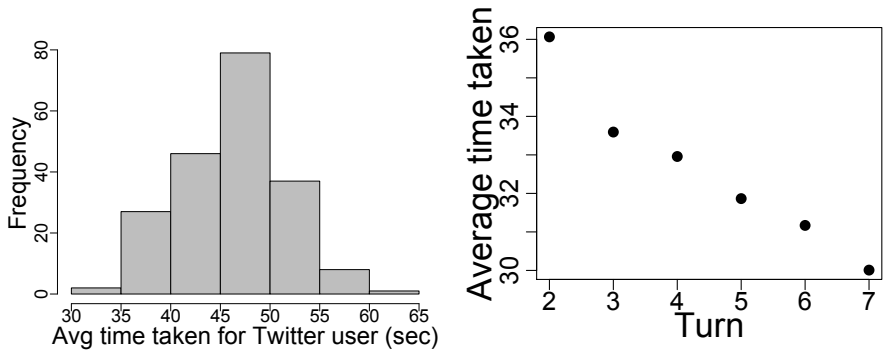
Figure 2: Number of guesses per session

# guesses	41,989
# sessions	15,724
Avg. time (sec) per guess	46
Avg. # guesses / session	2.67

Table 2: Statistics online game (after cleaning)

We calculate the time taken for a guess by taking the time difference between two guesses (therefore, no time for the first guess in each session could be measured). For each Twitter user, we calculate the average time that was taken to guess the gender and age of the user. (Figure 3a). There is a significant correlation (Pearson's r = 0.291, $p < 0.001$) between the average time the players took to evaluate the tweets of a Twitter user and the number of displayed tweets.

There is also a significant correlation between the average time taken for a user and the entropy over gender guesses (Pearson's r = 0.410, $p < 0.001$), and the average time taken for a user and the standard deviation of the age guesses (Pearson's r = 0.408, $p < 0.001$). Thus, on average, players spent more time on Twitter users for whom it was more difficult to estimate gender and age.



(a) Average time taken for Twitter users (b) Average time taken per turn

Figure 3: Time taken in game

We observe that as the game progresses, players tend to take less time to make a guess. This is shown in Figure 3b, which shows the average time taken for a turn (restricted to players with at least 7 guesses). There was no significant correlation between time spent on a guess and the performance of players and we did not find trends of performance increase or decrease as players progressed in the game.

3.3 Automatic Prediction

Besides studying human performance, we also compare the predictions of humans with those of an automatic system. We split the data into train and test sets using the same splits as used by Nguyen et al. (2013). We train a logistic regression model to predict gender (male or female), and a linear regression model to predict the age (in years) of a person.

More specifically, given an input vector $\mathbf{x} \in \mathbb{R}^m$, $x_1, \ldots, x_m$ represent features. In the case of gender classification (e.g. $y \in \{-1, 1\}$), the model estimates a conditional distribution $P(y|\mathbf{x}, \beta) = 1/(1 + exp(-y(\beta_0 + \mathbf{x}^\top \boldsymbol{\beta})))$, where β_0 and $\boldsymbol{\beta}$ are the parameters to estimate. Age is treated as a regression problem, and we find a prediction $\hat{y} \in \mathbb{R}$ for the exact age of a person $y \in \mathbb{R}$ using a linear regression model: $\hat{y} = \beta_0 + \mathbf{x}^\top \boldsymbol{\beta}$. We use Ridge (also called L_2) regularization to prevent overfitting.

We make use of the liblinear (Fan et al., 2008) and scikit-learn (Pedregosa et al., 2011) libraries. We only use unigram features, since they have proven to be very effective for gender (Bamman et al., 2014; Peersman et al., 2011) and age (Nguyen et al., 2013) prediction. Parameters were tuned using cross-validation on the training set.

4 Gender

Most of the computational work on language and gender focuses on gender classification, treating gender as fixed and classifying speakers into females and males. However, this assumes that gender *is* fixed and is something people have, instead of something people *do* (Butler, 1990).

In this section, we first analyze the *task difficulty* by studying crowd performance on inferring gender from tweets. We observe a relatively large group of Twitter users who employ language that the crowd associates with the *opposite* biological sex. This, then, raises questions about the upper bound that a prediction system based on only text can achieve.

Next, we place Twitter users on a *gender continuum* based on the guesses of the players and show that treating gender as a binary variable is too simplistic. While historically gender has been treated as binary, researchers in fields such as sociology (Lorber, 1996) and sociolinguistics (Holmes and Meyerhoff, 2003; Bergvall et al., 1996) find this view too limited. Instead, we assume the simplest extension beyond a binary variable: a one-dimensional gender continuum (or scale) (Bergvall et al., 1996). For example, Bergvall (1999) talks about a '*continuum of humans' gendered practices*'. While these previous studies were based on qualitative analyses, we take a quantitative approach using the crowd.

4.1 Task Difficulty

Majority vote We study crowd performance using a system based on the *majority* of the players' guesses. Majority voting has proven to be a strong baseline to aggregate votes (e.g. in crowdsourcing systems (Snow et al., 2008; Le et al., 2010)). On average, we have 210 guesses per Twitter user, providing substantial evidence per Twitter user. A system based on majority votes achieves an accuracy of 84% (Table 3a shows a confusion matrix). Table 3b shows a confusion matrix of the majority predictions versus the automatic system. We find that the biological sex was predicted incorrectly by both the majority vote system and the automatic system for 21 out of the 200 Twitter users (10.5%, not in Table).

Automatic classification systems on English tweets achieve similar performances as our majority vote system (e.g. Bergsma and Van Durme (2013) report an accuracy of 87%, Bamman et al. (2014) 88%). More significantly, the results suggest that 10.5% (automatic + majority) to 16% (majority) of the Dutch Twitter users do not employ language that the crowd associates with their biological sex. As said, this raises the question of whether we can expect much higher performances by computational systems based on only language use.

		Biological sex	
		Male	Female
Crowd	Male	82	16
	Female	16	86

(a) Crowd (majority)

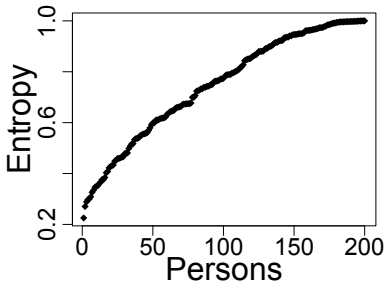
		Crowd	
		Male	Female
Automatic	Male	68	22
	Female	30	80

(b) Automatic vs crowd

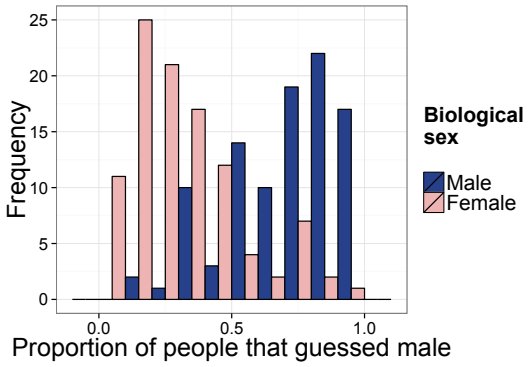
Table 3: Confusion matrices crowd prediction

Individual players versus an automatic system When considering players with 7 or more guesses, the average accuracy for a player is 0.71. Our automatic system achieves an accuracy of 0.69. The small number of tweets per Twitter user in our data (20-40) makes it more difficult to automatically predict gender.

Entropy We characterize the difficulty of inferring a user's gender by calculating the entropy for each Twitter user based on the gender guesses (Figure 4a). We find that the difficulty varies widely across users, and that there are no distinct groups of 'easy' and 'difficult' users. However, we do observe an interaction effect between the entropy of the gender guesses and the ages of the Twitter users. At an aggregate level, we find no significant trend. Analyzing females and males separately, we observe a significant trend with females (Pearson's r = 0.270, $p < 0.01$), suggesting that older female Twitter users tend to emphasize other aspects than their gender in tweets (as perceived by the crowd).



(a) Entropy over gender guesses



(b) A histogram of all Twitter users and the proportion of players who guessed the users were male. For example, there are 25 female users for which 10 - 20% of the players guessed they were male.

Figure 4: Gender prediction

4.2 Binarizing Gender, a Good Approach?

Using data collected through the online game we *quantitatively* put speakers on a gender continuum based on how their tweets are perceived by the crowd. For each Twitter user, we calculate the proportion of players who guessed the users were male and female. A plot is displayed in Figure 4b. We can make the following observations:

First, the guesses by the players are based on their expectations about what kind of behaviour and language is used by males and females. The plot shows that for some users, almost all players guessed the same gender, indicating that these expectations are quite strong and that there are stylistic markers and topics that the crowd strongly associates with males or females.

Second, if treating gender as a binary variable is reasonable, we would expect to see two distinct groups. However, we observe quite an overlap between the biological males and females. There are 1) users who conform to what is expected based on their biological sex, 2) users who deviate from what is expected, 3) users whose tweets do not emphasize a gender identity or whose tweets have large variation using language associated with both genders. We investigated whether this is related to their use of Twitter (professional, personal, or both), but the number of Twitter users in our dataset who used Twitter professionally was small and not sufficient to draw conclusions.

We now illustrate our findings using examples. The first example is a 15-year old biological female for who the crowd guessed most strongly that she is female (96% of n=220). Three tweets from her are shown below. She uses language typically associated with females, talking about spending time with her girlfriends and the use of stylistic markers such as hearts and alphabetical lengthening. Thus, she conforms strongly to what the crowd expects from her biological sex.

Tweet 4: *Gezellig bij Emily en Charlotte.*
Translation: *Having fun with Emily and Charlotte.*

Tweet 5: *Hiiiiii schatjesss!*
Translation: *Hiiiiii cutiesss!*

Tweet 6: ♥ *@USER*

Below are two tweets from a 40 year old biological female who does not employ linguistic markers strongly associated with males or females. Therefore, only 46% of the crowd (n=200) was able to guess that she is female.

Tweet 7: *Ik viel op mijn bek. En het kabinet ook. Geinig toch? #Catshuis*
Translation: *I went flat on my face. And the cabinet as well. Funny right? #Catshuis*

Tweet 8: *Jeemig. Ik kan het bijna niet volgen allemaal.*
Translation: *Jeez. I almost can't follow it all.*

Twitter users vary in how much they emphasize their gender in their tweets. As a result, the difficulty of inferring gender from tweets varies across persons, and treating gender as a binary variable ignores much of the interesting variation within and between persons.

Automatic system We now analyze whether an automatic system is capable of capturing the position of Twitter users on the gender continuum (as perceived by the crowd). We calculate the correlation between the proportion of male guesses (i.e. the position on the gender continuum) and the scores of the logistic regression classifier: $\beta_0 + \mathbf{x}^\top \boldsymbol{\beta}$. While the training data was binary (users were labeled as male or female), a reasonable Spearman correlation of $\rho = 0.584$ ($p < 0.001$) was obtained between the classifier score and the score based on the crowd's perception. We did not observe a significant relation between the score of the classifier (corresponding to the confidence of the gender prediction) and age.

5 Age

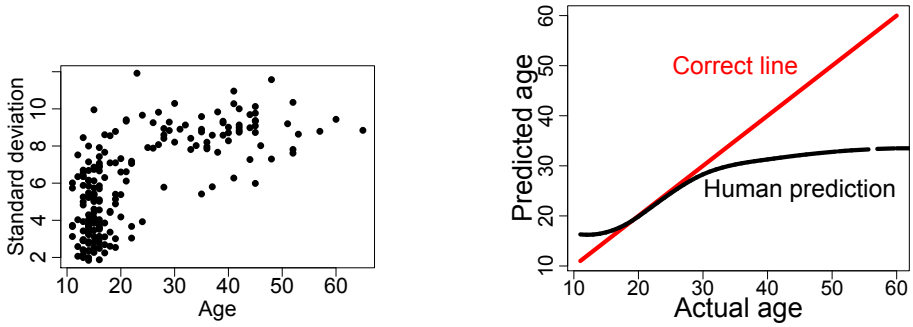
We start with an analysis of task difficulty, by studying crowd performance on inferring age from tweets. Next, we show that it is particularly hard to accurately infer the chronological age of older Twitter users from tweets.

5.1 Task Difficulty

The crowd's average guesses As with a system based on majority vote for gender prediction, we test the performance of a system that predicts the ages of Twitter users based on the average of all guesses. We find that such a system achieves a Mean Absolute Error (MAE) of 4.844 years and a Pearson's correlation of 0.866. Although the correlation is high, the absolute errors are quite large. We find that the crowd has difficulty predicting the ages of older Twitter users. There is a positive correlation (Pearson's $\rho = 0.789$) between the absolute errors and the actual age of Twitter users. There is a negative correlation between the errors (predicted - actual age) and the actual age of Twitter users (Pearson's ρ = -0.872).

We calculate the standard deviation over all the age guesses for a user (Figure 5a) to measure the difficulty of inferring a user's age. There is a positive correlation between age and standard deviation of the guesses ($\rho = 0.691$), which indicates that players have more difficulty in guessing the ages of older Twitter users.

Individual players versus an Automatic System To estimate the performance of individual players, we restrict our attention to players with at least 7 guesses. We find that individual players are, on average, 5.754 years off. A linear regression system achieves a MAE of 6.149 years and a Pearson correlation of 0.812. The small number of tweets in our data (20-40) increases the difficulty of the task for automatic systems.



(a) Standard deviation and actual age

(b) Average age prediction by humans.

Figure 5: Age prediction

5.2 Inferring the Age of Older Twitter Users

Figure 5b shows the average player predictions with the actual age of the Twitter users. The red line is the 'perfect' line, i.e. the line when the predictions would match the exact age. Black represents a fitted LOESS curve (Cleveland et al., 1992) based on the human predictions. We find that the players tend to overpredict the age of younger Twitter users, but even more strikingly, on average they consistently underpredict the age of older Twitter users. The prediction errors already start at the end of the 20s, and the gap between actual and predicted age increases with age.

This could be explained by sociolinguistic studies that have found that people between 30 and 55 years use standard forms the most, because they experience the maximum societal pressure in the workplace to conform (Holmes, 2013). On Twitter, this has been observed as well: Nguyen et al. (2013) found fewer linguistic differences between older age groups than between younger age groups. This makes it difficult for the crowd to accurately estimate the ages of older Twitter users. Younger people and retired people use more non-standard forms (Holmes, 2013). Unfortunately, our dataset does not contain enough retired users to analyze whether this trend is also present on Twitter.

6 Discussion

We now discuss the implications of our findings for research on automatically predicting the gender and age of authors from their texts.

Age and gender as *social* variables Most computational research has treated gender and age as fixed, biological variables. The dominant approach is to use supervised machine learning methods to generalize across a large number of examples (e.g. texts written by females and males). While the learned models so far are effective at predicting age and gender of *most* people, they learn stereotypical behaviour and therefore provide a simplistic view.

First, by using the crowd we have shown that Twitter users emphasize their gender and age in varying degrees and in different ways, so that for example, treating gender as a binary variable is too simplistic (Butler, 1990; Eckert and McConnell-Ginet, 2013). Many users do not employ the stereotypical language associated with their biological sex, making models that take a static view of gender ineffective for such users. More detailed error analyses of the prediction systems will increase understanding of the reasons for incorrect predictions, and shed light on the relation between language use and social variables.

Second, models that assume static variables will not be able to model the interesting variation (Eisenstein, 2013). Models that build on recent developments in sociolinguistics will be more meaningful and will also have the potential to contribute to new sociolinguistic insights. For example, modeling what influences speakers to show more or less of their identity through language, or jointly modeling variation between and within speakers, are in our opinion interesting research directions. The ever increasing amounts of social media data offer opportunities to explore these research directions.

Sampling We have shown that the difficulty of tasks such as gender and age prediction varies across persons. Therefore, creating datasets for such tasks requires maximum attention. For example, when a dataset is biased towards people who show a strong gender identity (e.g. by sampling followers of accounts highly associated with males or females, such as sororities (Rao et al., 2010)), the results obtained on such a set may not be representative of a more random set (as observed when classifying political affiliation (Cohen and Ruths, 2013)).

Task difficulty Our study also raises the question of what level of performance can be obtained for tasks such as predicting gender and age from only language use. Since we often form an impression based on someone's writing, crowd performance is a good indicator of the task difficulty. While the crowd performance does not need to be the upper bound, it does indicate that it is difficult to predict gender and age of a large number of Twitter users.

When taking the majority label, only 84% of the users were correctly classified according to their biological sex. This suggests that about 16% of the Dutch Twitter users do not use language that the *crowd* associates with their biological sex.

We also found that it is hard to accurately estimate the ages of older Twitter users, and we related this to sociolinguistics studies who found less linguistic differences in older age groups due to societal pressure in the workplace.

Limitations A limitation of our work is that we focused on language variation *between* persons, and not on variation *within* persons. However, speakers vary their language depending on the context and their conversation partners (e.g. accommodation effects were found in social media (Danescu-Niculescu-Mizil et al., 2011)). For example, we assigned Twitter users an overall 'score' by placing them on a gender continuum, ignoring the variation we find within users.

Crowdsourcing as a tool to understand NLP tasks Most research on crowdsourcing within the NLP community has focused on how the crowd can be used to obtain fast and large amounts of annotations. This study is an example of how the crowd can be used to obtain a deeper understanding of an NLP task. We expect that other tasks where disagreement between annotators is meaningful (i.e. it is not only due to noise), could potentially benefit from crowdsourcing experiments as well.

7 Conclusion

In this paper, we demonstrated the successful use of the crowd to study the relation between language use and social variables. In particular, we took a closer look at inferring gender and age from language using data collected through an online game. We showed that treating gender and age as fixed variables ignores the variety of ways people construct their identity through language.

Approaching age and gender as *social* variables will allow for richer analyses and more robust systems. It has implications ranging from how datasets are created to how results are interpreted. We expect that our findings also apply to other social variables, such as ethnicity and status. Instead of only focusing on performance improvement, we encourage NLP researchers to also focus on what we can *learn* about the relation between language use and social variables using computational methods.

Acknowledgements

This research was supported by the Royal Netherlands Academy of Arts and Sciences (KNAW) and the Netherlands Organization for Scientific Research (NWO), grants IB/MP/2955 (TINPOT) and 640.005.002 (FACT). The third author is supported through the Digital Humanities research grant by Tilburg University and a NIAS research fellowship. The authors would like to thank the players of the TweetGenie game.

References

Lora Aroyo and Chris Welty. 2013. Crowd truth: Harnessing disagreement in crowdsourcing a relation extraction gold standard. In *Proceedings of WebSci'13*.

David Bamman, Jacob Eisenstein, and Tyler Schnoebelen. 2014. Gender identity and lexical variation in social media. *Journal of Sociolinguistics*, 18(2):135–160.

Federica Barbieri. 2008. Patterns of age-based linguistic variation in American English. *Journal of Sociolinguistics*, 12(1):58–88.

Shane Bergsma and Benjamin Van Durme. 2013. Using conceptual class attributes to characterize social media users. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 710–720.

Victoria L. Bergvall, Janet M. Bing, and Alice F. Freed. 1996. *Rethinking Language and Gender Research: Theory and Practice*. Routledge.

Victoria L. Bergvall. 1999. Toward a comprehensive theory of language and gender. *Language in society*, 28(02):273–293.

Mary Bucholtz and Kira Hall. 2005. Identity and interaction: A sociocultural linguistic approach. *Discourse studies*, 7(4-5):585–614.

John D. Burger, John Henderson, George Kim, and Guido Zarrella. 2011. Discriminating gender on Twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1301–1309.

Judith Butler. 1990. *Gender Trouble: Feminism and the Subversion of Identity*. Routledge.

William S. Cleveland, Eric Grosse, and William M. Shyu. 1992. Local regression models. *Statistical models in S*, pages 309–376.

Raviv Cohen and Derek Ruths. 2013. Classifying political orientation on Twitter: It's not easy! In *Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media*, pages 91–99.

Maral Dadvar, Franciska de Jong, Roeland Ordelman, and Dolf Trieschnigg. 2012. Improved cyberbullying detection using gender information. In *Proceedings of the Twelfth Dutch-Belgian Information Retrieval Workshop (DIR 2012)*, pages 23–25.

Cristian Danescu-Niculescu-Mizil, Michael Gamon, and Susan Dumais. 2011. Mark my words!: linguistic style accommodation in social media. In *Proceedings of the 20th international conference on World Wide Web*, pages 745–754.

Sebastian Deterding, Dan Dixon, Rilla Khaled, and Lennart Nacke. 2011. From game design elements to gamefulness: Defining "gamification". In *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, pages 9–15.

Penelope Eckert and Sally McConnell-Ginet. 2013. *Language and gender*. Cambridge University Press.

Penelope Eckert. 1997. Age as a sociolinguistic variable. *The handbook of sociolinguistics*, pages 151–167.

Penelope Eckert. 2008. Variation and the indexical field. *Journal of Sociolinguistics*, 12(4):453–476.

Penelope Eckert. 2012. Three waves of variation study: the emergence of meaning in the study of sociolinguistic variation. *Annual Review of Anthropology*, 41:87–100.

Jacob Eisenstein. 2013. What to do about bad language on the internet. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 359–369.

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

Katja Filippova. 2012. User demographics and language in an implicit social network. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1478–1488.

Clayton Fink, Jonathon Kopecky, and Maksym Morawski. 2012. Inferring gender from the content of tweets: A region specific example. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*.

Nikesh Garera and David Yarowsky. 2009. Modeling latent biographic attributes in conversational genres. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 710–718.

Sumit Goswami, Sudeshna Sarkar, and Mayur Rustagi. 2009. Stylometric analysis of bloggers' age and gender. In *Proceedings of the Third International ICWSM Conference*, pages 214–217.

Janet Holmes and Miriam Meyerhoff. 2003. *The handbook of language and gender*. Wiley-Blackwell.

Janet Holmes. 2013. *An introduction to sociolinguistics*. Routledge.

John Le, Andy Edmonds, Vaughn Hester, and Lukas Biewald. 2010. Ensuring quality in crowdsourced search relevance evaluation: The effects of training question distribution. In *Proceedings of the SIGIR 2010 Workshop on Crowdsourcing for Search Evaluation (CSE 2010)*, pages 21–26.

Judith Lorber. 1996. Beyond the binaries: Depolarizing the categories of sex, sexuality, and gender*. *Sociological Inquiry*, 66(2):143–160.

Maxim Makatchev and Reid Simmons. 2011. Perception of personality and naturalness through dialogues by native speakers of American English and Arabic. In *Proceedings of the SIGDIAL 2011: the 12th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 286–293.

Alice E. Marwick and danah boyd. 2011. I tweet honestly, I tweet passionately: Twitter users, context collapse, and the imagined audience. *New Media & Society*, 13(1):114–133.

Arjun Mukherjee and Bing Liu. 2010. Improving gender classification of blog authors. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 207–217.

Robert Munro, Steven Bethard, Victor Kuperman, Vicky Tzuyin Lai, Robin Melnick, Christopher Potts, Tyler Schnoebelen, and Harry Tily. 2010. Crowdsourcing and language studies: the new generation of linguistic data. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 122–130.

Dong Nguyen, Noah A Smith, and Carolyn P. Rosé. 2011. Author age prediction from text using linear regression. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 115–123.

Dong Nguyen, Rilana Gravel, Dolf Trieschnigg, and Theo Meder. 2013. "How old do you think I am?": A study of language and age in Twitter. In *Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media*, pages 439–448.

Dong Nguyen, Dolf Trieschnigg, and Theo Meder. 2014. Tweetgenie: Development, evaluation, and lessons learned. In *Proceedings of COLING 2014*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Claudia Peersman, Walter Daelemans, and Leona Van Vaerenbergh. 2011. Predicting age and gender in online social networks. In *Proceedings of the 3rd international workshop on Search and mining user-generated contents*, pages 37–44.

James W. Pennebaker and Lori D. Stone. 2003. Words of wisdom: Language use over the life span. *Journal of personality and social psychology*, 85(2):291–301.

Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying latent user attributes in Twitter. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pages 37–44.

Sara Rosenthal and Kathleen McKeown. 2011. Age prediction in blogs: a study of style, content, and online behavior in pre- and post-social media generations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 763–772.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W. Pennebaker. 2005. Effects of age and gender on blogging. In *Proceedings of AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs*, pages 199–205.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263.

Svitlana Volkova, Theresa Wilson, and David Yarowsky. 2013. Exploring demographic language variations to improve multilingual sentiment analysis in social media. In *Proceedings of the 2013 Conference on Empirical Methods on Natural Language Processing*, pages 1815–1827.

Suzanne E. Wagner. 2012. Age grading in sociolinguistic theory. *Language and Linguistics Compass*, 6(6):371–382.

Exploring Syntactic Features for Native Language Identification: A Variationist Perspective on Feature Encoding and Ensemble Optimization

Serhiy Bykh
Seminar für Sprachwissenschaft
Universität Tübingen
sbykh@sfs.uni-tuebingen.de

Detmar Meurers
Seminar für Sprachwissenschaft
Universität Tübingen
dm@sfs.uni-tuebingen.de

Abstract

In this paper, we systematically explore lexicalized and non-lexicalized local syntactic features for the task of Native Language Identification (NLI). We investigate different types of feature representations in single- and cross-corpus settings, including two representations inspired by a variationist perspective on the choices made in the linguistic system. To combine the different models, we use a probabilities-based ensemble classifier and propose a technique to optimize and tune it. Combining the best performing syntactic features with four types of n-grams outperforms the best approach of the NLI Shared Task 2013.

1 Introduction and related work

Native Language Identification (NLI) is the task of identifying the native language of a writer by analyzing texts written by this writer in a non-native language. NLI started to attract attention in computational linguistics with the work of Koppel et al. (2005). Since then, the interest has increased steadily, leading to the First NLI Shared Task in 2013, with 29 participating teams (Tetreault et al., 2013).

The task of NLI is usually treated as a text classification problem with the L1s as classes. A wide range of features, reaching from character or word-based n-grams to different types of syntactic models have been employed in NLI. For example, Wong and Dras (2011) utilized character and part-of-speech (POS) n-grams as well as cross-sections of parse trees and Context-Free Grammar (CFG) features, i.e., local trees. Their approach with a binary representation of non-lexicalized rules (except for those rules lexicalized with function words and punctuation) outperformed a setup using only lexical features, such as n-grams, on data from the International Corpus of Learner English (ICLE; Granger et al., 2002). Swanson and Charniak (2012) used binary feature representations of CFG and Tree Substitution Grammar (TSG) rules replacing terminals (except for function words) by a special symbol. TSG outperformed CFG features in their settings. Among several options, Brooke and Hirst (2012) explored using non-lexicalized CFG production rules in a binary feature encoding on three corpora: ICLE, FCE (Yannakoudakis et al., 2011), and Lang-8 (Brooke and Hirst, 2013a). The authors conclude that including CFG features generally boosts the performance of the system. In the context of the First NLI Shared Task, in Bykh et al. (2013) we showed that non-lexicalized frequency-based CFG features contribute relevant information. Other recent work has focused on TSGs (Tetreault et al., 2012; Brooke and Hirst, 2013b; Swanson and Charniak, 2012; Swanson and Charniak, 2013; Swanson, 2013; Malmasi et al., 2013).

Before extending syntactic modeling further, in this paper we want to systematically explore the range of options involving CFG rule features for NLI. We consider non-lexicalized and lexicalized CFG features, and different feature representations, from binary encodings to a normalized frequency encoding inspired by a *variationist sociolinguistic* perspective.

Previous research in this domain often limited the use of lexicalized rules given that the lexicalization may lead to an unintended topic or domain dependence. Yet, NLI research has since established that lexical features, such as word-based n-grams, are among the best performing features both in single-

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1962–1973, Dublin, Ireland, August 23-29 2014.

and in cross-corpus settings (Brooke and Hirst, 2012; Bykh and Meurers, 2012; Jarvis and Crossley, 2012; Brooke and Hirst, 2013b; Bykh et al., 2013; Gebre et al., 2013; Jarvis et al., 2013; Lynum, 2013), making them an essential component of any approach with state-of-the-art performance. At the same time, the question whether an NLI approach and its results capture general characteristics of language and language learning instead of only encoding the characteristics of a specific data set remains an essential concern. In the experiments in this paper, we thus include experiments on both a topic-balanced single-corpus and on a highly heterogeneous cross-corpus data set.

The range of feature types used in NLI research raises a further question, namely how the different sources of information are best combined. The most simple solution is to put all features into a single vector. However, Tetreault et al. (2012) pointed out that the performance can be increased by using a probability-estimate based ensemble (meta-classifier), which was confirmed in Bykh et al. (2013) and Cimino et al. (2013). But which models are worth integrating into such a meta-classifier? Some of the models may be redundant despite performing well individually; on the other hand, some models may improve the ensemble despite performing relatively poorly by itself. We explore this issue by implementing a basic ensemble optimization algorithm performing model selection.

In terms of the structure of the paper, in section 2 we first introduce the corpora used in the single-corpus and cross-corpus settings. Section 3 then presents the first set of experiments, systematically exploring lexicalized and unlexicalized Context-Free Grammar Rules (CFGR) as features. Given the significant complexity of the overall feature space, we then explore model selection for optimizing the ensemble classifier in section 4. In section 5, we combine the CFGR features with n-grams, resulting in the best accuracy reported for the standard TOEFL11 test set. Section 6 sums up the paper and sketches some directions for future research.

2 Data

The research in this paper makes use of two sets of data:

First, there is the TOEFL11 (T11) data set (Blanchard et al., 2013), which was introduced for the NLI Shared Task 2013 and has become a standard frame of reference for NLI research. We use this standard setup for *single-corpus* evaluation, where each L1 is represented by 1100 essays, of which 100 essays are singled out in the standard *test* set. The remaining 1000 essays per L1 (= T11 *train* $\cup$ *dev*) constitute our training data in the single-corpus settings.

Second, we make use of a range of other learner corpora to study how well the results generalize. Concretely, for our *cross-corpus* settings we employ the NT11 corpus of Bykh et al. (2013), which consists of the ICLE (Granger et al., 2009), FCE (Yannakoudakis et al., 2011), BALC (Randall and Groom, 2009), ICNALE (Ishikawa, 2011), and TÜTEL-NLI (Bykh et al., 2013) corpora. In total NT11 includes 5843 texts, with the following division into languages: Arabic (846), Chinese (1048), French (456), German (500), Hindu (400), Italian (467), Japanese (447), Korean (684), Spanish (446), Telugu (200), Turkish (349). In the cross-corpus settings, we train on NT11 and test on the standard T11 *test* set.

3 Systematically exploring Context-Free Grammar Rules (CFGR)

3.1 Features

In this paper, we focus on the CFG production rules (CFGR) as syntactic features for the task of NLI. CFG rules are the most basic and widely used local syntactic units modularizing the overall syntactic analysis of a sentence. We parsed the T11 and NT11 corpora using the Stanford Parser (Klein and Manning, 2002) and extracted all CFG rules from the T11 and NT11 training sets. On this basis we defined the following tree feature types:

1. $CFGR_{ph}$: Only *phrasal* CFG production rules excluding all terminals
   - S $\rightarrow$ NP VP, NP $\rightarrow$ D NN, ...
2. $CFGR_{lex}$: Only *lexicalized* CFG production rules of the type *preterminal* $\rightarrow$ *terminal*
   - JJ $\rightarrow$ *nice*, JJ $\rightarrow$ *quick*, NN $\rightarrow$ *vacation*, ...
3. $CFGR_{ph \cup lex} = CFGR_{ph} \cup CFGR_{lex}$ (i.e., the union of the above two)

A variationist perspective on feature representation We explore four different feature representations: The two standard ones are a frequency-based (*freq*) representation, where the values are the raw counts of the occurrences of the rule in the given parsed document, and a binary (*bin*) representation, which only indicates whether a rule is present or absent in that document.

Complementing these standard feature representations, we explored two options that take as starting point the observation that CFG rules with the same left-hand side category represent different ways to rewrite that category. So in a sense, under a top-down perspective, there is a choice between different ways of realizing a given category.

This is reminiscent of variationist sociolinguistic analysis, where one studies the linguistic choices made by a given speaker and connects the choices with extra-linguistic variables such as the age or gender of a speaker. For example, in William Labov's field-defining study "The Social Stratification of (r) in New York City Department Stores" from his book "Sociolinguistic Patterns" (Labov, 1972), he found that the presence or absence of the consonant [r] in postvocalic position (e.g., *car*, *fourth*) correlates with the ranking of people in status or prestige, i.e., social stratification. Speakers thus make choices in how to realize a given *variable* by producing one of the *variants* (see also Tagliamonte, 2011). Inspired by this perspective, in Meurers et al. (2013) we discussed how a variationist perspective on syntactic alternations can provide interpretable features for NLI classification.

Under a variationist perspective, producing one of the variants of a given variable also means not choosing the other variants of that variable. So it is this grouping of observations that we want to take into account in terms of encoding local trees as features when we interpret the mother category as the variable to be realized and the different CFG rules with that left-hand side as variants of that variable. This results in two feature representations, a simple one (var_s) and a weighted one (var_w).

The var_s and var_w frequency normalizations for each variant v from the set of variants V realizing a particular variable out of the set of variables $\overline{V}$ is defined as follows:

$$var_s(v \in V) = \frac{f(v)}{F(V)}$$

$$var_w(v \in V) = var_s(v) \cdot w(V)$$

Here, $f(v)$ yields the frequency x of a particular variant v, $F(V)$ is the sum over the frequencies of all variants v realizing the variable V, and $w(V)$ is the weight for the variable V:

$$f(v) = x$$

$$F(V) = \sum_{v \in V} f(v)$$

$$w(V \in \overline{V}) = \frac{F(V)}{\sum_{i=1}^{n} F(\overline{V}_i)}$$

The weighting applied in var_w takes into account the frequency proportion of each variable V in the overall variables set $\overline{V}$, assigning higher weights for more frequent variables. Mathematically it reduces to normalizing each variant by the sum of the frequencies over all variants across all variables, i.e., to the relative frequency of each variant v with respect to the set of all variables $\overline{V}$. At the same time, we will see in the next section that the individual variables keep an independent status in terms of the classification setup, where we train a separate classifier for each variable.

3.2 Results

Classifier We use the *L2-regularized Logistic Regression* from the LIBLINEAR package (Fan et al., 2008), which we accessed through WEKA (Hall et al., 2009). To obtain results for all feature representations which are comparable across the different settings we uniformly scale all values employing the -Z option of WEKA. This means that the *freq* feature representation based on the raw frequencies in essence also becomes normalized. This is particularly relevant in the context of the cross-corpus evaluation, where raw frequencies are particularly questionable given highly variable text sizes.

Single- vs. cross-corpus results The results for the *three feature types* using the *four different feature representations* are presented in Table 1. The chance baseline for the given data setup is 9.1%. There are big accuracy differences between the single- and cross-corpus settings despite very similar feature counts. The drop for the cross-corpus settings is roughly around $\frac{1}{2}$ compared to the single-corpus settings. This is in line with previous results on the same data sets using a wide range of features (Bykh et al., 2013), confirming the fact that obtaining high cross-corpus results remains challenging in NLI.

features	single-corpus (sc): T11 training				
	freq	*bin*	var_s	var_w	feat. #
$CFGR_{ph}$	**50.00%**	44.27%	48.45%	49.82%	14,713
$CFGR_{lex}$	75.73%	72.45%	71.00%	**76.91%**	83,402
$CFGR_{ph \cup lex}$	78.18%	73.55%	75.36%	**78.82%**	98,115

features	cross-corpus (cc): NT11 training				
	freq	*bin*	var_s	var_w	feat. #
$CFGR_{ph}$	21.27%	22.91%	26.27%	**27.73%**	15,253
$CFGR_{lex}$	26.73%	32.00%	28.82%	**36.82%**	78,923
$CFGR_{ph \cup lex}$	28.27%	34.27%	32.55%	**38.82%**	94,176

Table 1: Results for the $CFGR$ feature variants obtained on the standard T11 *test* set

Best feature type The $CFGR_{lex}$ feature type clearly outperforms the more abstract $CFGR_{ph}$ feature type, yielding up to 28% difference in accuracy for the single-corpus and up to 9% for the cross-corpus settings. In contrast to previous research assuming that lexicalized trees are too topic-specific, the results show that $CFGR_{lex}$ is a valuable feature type in both the single-corpus and the cross-corpus settings. The $CFGR_{lex}$ features combine syntactic and lexical information, such as the fact that a given token with a particular POS is used, e.g., the token *can* being used as a *noun* in *There is a **can** of beer in the fridge* instead of as the more frequent *modal verb* use in *He **can** dance*. Note that this is different from using word and POS unigrams as features, where the relevant connection is lost. In both the T11 data, which is topic balanced, for single-corpus evaluation and the very heterogeneous NT11 data containing a wide range of topics for cross-corpus evaluation, we obtained consistently better results for $CFGR_{lex}$ than for $CFGR_{ph}$. Some syntactic rules including lexical information thus seem to generalize well across topics. Combining $CFGR_{ph}$ and $CFGR_{lex}$ into $CFGR_{ph \cup lex}$ gives an additional boost in performance.

Best feature representation There are clear differences in Table 1 between the results for the four feature representations. var_w yields the best accuracies in five out of six settings, across different feature types and corpora.

The results show that WEKA-normalized raw frequencies such as *freq* yield the worst results in a cross-corpus setting but perform very well single-corpus, which is in line with the assumption that raw frequency features do not generalize well. In our experiments, the performance of *freq* in a cross-corpus setting is up to 10.55% worse than what is yielded by var_w, despite comparable single-corpus performance. *freq* also consistently performs worse than var_s in the cross-corpus setting, despite outperforming var_s single-corpus.

Using binary features (*bin*) yields better results cross-corpus than *freq*, whereas in the single-corpus setting it is the other way round. The abstraction introduced by the binary feature representation thus shows a positive effect in terms of the capability of the features to generalize to other data sets.

For the abstract $CFGR_{ph}$ features, var_s performs better than *freq* or *bin* in the cross-corpus setting.

The fact that the var_w is performing consistently better than var_s shows that weighting is important. Hence, incorporating the insight from variationist sociolinguistics is not only conceptually interesting as a theoretical perspective, but also provides a quantitative advantage in terms of performance.

CFGR categories as variables As mentioned above, the best performance is achieved by combining $CFGR_{ph}$ and $CFGR_{lex}$ into the $CFGR_{ph \cup lex}$ feature type using the weighted variationist feature representation var_w. Thus, we focused on that feature type and explored it more in depth. We did so by splitting the overall var_w normalized $CFGR_{ph \cup lex}$ feature set by the variable, i.e., the different *mother nodes*. We trained separate models, where each of those models consists of features encoding the different variants, i.e., the different realizations in which a given mother node can be rewritten. Our aim was to investigate the accuracy of the individual variable-based models and their contribution to the overall performance. Figures 1 and 2 depict the single-corpus (sc) and cross-corpus (cc) accuracies yielded by each individual variable-based model, for presentation reasons shown separately for the $CFGR_{ph}$ and the $CFGR_{lex}$ subsets.

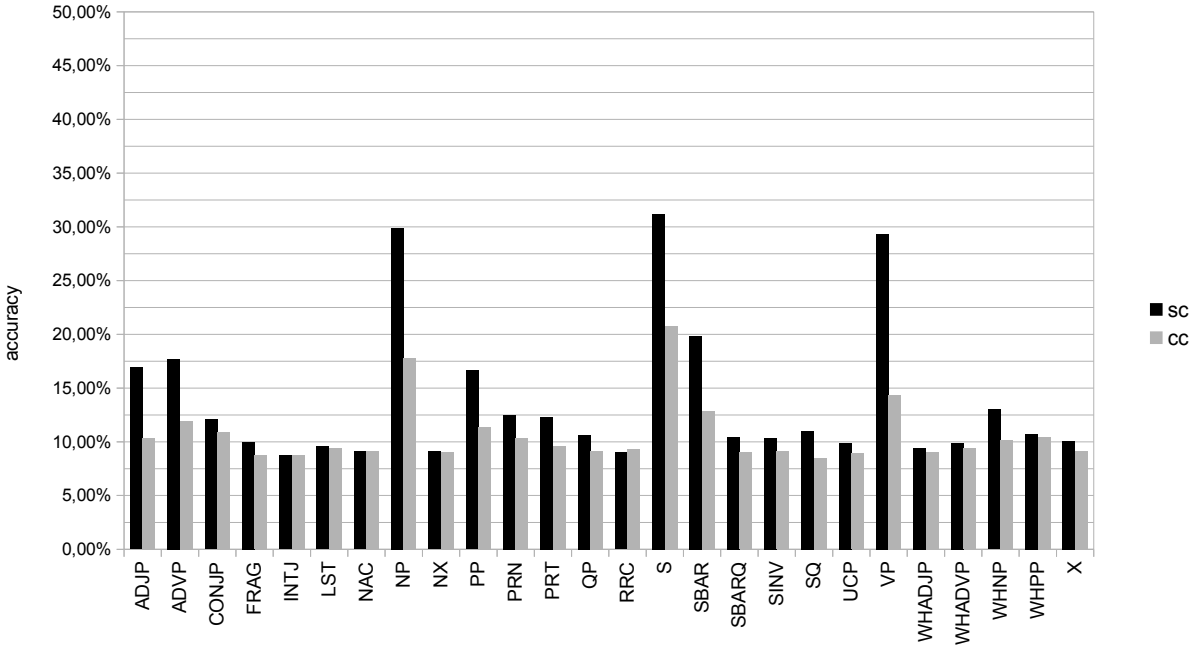


Figure 1: Accuracy for the individual $CFGR_{ph}$ variable based models, var_w normalized

The $CFGR_{ph}$ results in Figure 1 show that a small subset of variables performs relatively well. Most of the models perform poorly, yielding accuracies close to the chance baseline. The best performing variables are essentially the main phrasal categories, such as S, NP, VP, PP, ADJP, ADVP or SBAR.

The results for the $CFGR_{lex}$ in Figure 2 show a similar pattern. There is a subset of variables which perform relatively well, usually models based on the main POS categories, such as the nominal (NN) and verbal (VB) categories as well as adjectives (JJ), prepositions (IN) and adverbs (RB). Some punctuation marks also seem to play a role. The rest of the models yields accuracies around the chance baseline. This might be due to data sparsity given that the main POS categories also are the most frequent. But those main categories also have the highest number of variants through which they can be realized. The good performance of the models for the variables with the highest number of variants thus confirms the assumption that the choice of one of the realization options of a given category is influenced by the L1.

Should we focus only on those high-performing models – or do the other models also contain relevant, independent information which is worth preserving? We address that question in the next section.

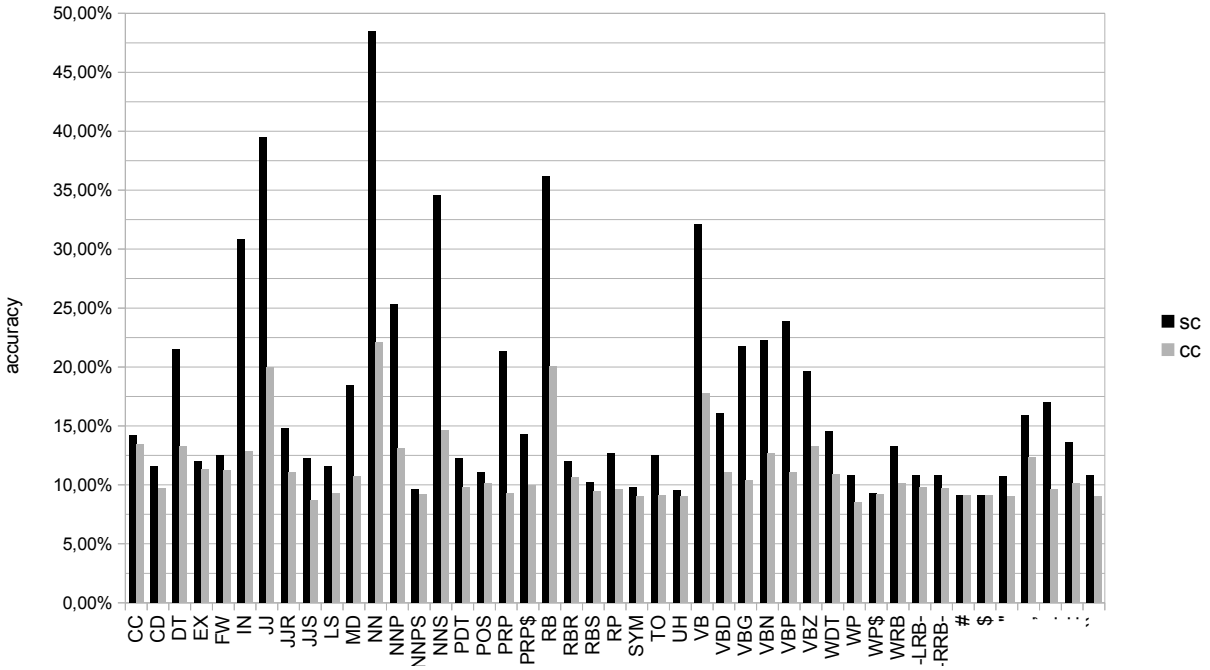


Figure 2: Accuracy for the individual $CFGR_{lex}$ variable based models, var_w normalized

4 Ensemble optimization and tuning

Ensemble generation To combine the individual models, we employ a probability-estimate-based ensemble approach, following Tetreault et al. (2012) and Bykh et al. (2013). This meta-classifier combines the probability distributions provided by the individual classifier for each of the incorporated models as features. To obtain the ensemble training files, we performed 10-fold cross-validation for each model on the corresponding training set and took the probability estimate distributions. For testing, we took the probability estimate distribution yielded by each individual model trained on the corresponding training set and tested on the T11 *test* set. To obtain the probability estimates for the individual models we used LIBLINEAR as described in section 3.2. The ensembles were trained and tested using LIBSVM with an RBF kernel (Chang and Lin, 2011), which outperformed LIBLINEAR for this purpose.

Ensemble optimization (*+opt*) The growing range of features used for NLI raises the question of how to perform model selection. Even when analyzing a single feature type in depth, as we do in section 3.2, we already must determine which of the low-performing models to keep in an ensemble. We approach the question with a simple incremental ensemble optimization algorithm performing model selection.

Algorithm 1 Ensemble Optimization / Ensemble Model Selection

$M_a \leftarrow \{m_1, ..., m_n\}$ ▷ overall ensemble, i.e., all ensemble models
$M_b \leftarrow \emptyset$ ▷ current best performing ensemble
while $M_a \neq \emptyset$ **do** ▷ iterate until M_a is empty
  $m_b \leftarrow \text{MAX}(M_a)$ ▷ get the model with the highest accuracy m_b out of M_a
  $M_t \leftarrow M_b \cup \{m_b\}$ ▷ join the previous best performing ensemble M_b and $\{m_b\}$
  if $\text{ACC}(M_t) > \text{ACC}(M_b)$ **then** ▷ check if the new ensemble is performing better than M_b
    $M_b \leftarrow M_t$ ▷ if the accuracy improves, store the new ensemble in M_b
  end if
  $\text{REMOVE}(m_b, M_a)$ ▷ remove m_b from M_a
end while

In each iteration step the optimization algorithm shown in Algorithm 1 retrieves the current best single model m_b out of the model set M_a (which is initialized with the overall model set for a particular setting), joins it with the previous best performing ensemble M_b (which is initialized to $\emptyset$), compares the accuracy of that new ensemble with the accuracy of the previous best ensemble. It retains the new ensemble as the best ensemble if the accuracy improves, or keeps the previous best ensemble as best ensemble otherwise. In Algorithm 1, we describe only the gist of the optimization, omitting some details to keep it transparent. Some ambiguities have to be resolved. If there are several models in M_a yielding the same accuracy, one has to decide, which of them to pick as the next m_b. We resolve that issue by always picking the model with the least number of features. When several models yield the same accuracy and have the same number of features, we resort to alphabetical order. The optimization is always carried out using 10-fold cross-validation results on the training data (to obtain the accuracy ranking on M_a and to perform each optimization step). The *test* set is not part of the optimization at any point. Only after optimization is the resulting ensemble applied to the *test* set and we report the corresponding accuracies.

Ensemble tuning (*+all*) In order to further tune the ensemble, we explore the following idea: We generate a *single ensemble model* m_{n+1} based on *all* of the features used in a particular setting, i.e., all the features incorporated by the models $m_1 \dots m_n$. Then we include that m_{n+1} model in the M_a ensemble as just another model, and use that new M_a^{+1} ensemble either directly or as basis for the optimization. Since m_{n+1} incorporates all of the features of interest for a particular setting, it is expected to yield more reliable probability estimates than the other individual ensemble models in M_a^{+1}, each covering only a subset of that feature set. Incorporating such an m_{n+1} into the ensemble may stabilize the resulting system, i.e., the machine learning algorithms may learn to rely on m_{n+1} in settings, where the rest of the included models $m_1 \dots m_n$ show a rather poor individual performance and are of limited use. In the tables and explanations below, we refer to the model m_{n+1} as [*all*] and to the M_a^{+1} ensemble as *+all*.

For building the m_{n+1} model included in the M_a^{+1} ensemble there are two options. We can build it on the basis of the probabilities of the models or on the union of the original feature values of those models. In the former case, the final ensemble model essentially is a meta-meta-classifier. For the settings integrating the same type of feature representations (cf. results in Tables 2 and 4), we use the original feature values merged into a single vector to build m_{n+1}. For the settings integrating different feature types (cf. results in Table 6), we use the probability estimates from the models $m_1 \dots m_n$ to build m_{n+1}.

Ensemble results for the CFGR variables The ensemble results for the separate variable-based models for the $CFGR_{ph \cup lex}$ feature type are presented in Table 2. We provide single-corpus (*sc*) and cross-corpus (*cc*) results for different ensemble settings, where *+/- opt* states whether ensemble optimization was performed, and *+/- all* whether tuning was employed. Concretely, *(-opt, -all)* means that the ensemble M_a was used without any optimization or tuning, and correspondingly *(+opt, +all)* means that the optimized and tuned version of M_a (i.e., the optimized version of the ensemble M_a^{+1}) was employed. In the remaining two cases *(+opt, -all)* and *(-opt, +all)* either optimization or tuning was used, respectively. The column *baseline* lists the corresponding results from Table 1, which were obtained by putting all the features in a single vector. The number in parentheses specifies the number of models combined in the ensemble: in the *features* column, it shows the overall number of separate variable-based models, and in the *+opt* columns, it is the number of models selected by the optimization algorithm.

features	data	baseline	ensemble			
			-opt		+opt	
			-all	+all	-all	+all
$CFGR_{ph \cup lex}$ (71)	sc	78.82%	66.00%	79.18%	71.27% (14)	**79.64% (8)**
	cc	38.82%	18.09%	34.18%	32.55% (10)	**39.00% (1)**

Table 2: Results for the $CFGR_{ph \cup lex}$ ensembles with different optimization settings

The results show that generating an ensemble using all of the individual variable-based models without optimization and tuning *(-opt, -all)* leads to a big accuracy drop compared to the baseline. The fact that

the drop in the cross-corpus setting is more than 20% is particularly striking. We assume that this is due to the poor performance of most of the individual models, yielding probabilities of little use overall. The few relatively well-performing models we discussed in section 3.2 apparently are flooded by the noise introduced by the others. Thus, for a set of rather low-performing models without any optimization, it seems preferable to provide the classifier with access to the individual features instead of to the noisy probability estimates. The optimization *(+opt, -all)* leads to a clear improvement over the non-optimized settings. In the single-corpus setting only 14 of the 71 models were kept and in cross-corpus only 10.

Table 3 shows the selected models in the order in which they are selected by the ensemble optimization algorithm. For *(+opt, -all)*, the table basically consists of the best performing variables (i.e., the models containing as features the different ways to rewrite the given mother category) as discussed in section 3.2, suggesting that the algorithm makes meaningful choices.

data	$CFGR_{ph \cup lex}$: selected models	
	+opt, -all	+opt, +all
sc	[NN]+[JJ]+[RB]+[NNS]+[VB]+[NP]+[S]+[VP] +[IN]+[VBP]+[VBG]+[VBN]+[NNP]+[,] (14)	[*all*]+[NN]+[JJ]+[RB]+[PRP]+[VBN]+[NNP]+[WDT] (8)
cc	[NN]+[JJ]+[NNS]+[NP]+[RB]+[VB]+[VP]+[NNP] +[S]+[IN] (10)	[*all*] (1)

Table 3: The $CFGR_{ph \cup lex}$ model sets selected by optimization

The flipside of the coin is that low-performing models generally were not found to have a positive effect and thus were not included. Yet, optimization by itself is not successful overall given that the *(+opt, -all)* accuracy remains below the single feature set baseline.

Applying tuning without optimization *(-opt, +all)* outperforms the optimization result. Thus, including the overall model [*all*] in the ensemble improves the meta-classifier. In the single-corpus setting, the accuracy is slightly higher than the baseline, in cross-corpus it remains below the baseline.

Turning on both optimization and tuning *(+opt, +all)* yields the overall best results of Table 2, 79.64% for single-corpus and 39% for the cross-corpus setting. The corresponding entry in Table 3 shows that tuning significantly reduces the number of selected models. This is not unexpected given that the overall model [*all*] essentially includes all the information. In the cross-corpus setting, [*all*] indeed is the only model selected. Interestingly, in the single-corpus setting, the optimization algorithm identifies some additional models to improve the accuracy, mainly ones that also perform well individually. While this amounts to adding information that in principle is already available to the [*all*] model, the improvement may stem from the abstract nature of the probability estimates used as features of the meta-classifier. When both optimization and tuning are applied, the tuning apparently stabilizes the ensemble leading to higher performance, and the optimization algorithm further improves the result by reducing the noise.

5 Combining CFGR with four types of n-grams

Based on the systematic exploration of the CFGR domain, we turn to combining our new feature type $CFGR_{ph \cup lex}$ with n-gram features as the best performing features for NLI (Tetreault et al., 2013; Jarvis et al., 2013). Adapting the n-gram approach we presented in Bykh and Meurers (2012), we use all recurring n-grams with $1 \leq n \leq 10$ at different levels of representation, including the word-based (W), open-class POS-based (OP) and POS-based (P) n-grams from our previous work as well as lemma-based (L) n-grams (Jarvis et al., 2013). We employ binary feature encoding for all n-gram types.

For POS-tagging we use the OpenNLP[1] toolkit, for lemmatizing we employ the MATE[2] tools (Björkelund et al., 2010). To obtain a fine grained, flexible n-gram setting, we generate an ensemble model for each n-gram type and each n, which results in 40 n-gram models.

[1] `http://opennlp.apache.org`
[2] `https://code.google.com/p/mate-tools`

Table 4 provides the results for the n-gram ensembles built on the basis of the recurring word-, lemma-, POS-, OCPOS-based n-grams with $1 \leq n \leq 10$ in the same format as Table 2 for $CFGR_{ph \cup lex}$.[3] Different from the $CFGR_{ph \cup lex}$ case, the results for the n-gram ensemble model without optimization or tuning *(-opt, -all)* already are 4–5% higher than the single vector baseline.

features	data	baseline	ensemble			
			-opt		+opt	
			-all	+all	-all	+all
N-GRAMS (40)	sc	77.09%	82.27%	82.55%	**83.00% (13)**	82.27% (8)
	cc	31.00%	34.91%	34.55%	**36.45% (6)**	35.45% (6)

Table 4: Results for the n-gram ensembles with different optimization settings

The best results, 83% for single-corpus and 36.45% for the cross-corpus setting, are obtained by applying the optimization. The n-gram ensembles seem to benefit more from optimization than from tuning in general. The feature counts for the n-grams (single-corpus: 4,822,874; cross-corpus: 3,687,375) are far higher than for $CFGR_{ph \cup lex}$ (single-corpus: 98,115; cross-corpus: 94,176), so there may be more noise in the [*all*] model, making it less useful for the tuning step.

Table 5 lists the models selected by the optimization algorithm in order in which they are selected. The n-gram types and the n of the model is indicated, e.g., "[OP-3]" means "OCPOS-based trigrams".

data	N-GRAMS: selected models	
	+opt, -all	+opt, +all
sc	[W-2]+[L-2]+[W-1]+[L-1]+[L-3]+[W-3]+[OP-3] +[OP-1]+[OP-5]+[P-3]+[P-5]+[P-2]+[OP-8] (13)	[*all*]+[W-2]+[L-2]+[W-1]+[L-1]+[L-3]+[OP-4]+[L-4] (8)
cc	[W-2]+[W-1]+[L-1]+[L-3]+[W-3]+[OP-2] (6)	[W-2]+[W-1]+[*all*]+[L-1]+[L-3]+[P-4] (6)

Table 5: The n-gram model sets selected by optimization

For the more surface-based n-gram (word- and lemma-based), the optimizer selected only up to $n = 3$, whereas for the more abstract ones (POS- and OCPOS-based), models up to $n = 8$ were included. Thus, when abstracting from the surface, one can get some useful information out of longer n-grams that apparently is not contained in the short surface-based ones. Different from the $CFGR_{ph \cup lex}$ variables-based ensemble, we here find that relatively low-performing models such as those considering longer n n-grams are kept when optimizing the ensemble.

Having established the performance of the n-gram ensembles, we can turn to combining the $CFGR_{ph \cup lex}$ and n-gram models. The results are presented in Table 6.

features	data	ensemble			
		-opt		+opt	
		-all	+all	-all	+all
(a) $CFGR_{ph \cup lex}$ (71) + N-GRAMS (40)	sc	82.09%	82.91%	82.91% (20)	**83.55% (6)**
	cc	34.09%	36.00%	36.73% (8)	**38.45% (3)**
(b) $CFGR_{ph \cup lex}$ (71) + N-GRAMS [+opt, -all] (ME)	sc	83.09%	83.73%	82.64% (4)	**84.18% (5)**
	cc	37.36%	39.55%	38.00% (3)	**40.27% (3)**
(c) $CFGR_{ph \cup lex}$ [+opt, +all] (ME) + N-GRAMS (40)	sc	83.73%	**84.82%**	84.73% (13)	83.82% (13)
	cc	36.82%	38.91%	42.00% (5)	**43.00%** (4)
(d) $CFGR_{ph \cup lex}$ [+opt, +all] (ME) + N-GRAMS [+opt, -all] (ME)	sc	**83.45%**	**83.45%**	**83.45% (2)**	83.36% (2)
	cc	41.27%	**42.00%**	41.27% (2)	40.55% (2)

Table 6: Optimization results combining n-grams and $CFGR_{ph \cup lex}$

[3]For space reasons, we cannot present the individual results for the separate n-gram models here, but interested readers can consult Bykh and Meurers (2012), where word-, POS- and OCPOS-based n-gram results are discussed in detail. The lemma-based n-grams we are adding here perform very much like the word-based n-grams.

We explore four different ways to combine the two model sets, and the table shows the best results for each of the setups in bold, once for the single-corpus and once for the cross-corpus setting.

For the results of setup (a), we use the ensemble consisting of all individual models separately.

In (b), the $CFGR_{ph \cup lex}$ models are included as in (a), but we replace the n-gram models by a *single meta-ensemble model (ME)* generated using the best n-grams setting *(+opt, -all)*, which consists of 13 models for single-corpus and six models for the cross-corpus setting (see Table 4). ME thus is a *meta-meta-classifier*, generated by applying the ensemble model generation routine to an ensemble.

In (c), we invert the (b) setting: The $CFGR_{ph \cup lex}$ features are replaced by a meta-ensemble generated using the best performing $CFGR_{ph \cup lex}$ setting *(+opt, +all)*, which consists of eight models for the single-corpus, and one model for the cross-corpus setting (see Table 2).

Finally, in (d) we combine the meta-ensemble for $CFGR_{ph \cup lex}$ with the meta-ensemble for the n-grams obtaining an ensemble consisting of two models

The best results of 84.82% in the single-corpus setting and 43% cross-corpus, underlined in the table, are obtained in setup (c). These are the overall best results across all experiments described in this paper. The best result in the single-corpus setting involves tuning only, whereas in the cross-corpus setting it involves tuning and optimization selecting the models [*all*]+[*CFGR +all +opt*]+[W-2]+[W-1].

The single-corpus accuracy of 84.82% is the best result reported so far for the NLI Shared Task 2013 data with the T11 *train* $\cup$ *dev* set for training and the T11 *test* set for testing. The best previous result was 83.6% (Jarvis et al., 2013).

In the cross-corpus setting, the 43% accuracy also outperforms the previous best result on the NT11 data (Bykh et al., 2013) by 4.5%.

In sum, the overall best results in the single-corpus and cross-corpus settings are obtained starting with the whole n-gram model set plus an optimized $CFGR_{ph \cup lex}$ meta-ensemble. This confirms the usefulness of the optimized ensemble setup and underlines that combining a range of linguistic properties, from n-grams at different levels of abstraction to local syntactic trees characteristics, is a particularly fruitful approach for native language identification as a good example of an experimental task putting linguistic modeling to the test with real-life data.

6 Conclusions

In the research presented, we systematically explored *non-lexicalized* and *lexicalized CFG production rules (CFGR)* as features for the task of NLI using both single-corpus and cross-corpus settings. Including lexicalized CFG rule features clearly improved the results in both setting so that it seems worthwhile not to discard them a priori, which was the standard in previous research.

Pursuing a *variationist perspective* to CFGR feature representation resulted in improved performance and it supported an in-depth exploration of the contribution of the different variables and variants as well as of the value of local syntactic features for NLI in general. Training a separate classifier for each variable provides quantitative advantages by facilitating high-performing ensemble setups and supports a qualitative discussion of the categories reflecting the choices made by the learners with a given L1.

Investigating different meta-classifier setups, we explored *ensemble optimization and tuning* techniques that improved the accuracy over putting all features in a single vector or a basic ensemble setup.

Combining the syntactic CFGR with four types of n-grams yielded a *single-corpus* accuracy of *84.82%* on the TOEFL11 *test* set. To the best of our knowledge this is the highest accuracy reported so far on this standard data set of the NLI Shared Task 2013. The combined model also outperformed our best previous cross-corpus result on the NT11 corpus.

In terms of future work, we intend to explore a broader range of linguistic features from a variationist perspective, for example on the morphological level. To investigate the generalizability of the types of features used, we also plan to apply our approach to NLI targeting second langauges other than English.

References

Anders Björkelund, Bernd Bohnet, Love Hafdell, and Pierre Nugues. 2010. A high-performance syntactic and semantic dependency parser. In *Demonstration Volume of the 23rd International Conference on Computational Linguistics (COLING 2010), Beijing*, pages 23–27. https://code.google.com/p/mate-tools/.

Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. 2013. TOEFL11: A corpus of non-native english. Technical report, Educational Testing Service.

Julian Brooke and Graeme Hirst. 2012. Robust, lexicalized native language identification. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 391–408, Mumbai, India.

Julian Brooke and Graeme Hirst. 2013a. Native language detection with 'cheap' learner corpora. In Sylviane Granger, Gaëtanelle Gilquin, and Fanny Meunier, editors, *Twenty Years of Learner Corpus Research. Looking Back, Moving Ahead. Proceedings of the First Learner Corpus Research Conference (LCR 2011)*, Louvain-la-Neuve, Belgium. Presses universitaires de Louvain.

Julian Brooke and Graeme Hirst. 2013b. Using other learner corpora in the 2013 nli shared task. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

Serhiy Bykh and Detmar Meurers. 2012. Native language identification using recurring n-grams – investigating abstraction and domain dependence. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 425–440, Mumbay, India.

Serhiy Bykh, Sowmya Vajjala, Julia Krivanek, and Detmar Meurers. 2013. Combining shallow and linguistically motivated features in native language identification. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Andrea Cimino, Felice Dell'Orletta, Giulia Venturi, and Simonetta Montemagni. 2013. Linguistic profiling based on general–purpose features and native language identification. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874. Software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear.

Binyam Gebrekidan Gebre, Marcos Zampieri, Peter Wittenburg, and Tom Heskes. 2013. Improving native language identification with tf-idf weighting. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

S. Granger, E. Dagneaux, and F. Meunier. 2002. *International Corpus of Learner English*. Presses Universitaires de Louvain, Louvain-la-Neuve.

Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot, 2009. *International Corpus of Learner English, Version 2*. Presses Universitaires de Louvain, Louvain-la-Neuve.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. In *The SIGKDD Explorations*, volume 11, pages 10–18.

Shin'ichiro Ishikawa. 2011. A new horizon in learner corpus studies: The aim of the ICNALE projects. In G. Weir, S. Ishikawa, and K. Poonpon, editors, *Corpora and language technologies in teaching, learning and research*, pages 3–11. University of Strathclyde Publishing, Glasgow, UK. http://language.sakura.ne.jp/icnale/index.html.

Scott Jarvis and Scott A. Crossley, editors. 2012. *Approaching Language Transfer through Text Classification: Explorations in the Detection-based Approach*. Second Language Acquisition. Multilingual Matters.

Scott Jarvis, Yves Bestgen, and Steve Pepper. 2013. Maximizing classification accuracy in native language identification. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

Dan Klein and Christopher D. Manning. 2002. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*.

Moshe Koppel, Jonathan Schler, and Kfir Zigdon. 2005. Determining an author's native language by mining a text for errors. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining (KDD '05)*, pages 624–628, New York.

William Labov. 1972. *Sociolinguistic Patterns*. University of Pennsylvania Press.

André Lynum. 2013. Native language identification using large scale lexical features. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

Shervin Malmasi, Sze-Meng Jojo Wong, and Mark Dras. 2013. Nli shared task 2013: Mq submission. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

Detmar Meurers, Julia Krivanek, and Serhiy Bykh. 2013. On the automatic analysis of learner corpora: Native language identification as experimental testbed of language modeling between surface features and linguistic abstraction. In *Diachrony and Synchrony in English Corpus Studies*, Frankfurt am Main. Peter Lang.

Mick Randall and Nicholas Groom. 2009. The BUiD Arab learner corpus: a resource for studying the acquisition of L2 english spelling. In *Proceedings of the Corpus Linguistics Conference (CL)*, Liverpool, UK.

Benjamin Swanson and Eugene Charniak. 2012. Native language detection with tree substitution grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 193–197, Jeju Island, Korea, July. Association for Computational Linguistics.

Ben Swanson and Eugene Charniak. 2013. Extracting the native language signal for second language acquisition. In *Proceedings of NAACL-HLT*. Association for Computational Linguistics.

Ben Swanson. 2013. Exploring syntactic representations for native language identification. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-8) at NAACL-HLT 2013*, Atlanta, GA.

Sali A. Tagliamonte. 2011. *Variationist Sociolinguistics: Change, Observation, Interpretation*. John Wiley & Sons.

Joel Tetreault, Daniel Blanchard, Aoife Cahill, and Martin Chodorow. 2012. Native tongues, lost and found: Resources and empirical evaluations in native language identification. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 2585–2602, Mumbai, India.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Building Educational Applications Using NLP*, Atlanta, GA, USA, June. Association for Computational Linguistics.

Sze-Meng Jojo Wong and Mark Dras. 2011. Exploiting parse structures for native language identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1600–1610, Edinburgh, Scotland, UK., July.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 180–189, Stroudsburg, PA, USA. Association for Computational Linguistics. Corpus available: `http://ilexir.co.uk/applications/clc-fce-dataset`.

Applying automatically parsed corpora to the study of language variation

Jelke Bloem **Arjen Versloot** **Fred Weerman**

Amsterdam Center for Language and Communication
University of Amsterdam
1012 VB Amsterdam, Netherlands
{j.bloem, a.p.versloot, f.p.weerman}@uva.nl

Abstract

In this work, we discuss the benefits of using automatically parsed corpora to study language variation. The study of language variation is an area of linguistics in which quantitative methods have been particularly successful. We argue that the large datasets that can be obtained using automatic annotation can help drive further research in this direction, providing sufficient data for the increasingly complex models used to describe variation. We demonstrate this by replicating and extending a previous quantitative variation study that used manually and semi-automatically annotated data.

We show that while the study cannot be replicated completely due to limitations of the existing automatic annotation, we can draw at least the same conclusions as the original study. In addition, we demonstrate the flexibility of this method by extending the findings to related linguistic constructions and to another domain of text, using additional data.

1 Introduction

There are many examples of linguistic variation that are not easily explained in terms of rules. One may find two grammatically correct constructions that can be used to express similar meanings, yet speakers still use both of them. A well-known example in English is the dative alternation, where a transitive verb such as *give* can be phrased as a double object construction (1) or as a prepositional dative (2):

(1) He gave his friend the ticket.

(2) He gave the ticket to his friend.

Studies of such phenomena tend to find that there are multiple variables that may influence whether a speaker chooses one or the other construction. This has prompted various multivariate studies by quantitative linguists to analyze instances of such variation in language corpora, starting with Gries (2001), or Bresnan et al. (2007) for a study on the dative alternation. The multivariate statistical models that these studies employ can quantify the contribution of each variable to the variation in probabilistic terms, rather than examining them in isolation.

We show the benefits of using automatically annotated corpora for the study of language variation by replicating a previous, manual multivariate study on Dutch verbal cluster variation (De Sutter, 2009), and extending it to fit more types of clusters. We also show that the same variables are explanatory in at least two different text domains, Wikipedia text and European Parliament proceedings. The larger scale of our investigation allows us to generalize the claims of the previous study to Dutch two-verb cluster variation in general. This topic makes for a good methodological case study for the use of automatically annotated corpora, as verbal clusters are a widely studied phenomena in Dutch syntactic literature (Evers, 1975; Den Besten and Edmondson, 1983; Haegeman and van Riemsdijk, 1986; Zwart, 1996; Wurmbrand, 2004), and the optionality in verbal cluster order has received particular attention in some recent dissertations (De Sutter, 2005; Coussé, 2008; Arfs, 2007). In addition, a methodologically sound quantitative study, which did not make use of automatically annotated data, already exists to compare to (De Sutter, 2009).

This particular case of variation allows for a lot of optionality. The verbal clusters found at the ends of Dutch clauses allow for almost free order variation when there are two verbs. For example, in two-verb clusters the auxiliary verb can be positioned before or after the main verb:

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1974–1984, Dublin, Ireland, August 23-29 2014.

(3) Ik denk dat ik het **begrepen heb**
I think that I it understood have
'I think that I have understood it.'

(4) Ik denk dat ik het **heb begrepen**
I think that I it have understood

As in the dative alternation example, the variation in these two-verb clusters seems to be influenced by multiple variables, beyond the constraints of grammatical rules. Therefore, we consider the multivariate model by De Sutter (2009) to be the most accurate model of Dutch verbal clusters developed so far. Unfortunately, it is also too limited and does not cover all of the constructions that are generally considered to be two-verb clusters. The author claims to have done this for reasons of methodological rigor. However, in a multivariate model the contribution of each variable can be studied independently. If an additional verbal cluster construction is added and it is marked with a variable as being a different construction, it should not make much of a difference for the other variables, assuming that the same set of variables is involved for all verbal cluster constructions. We have included these additional cluster types to create a model with a larger scope, and show that the effects of de Sutter's smaller model are still present. We also compare to a smaller model of our own, created from the large corpus but without the additional cluster types, to verify that our results aren't just an effect of including the additional constructions.

Another reason for excluding the other types of verbal clusters might have been the annotation effort involved in finding corpus examples of them. We avoid this issue by using an automatically annotated corpus, and can extract large samples of various types of constructions simply by defining what counts as a cluster in the syntactic annotation.

In section 2 we briefly discuss Dutch verbal clusters and the variation found in them. We then discuss previous work on modeling of Dutch verbal clusters, including the model by De Sutter (2009) in section 3. Section 4 describes the automatically annotated corpus that we used. In section 5 we discuss the model we created from this data and compare it to that of De Sutter (2009). We also compare models created from two different text types. Section 6 discusses the implications of these results.

2 Verbal cluster variation

In this section, we will briefly summarize how verbal clusters are formed, and discuss the extent of the variation they exhibit. To refer to the two verbal cluster orders, we will follow terminology introduced by Stroop (1970), where construction (3) is called the 2-1 order and construction (4) is 1-2. This is because the finite auxiliary is considered to be the verb that is highest in the syntactic tree, while the main verb is the lowest. This fact lets us number the verbs.

In generative literature, the formation of these clusters is described as a verb movement process known as verb raising, where the main verb is moved upwards in the syntactic tree from its phrase to be joined with the auxiliary verb. This explains the common observation that verbal clusters cannot be interrupted (Evers, 1975), though there are some instances of cluster interruptions, particularly in Flemish Dutch (Evers, 2003). A broad overview of verb raising across Germanic languages is provided in Wurmbrand (2006).

There are various types of two-verb clusters that exhibit order variation:

Auxiliary cluster Examples (3) and (4) show two-verb clusters with auxiliary heads. Following De Sutter (2009) we consider this to be any cluster that is headed by the auxiliaries *hebben* "to have", *zijn* "to be" and *worden* "to be".

Modal cluster A modal verb (*willen* "want", *kunnen* "can") may also be used as a cluster head. Modal clusters are generally treated as a different construction in the literature, as different grammatical rules may apply to it, particularly in other Germanic languages. In Dutch we can observe that the 2-1 order is far more common in this construction, and some authors even say there is no optionality here (i.e. Zuckerman (2001)). We observe that in the Wikipedia part of the Lassy Large corpus, modal clusters occur in the 1-2 order only 0.5% of the time. However, they are considered to be grammatical.

Clusters with other verbs There are verbs such as *staan* 'to stand' and *helpen* 'to help' that can also be raised to form a verbal cluster in certain contexts. This list includes causal verbs, aspectual verbs, and some harder to classify ones, and seems too diverse to be grouped together. For brevity and due to their relative rarity (together

they form about 5.5% of the clusters) we did not include these constructions in our study, though it would certainly be possible to explore them in a large corpus study and perhaps contribute to their classification.

Te-infinitival clusters In the cluster types we discussed so far the auxiliary verb was finite, but there are infinitival clusters as well, where both the auxiliary and the main verb are infinite, and the main verb is marked by the infinitival marker *te*. These clusters are uncommon (2.2% of our dataset) and have not been the focus of any study on variation, though since these clusters form a clear group, we have included them. They occur with both auxiliary and modal heads.

Main clause cluster Verbal clusters can occur in main clauses as well, though in a different form. Stroop (2009) states that three-verb clusters in main clauses are comparable to two-verb clusters in subordinate clauses. In that study, he discusses various cluster types in a corpus of spoken Dutch and observes the distributions between 1-2 and 2-1 orders. While there are three verbs in these main clause clusters, only the last two verbs have free order variation, due to the V2-effect present in Germanic languages:

(5) De fuut **kan** in alle wateren van enig formaat **aangetroffen worden** .
The grebe can in all waters of some size found be .
'The grebe can be found in all bodies of water of substantial size.'

(6) Wegwerpbatterijen **kunnen** niet **worden opgeladen** .
Disposable batteries can not be charged .
'Disposable batteries cannot be charged.'

The finite verb must always be in verb-second position.

Stroop furthermore observes, when looking at larger clusters, that variations of three-verb subordinate clause clusters are distributed similarly to variations of four-verb main clause clusters (that have three verbs with varying order). This observation holds for both Dutch and Flemish data, even though the frequencies of orders are different between the languages. Factors that influence order variation seem to be able to affect main clause and subordinate clause clusters in the same way, as Stroop demonstrated for the regional factor. We are therefore convinced that main clause clusters should be included in studies on verbal cluster variation.

The rules and mechanisms discussed in generative literature allow for a lot of optionality, as discussed above, and thus mainly outline the constructions in which order variation can occur. These accounts generally left open the question of the variation found in the surface order. It appears that syntacticians did not concern themselves with explaining it and considered it to be an effect of a non-syntactic process. This is evident in the analysis adopted by Haegeman and van Riemsdijk (1986).

The issue of explaining the factors that influence the choice between two variants was later picked up by other researchers who were interested in non-syntactic effects as well. Coussé et al. (2008) provide a summary of recent work on verbal cluster variation, in particular, three dissertations on the topic (De Sutter, 2005; Coussé, 2008; Arfs, 2007). A diverse set of variables that may influence the use of 1-2 and 2-1 cluster orders has been found, and they group them into four categories: contextual factors (region and mode of communication), rhythmic factors (adherence to the standard stress pattern of Dutch), semantic factors, and discourse factors (mainly the syntactic priming effect).

From this, Coussé et al. (2008) conclude that the choice of verbal cluster order is influenced by a complex set of interacting factors. Therefore, any model representing this phenomenon would need to take many factors into account. The multivariate modeling technique used by De Sutter (2005) seems to fit this criterion.

We will now discuss some related studies where multivariate modeling has been applied to the study of language variation, as well as a proposal to involve automatically annotated data in linguistic studies, and then discuss De Sutter's multivariate study on verbal cluster order that we are replicating.

3 Multivariate modeling of language variation

In corpus linguistics, linguistic phenomena examined over larger sets of data have often been found to be too complex to model in terms of a single independent variable. In this case, rather than running one statistical test for each variable, it is considered best practice to test for all variables in a single test. The statistical power of such a test will be greater than that of running several tests and applying corrections, which increases the chances of erroneously rejecting the null hypothesis with each test. Starting with Gries (2001), this methodology has

been applied to the study of language variation. A well-known example is the dative alternation study of Bresnan et al. (2007). In these studies, multivariate statistical models are used to quantify the effect size of each variable, indicating their relative importance. Being corpus studies, these quantitative studies generally also emphasize evidence from larger samples of language and operationalize their variables in a precise way.

Gries (2001) discusses the case of English transitive phrasal verbs (*to pick up the book* / *to pick the book up*), and explains 84% of the variation based on a multivariate model containing many variables from previous work. He also critiques previous work on language variation. Firstly, studies often relied on introspective analysis and made-up examples, which can be too subjective, not representative, and in the case of acceptability judgements, not necessarily output of the human language system. Secondly, when only a single variable is examined at a time, many other possible variables may influence the result, even when seemingly minimal pairs are used. Lastly, the provided models cannot be used to predict variation in natural discourse situations. Variables are not weighed, and if two variables have conflicting preferences, the possibility of a prediction is already ruled out.

Despite the methodological precision, the data of the study consisted of only 403 sentences in total, about 200 for each construction. Furthermore, they were chosen manually, introducing some subjectivity into the study. When the statistical tendencies of around 30 variables are studied, 403 sentences do not provide a lot of detail. We believe that these quantitative studies can be further improved by using data from automatically annotated corpora.

3.1 Using automatically annotated data

Corpora that have been annotated by an automatic parser rather than manually, will contain far more data, allowing for larger sample sizes of particular constructions to be found. These samples can be extracted automatically as well. To do so, an exact definition of what constitutes the construction must be formulated at the level of syntax — for example, two verbs that are adjacent and in the same subordinate clause, with one being the head of the other. All of the sentences that match this definition can then be extracted from the corpus. This process avoids subjectivity beyond the definition of the construction. However, it limits the variables that can be used in the study to the ones that are, or can be, automatically annotated in a corpus. It also comes at the cost of accuracy, though in most cases, it is expected that the larger sample size makes up for any random parsing errors. Systematic errors (for example, constructions that the parser consistently fails to annotate) may skew the results however, so care should be taken that the parser is able to annotate the constructions of interest at all.

Some automatically annotated corpora have become available in recent years, though they have not yet been widely used for linguistic study. Nevertheless, I will discuss a few studies that have applied automatically annotated corpora for the purposes of language variation research. The use of automatically parsed corpora as a linguistic resource has been discussed by van Noord and Bouma (2009). They argue that parsing technology has advanced enough to be incorporated into other language technology that can build upon its results. This allowed for the creation of very large corpora of parsed sentences that are sufficiently large to compensate for any parsing error 'noise'. Several applications of such very large corpora can be found in the article.

For the Dutch language, the potential uses of the 500 million word automatically annotated LASSY Large corpus have been discussed (van Noord, 2009). This paper also mentions that some natural language processing tasks, such as learning selection restrictions of specific verbs, cannot be performed successfully using smaller corpora because the sample size would be too small. This issue is likely to apply to any linguistic phenomenon at the level of open-class lexical items (i.e. nouns, verbs or combinations thereof), most of which are rare. For the case of verbal clusters, this shows that using such a large corpus would be required to study the effect of specific main verbs on the order variation.

Automatically annotated data has already been used to study language variation for the famous case of the dative alternation, discussed in the introduction (examples (1), (2)). Lehmann and Schneider (2012) used a 580 million word dependency-parsed corpus of English to study the influence of specific lexical types on this alternation. These types consist of 'triplets' of words: a ditransitive verb, a direct object head and an indirect object head. These three slots of the triplet are all filled with open-class words, therefore requiring vast amounts of data to study: "We indeed find that 580 million words are barely enough data to yield results for full lemma triplets". Unfortunately, the study is a monovariate study where lexical type was the only variable investigated. This does not exclude the possibility of other underlying factors, which the lexemes may happen to correlate with, influencing the variation.

The only other similar variational study we are aware of is on the optionality of the Dutch *om*-complementizer, a variation similar to the optionality of English *that* for relative clauses. In Dutch, *om* as a head of a *to*-infinitival clause is optional in many, though not all cases:

(7) Anna probeerde (om) een bom onschadelijk te maken
Anna tried (CMP) a bom harmless to make
'Anna tried to defuse a bomb.'

Bouma (2013) provides further examples, and creates a model of this variation using data from part of the LASSY Large corpus. This model is multivariate, and includes both lexical effects and other variables, such as clause length. It is implemented as a mixed-effects logistic regression model, in which the verbal governor (i.e. the verb *proberen* in example (7)) is a random effect, and the other non-lexical variables are fixed effects. The other variables are all related to processing complexity, as it is theorized that *om*-insertion reduces processing load by reducing ambiguity. He finds that the best model includes both semantic and complexity features and report a concordance score of 0.809, indicating that the model has modest predictive qualities.

While there are not many examples of such studies, using automatically annotated sources of data appears to be fruitful for studying complex phenomena where many factors may play a role, and a large sample size is desirable. We consider Dutch verbal cluster variation to be such a phenomenon.

3.2 Verbal cluster variation

Going back to the topic of variation in Dutch verbal clusters, we believe that the most methodologically sound study can be found in the dissertation of De Sutter (2005), summarized in De Sutter (2009). In this study, various variables from previous work on verbal clusters were modeled using a multivariate model. Like in Gries (2001), and unlike in Bouma (2013), the starting point was not to create the most optimal model, but to determine the effect of each variable from previous linguistic work. Verbal clusters were extracted semi-automatically from a part of the De Standaard CONDIV corpus, which contains texts from a Flemish newspaper spanning a time period of a few months. By choosing this part of the corpus, the author controlled for regional, register and diachronic variation.

As well as limiting the source data, De Sutter also limited cluster types. Only clusters in complement clauses, introduced by the complementizer *dat* 'that' (no main clause clusters or other subordinates), containing a participle main verb (no infinitival clusters), and only clusters with the non-modal auxiliaries *hebben* "to have", *zijn* "to be" and *worden* "to be" have been included. Only two-verb clusters were considered. These criteria resulted in 2.390 two-verb clusters, 1.601 (66.99%) of which were in the 2-1 order. The data were then annotated for 10 variables, which were mostly extracted from the data manually, and the operationalization of the variables was carefully considered in order to be as objective as possible. The statistical model is then used to reveal the contribution of each variable towards either an 2-1 or 1-2 order choice.

We have tested the same variables, which were identified in previous literature, on our data set as far as they could be operationalized. We will summarize the data and methodology that we used in our study in the next section, and compare it to that of De Sutter (2009). The results will be discussed in section 5.

4 Method and data

We create a multivariate logistic regression model for explaining verbal cluster variation much like that of De Sutter (2009), but based on automatically annotated data. In some cases where we had to choose between creating an optimal model and creating a comparable model, we chose comparability. For example, it is generally best practice to use the most frequent value of a categorical variable as the reference value to compare the effects of the other values to. However, to maintain comparability of the effect sizes of both models we chose to use the same reference values as De Sutter. To demonstrate some benefits of automatically annotated data, we did aim to include more types of two-verb verbal clusters that exhibit optionality, including the major constructions left out by De Sutter. As mentioned, the only constructions exhibiting optionality that we did not include, are the cluster with 'other' verbs instead of auxiliary or modal verbs.

We used the Lassy Large corpus as our source of automatically annotated Dutch language data (van Noord et al., 2013). It contains texts from various written sources annotated with full syntactic dependency trees. The sentences have been parsed automatically by the Alpino parser for Dutch (van Noord et al., 2006). This parser is currently the state of the art, and an evaluation over different types of text shows an average concept accuracy (in terms of correct named dependencies) of 86.52% (van Noord, 2009). For our main comparison, we used the Wikipedia part of this corpus, which consists of the entirety of the Dutch version of the freely editable online encyclopedia Wikipedia on the 4th of August, 2011 (about 145 million words). From this data, 411.623 two-verb verbal clusters were extracted (71.65% of which were in the 1-2 order). We chose to use this part of the corpus as we believe

it to be a good representation of 'average' standard Dutch. While Wikipedia texts have been written and edited by many speakers from different parts of the Dutch-speaking world, and probably by second language learners of Dutch as well, non-standard language is likely to be edited out by other editors. The accuracy of the Alpino parser on this text type was 88.38% in van Noord (2009), better than average but not as good as newspaper texts.

We do recognize that languages can not really be averaged, and a model based on such data will not be able to account for regional diversity or individual differences. Significant regional differences have been observed in the usage of verbal clusters (i.e. the data of Barbiers et al. (2006)), so it would be interesting, and possible, to study this using an automatically annotated corpus where authorship metadata is available. 'Region' could then be added as a variable to the model to explain some of the variation. In this study we won't address this, however, we address the issue of language diversity by comparing the results from the Wikipedia part of the corpus to a model created from the Europarl part of the corpus (containing European parliament texts) in section 5.1.

The verbal clusters were extracted from the corpus using XPath 2.0 queries via the DACT command line tools. These queries precisely define what constitutes a verbal cluster, and every word group matching one of these definitions was extracted. Contextual information necessary to determine the value of the independent variables of each cluster was extracted in a similar way.

In defining and operationalizing the variables discussed by De Sutter (2009), we were limited to the information available in the annotation of the Lassy Large corpus, at least without doing any manual annotation. Some variables had to be operationalized in a different way, or could not be extracted at all. We now briefly summarize our operationalizations in comparison to those of De Sutter. For a more detailed description and motivation for each of the variables, we refer to De Sutter (2009).

The variables we operationalized are listed in the results table (1). To operationalize the TYPE OF THE AUXILIARY VERB, De Sutter divided the auxiliary verbs up into five grammatical classes: *zijn* "to be" as a copulative verb, *zijn* as a passive auxiliary, *zijn* or *hebben* "to have" as temporal auxiliaries, *worden* "to be", and unclassifiable. He developed an algorithm to identify them, which is a complex three-stage pipeline. It is described in De Sutter (2005, p. 205-230), involving 5 syntactic, 5 morphological and 2 semantic criteria. We did not try to re-create it because we would prefer to work with readily available corpus resources as much as possible for methodological demonstration purposes. The algorithm is also not perfect, hence the 'unclassifiable' class. Instead, we categorized the auxiliary verbs at the lexical level. We did group the modal verbs together (which De Sutter did not include in his study), as they appeared to behave very similarly in preliminary checks.

MORPHOLOGICAL STRUCTURE OF THE MAIN VERB encodes whether the main verb is separable or not. Separable particle verbs such as 'wash up' are written as a single word in Dutch, unless the particle is separated from the verb, which may happen even if it interrupts a verbal cluster. LENGTH OF THE MIDDLE FIELD is simply the number of words in the clause before the verbal cluster. Next, there are two variables relating to the word before the verbal cluster. INFORMATION VALUE is operationalized as the openness of this word's class, for which there are three classes: **high**ly informational (nouns, verbs, numerals), **intermediate**ly informational (adjectives and adverbs) and **low** informational (pronouns, conjunctions and prepositions). INHERENCE refers to multi-word units (MWUs), which is a complex concept with no clear definition, but generally describes some sort of collocation of words. The corpus includes annotation on MWUs, and we make use of this annotation to decide whether the preverbal word is part of one.

EXTRAPOSED CONSTITUENTS are constituents that come after the verbal cluster, for which there are three ways to attach to the rest of the sentence: none (no constituent), attached to the main verb of the cluster, or attached somewhere higher up in the tree than the cluster (preverbally). We had to operationalize this variable differently. De Sutter made a distinction between adjuncts and complements, grouping adjuncts with 'none', but we could not extract this distinction from the corpus, hence the difference. The FREQUENCY OF THE MAIN VERB was estimated by counting the number of occurrences of the verb's root in the entire Lassy Large corpus. Lastly, we added two variables to distinguish the new cluster types discussed in section 2 that De Sutter didn't include: FINITENESS OF THE HEAD to mark infinitival clusters, and CONTEXT CLAUSE TYPE for subordinate versus main clause clusters.

There were also two variables that we could not include. Firstly, the DISTANCE BETWEEN ACCENTS, which relates to the hypothesis that order variation may occur to match the stress pattern of Dutch (Schutter, 1996). Our corpus does not contain stress or accent information, and we are not aware of a method for automatic annotation. This variable turned out to have almost no effect in De Sutter's model. Secondly, we left out SYNTACTIC PERSISTENCE, which refers to a priming effect of a previous construction on the next. We decided not to include this variable, as we are mainly using a corpus based on Wikipedia. We cannot make sure that the writer wrote or even read the previous verbal cluster in the text. This variable did have an effect in De Sutter's model (OR=3.28).

5 Results

Table 1 shows the comparison to the model of De Sutter (2009). The table lists all of the explanatory variables used in our logistic regression model, along with their effect size in our model and in the model of De Sutter. Based on these variables, the model can predict the dependent variable of verbal cluster word order, which is expressed as a binary variable representing either 1-2 or 2-1 word order.

For each explanatory variable in the table, one of the possible values was taken as the reference value, or baseline. The baselines were selected to be the same as those of De Sutter, in order to have comparable models. In the cases where variables had to be operationalized differently (where there are gaps in the table), the baselines are of course also different, though we tried to pick the most similar values as the reference. The effect size of each variable for both studies is given as an odds ratio. An odds ratio further from 1 indicates a stronger effect. Odds ratios > 1 indicate an association with the 1-2 word order, odds ratios < 1 indicate 2-1 order. An exception is the MAIN VERB FREQUENCY variable — it is a continuous variable, where an odds ratio cannot be interpreted in the same way. Instead, we show the β estimate of effect size (representing an effect on the dependent variable, cluster order, in terms of standard deviations), and its average standard error.

We did not perform statistical tests to assess whether the effect sizes of the two models differ significantly. Some differences are to be expected, considering the different data sources (Flemish newspapers versus Wikipedia) or other uncontrolled variables. We would mainly like to see whether the same categories show substantial effects relative to their reference values, and whether the effects are in the same direction (either the 1-2 or 2-1 order) in both models. In cases where variables were operationalized in different ways between the two studies, this is indicated by leaving the missing operationalizations blank. The reference value for each variable and study is listed as **1.00**, as the reference value can obviously not have an effect compared to itself, and 1 is the neutral value.

As a first observation for the results in table 1, we can see that the directions and size of the effects are generally similar, except where the variables had to be operationalized differently. For the variables that were operationalized the same — MORPHOLOGICAL STRUCTURE OF THE PARTICIPLE, LENGTH OF MIDDLE FIELD, INHERENCE and MAIN VERB FREQUENCY we observe similar relative effects as De Sutter (2009), and in the same directions. The INFORMATION VALUE of the preverbal constituent shows an effect in the same direction, but far smaller, and not in the same order — Intermediate is more strongly associated with the 1-2 order than High. Perhaps the larger sample size caused this. We have checked that it is not due to the additional cluster types — a model with only subordinate, finite, nonmodal clusters shows an even smaller information value effect size.

We do see differences in the two variables that had to be operationalized differently in our study. For the variable GRAMMATICAL RELATION OF EXTRAPOSITION TO HEAD we interestingly find a very strong effect, though the directions of the effects are reversed compared to de Sutter. This is a complex syntactical property, so we cannot guarantee that it was implemented in exactly the same way besides the noted difference. Either way, we can still conclude that this variable has a strong effect on verbal cluster order. The most striking difference is in the variable TYPE OF AUXILIARY. This is due to the available annotation — as discussed in the last section, a complex additional procedure would be needed to identify the grammatical classes of De Sutter. We still find somewhat of a lexical effect with our operationalization. The auxiliary verbs have a lot of grammatical ambiguity, so it is smaller than the grammatical effect found by De Sutter. We do note a strong tendency of clusters with modal verbs to occur in the 1-2 order. This confirms previous observations that modal clusters have a strong preference for the 1-2 order (Wurmbrand, 2006).

De Sutter does not provide a value of overall model fit that can be compared across models, however, we can look at the concordance index (c-index). This value is an indication of the predictive power of a model. A c-index of 0.5 corresponds with chance level, and 1 indicates perfect prediction. Like de Sutter, we report the c-index after 100 bootstrap repetitions to compensate for overfitting. He reports c = 0.803, and our model has a concordance index of 0.8635. However, it should be noted that these two c-indexes cannot be directly compared between different models, since the variables are somewhat different. These values do indicate that the models are good enough for prediction tasks. They are also similar to the c-score Bouma (2013) reported. We can furthermore look at the intercept of our model, which is 0.6035, and represents the odds of predicting an 1-2 outcome in the case where all the variables have their default value, an indication of the difficulty of the task. Clearly the model predicts better than that. However, it should be noted that this is not a typical predictive task. There is no 100% gold standard as in a parsing task for example, and both orders are grammatically correct. It might very well be that a large amount of the variation is random, depends on extralinguistic factors, or factors not captured by the annotation scheme. The focus here is mainly on finding out how much can be explained by the linguistic variables under discussion, and the effect sizes reported in table 1 are the main measure of that.

Variable	**Categories**	Odds ratio **De Sutter (2009)**	Odds ratio **This study**
TYPE OF AUXILIARY	Copular *zijn*	**1.00**	
	Auxiliary of time	18.30 ***	
	Passive *zijn*	7.82 ***	
	zijn		**1.00**
	worden	11.73 ***	1.19 ***
	hebben		2.19 ***
	Modal verb		132.42 ***
MORPHOLOGICAL STRUCTURE OF THE MAIN VERB	Non-separable	**1.00**	**1.00**
	Separable	3.87 ***	4.92 ***
LENGTH OF MIDDLE FIELD	0-2 words	**1.00**	**1.00**
	3-5 words	2.03 ***	2.42 ***
	6-8 words	2.29 ***	3.23 ***
	9-11 words	2.29 ***	3.34 ***
	12-14 words	2.57 **	3.33 ***
	>14 words	1.98	3.15 ***
INFORMATION VALUE	Low	**1.00**	**1.00**
	Intermediate	1.41	1.21 ***
	High	1.94 ***	1.11 ***
INHERENCE	No fixed expression	**1.00**	**1.00**
	Fixed expression	2.26 ***	2.10 ***
GRAMMATICAL RELATION OF EXTRAPOSITION TO HEAD	Adjunct/no extraposition	**1.00**	
	Complement of main verb	0.47 ***	
	Complement of preverbal head	1.21	
	No extraposition		**1.00**
	Comp/adj of main verb		51.44
	Comp/adj of preverbal head		0.44
MAIN VERB FREQUENCY		$\beta = 2.44^{E-06}$ ASE=7.74^{E-07}**	$\beta = 3.73^{E-08}$ ASE=1.05^{E-08}***
FINITENESS OF THE HEAD	Finite head		**1.00**
	Infinite head		0.03
CONTEXT CLAUSE TYPE	Subordinate clause		**1.00**
	Main clause		0.34

Table 1: Comparison of the size of the effect of the variables on verbal cluster variation for the two studies

One could make the objection that our model is not comparable to that of de Sutter, because it contains different constructions (main clause, infinitival and modal clusters). We controlled for these construction types with variables, but just to verify that this works, we also created a model that excludes all of these additional construction types. The observed effects were very similar to the full model. For reasons of space we cannot provide the entire table, but some of the effects are: *worden* = 2.34 (was 2.19 in the main model) *hebben* = 1.21 (was 1.19), SEPARABLE = 5.01 (was 4.92). We do note a lower c-score here of 0.7649, it appears more difficult to predict verbal cluster order when the clusters are all of the same type. This is somewhat lower than De Sutter's c = 0.803, likely due to the variables we could not include. More interestingly, we were also able to create models for specific construction types, i.e. main clause verbal clusters only, which has not been done before. For reasons of space we cannot list or elaborate on the results here, but we find that the same variables also affect main clause cluster variation. There are some differences in effect direction, mainly for the auxiliary verbs, which makes sense as it is a different construction.

5.1 Europarl corpus

In section 4 we discussed our choice for the Wikipedia part of the Lassy Large corpus. However, this corpus consists of other kinds of sources as well, and now that we have a highly automated way of building the model, it is relatively easy to test it on a different part of the corpus to see whether the same variables hold in a different domain of text. The Dutch Europarl part of the corpus consists of the translated proceedings of the European

Parliament. These texts have been written in a rather formal register by translators of the European Union. This part consists of 37 million words, from which 467.521 verbal clusters were extracted. Interestingly, that's 55.898 more clusters than were extracted from the larger (145 million words) Wikipedia corpus, indicating far more complex syntax in the Europarl corpus.

We find that 86.78% of the clusters is in the 1-2 order, a far higher percentage than is generally reported and higher than the Wikipedia corpus (71.65%). Higher proportions of 1-2 orders are generally associated with more formal registers and with editing guidelines — prescriptivists in the past have considered it to be the Dutch order, while the 2-1 order was nonstandard or German (Coussé et al., 2008). Again, we will not produce the entire table of effect sizes here — it suffices to say that all of the effect directions are the same, and the sizes are also very similar, even for variables that vary between constructions such as TYPE OF AUXILIARY. We may conclude from this that the previously discussed findings are not domain-specific. It is also a demonstration of the flexibility of using automatically annotated data.

6 Discussion

Using an automatically annotated corpus, we have shown that verbal cluster order variation is influenced by the various language-internal variables identified by De Sutter (2009) for all two-verb cluster types with order variation. While the (relative) importance of these variables for non-modal, finite clusters with a complementizer-marked subordinate clause was already established, this study shows that they apply to other types of two-verb clusters as well, even when testing on a larger and more varied dataset. We furthermore showed that these findings can be extended into another domain of text, and are not domain-specific.

Our main contribution is to show that using automatically annotated corpora is an excellent source for obtaining more data for language variation studies. Although some variables (stressed syllables) could not be measured due to a lack of automatically annotated data, advances in other language technology, such as word sense disambiguation, is likely to open up more possibilities for additional kinds of annotation to be applied to huge corpora automatically. It would also be possible to obtain more annotation by combining information from other sources. For example, to estimate syllable stress, one could consider scraping this information from phonetic transcriptions in dictionaries and adding that information to the corpus. Using these huge, automatically annotated resources seems like a natural extension of the recent trend of using large multivariate models, and allows the creation of explanatory models for uncommon linguistic constructions. We have also demonstrated the flexibility of the automatic approach — a model can easily be tested on a different dataset, provided that the right annotation is available.

De Sutter (2009) draws several conclusions, which could also be drawn from our study. Firstly, that the variation appears to be affected by 8 variables simultaneously (7 in our case), and can be predicted well enough by the model. Secondly, there are various methodological conclusions: "We have shown that syntactic variation research needs a rigorous quantitative, corpus-based approach ...". We can only add to this conclusion by stating that this rigorous definition of variables aids in automatic extraction of samples, which lets us retrieve all relevant constructions from large automatically annotated corpora, to the extent that the annotation allows. This, in turn, opens up options for more detailed analysis as outlined in this paper.

6.1 Linguistic interpretation

Even though we have found patterns in the variation and associations with variables that were hypothesized to be related to the phenomenon, this in itself is not an explanation of verbal cluster variation in terms of linguistic theory. We will briefly address this by referring to the hypothesis of De Sutter (2005), who states that the choice between 1-2 and 2-1 order may be related to processing difficulty. He assumes the 2-1 order to be easier to process because he considers the 1-2 order to be a prestige option, implying that the 2-1 order is the default. However, this explanation can go both ways. Recent evidence from child language acquisition supports a default 1-2 order (Meyer and Weerman, 2014). They theorize that children learn about verb raising (which forms clusters) when they acquire the 1-2 order. The most common view is that both 1-2 and 2-1 clusters come from verb raising (Evers, 1975), in which the main verb is moved up the syntactic tree to join the head. Before verb raising, the head verb is in final position, following the base Object-Verb order of Dutch. Raising the verb to form a 2-1 cluster is therefore a vacuous movement, the surface order will be the same as the underlying structure (the head comes last), and provides no evidence of any sort of special verb raising mechanism to child learners of the language. On the other hand, raising the verb to the right to create a 1-2 order violates the base word order of Dutch, and is therefore more straightforward to notice and learn. 2-1 orders can simply be interpreted as Object-Verb orders, until the learner figures out the mechanism of verb raising from the 1-2 order evidence (Meyer and Weerman, 2014). In this theory, 1-2 orders would be the earliest form of verb raising, and therefore more entrenched and easier to process.

Either way, in linguistic contexts that are more difficult to process, speakers are expected to be more likely to use the more entrenched order that is easier to process, whichever it may be. The model may be inconclusive on this matter. De Sutter argues for the 2-1 order by looking at the MAIN VERB FREQUENCY variable, stating that higher-frequency items are easier to access, and the model shows that higher-frequency words are more associated with the 2-1 order. Here, a **less** difficult context is associated with the 2-1 order. However, we can make the opposite argument for the variable LENGTH OF MIDDLE FIELD. It seems plausible that longer clauses are **more** difficult to process, and longer clauses are also associated with the 2-1 order. To invoke the processing hypothesis here, one would have to assume that the 1-2 order is the default and easier to process. Given these two opposites predicted by Meyer and Weerman (2014) and De Sutter (2005), it would be interesting to look for additional variables that are related to processing difficulty and can be extracted from the corpus automatically, such as syntactic complexity. These can then be added to the model to test which order occurs in contexts with higher processing load.

6.2 Future work

We consider several other directions for future work. The model can be extended to other domains that have been discussed in the literature, such as spoken language data from the Corpus of Spoken Dutch which uses a similar annotation scheme. As discussed earlier, regional variation is an interesting topic that could also be modeled using large amounts of data, for example by using the SoNaR corpus which includes metadata on the authors of many of its texts and adding a region variable to the model. We have yet to investigate clusters with more than two verbs, to which the automatic approach is uniquely suited. Larger verbal clusters are less frequent, and thus the best place to find rare constructions is in the largest available corpus. Now that large samples of data are easily available, a method such as collostructional analysis may be used to explore the association between particular main verbs and the 1-2/2-1 order, providing more detail on possible semantic factors.

In this study, we have aimed to follow the methodology of De Sutter (2009) closely, but this also had several downsides. It would also be possible to aim for creating the best possible model over the dataset, though differences that might arise from this (for example, different reference levels) would then make comparisons more difficult. This would allow testing of some potential methodological improvements. Multilevel modeling may be used as in Bouma (2013) to model the effects of individual lexical items. A form of Principal Component Analysis could be applied to generalize over the variables and try to reduce their number. Choosing verbal cluster order could also be viewed as a two-class classification problem, for which many other modeling methods exist.

Acknowledgements

We thank Gosse Bouma and the anonymous reviewers for their useful comments, and Gertjan van Noord and Daniël de Kok for helping to provide efficient access to the corpus data.

References

Mona Arfs. *Rood of groen? De interne woordvolgorde in tweeledige werkwoordelijke eindgroepen met een voltooid deelwoord en een hulpwerkwoord in bijzinnen*. Göteborg University, 2007.

Sjef Barbiers, H Bennis, G De Vogelaer, M Devos, and M van der Ham. Dynamische syntactische atlas van de Nederlandse dialecten (DynaSAND). *Amsterdam, Meertens Instituut. URL: http://www.meertens.knaw.nl/sand*, 2006.

Gosse Bouma. Om-omission in Dutch verbal complements. *Manuscript in preparation*, 2013.

Joan Bresnan, Anna Cueni, Tatiana Nikitina, R Harald Baayen, et al. Predicting the dative alternation. *Cognitive foundations of interpretation*, pages 69–94, 2007.

Evie Coussé. *Motivaties voor volgordevariatie. Een diachrone studie van werkwoordvolgorde in het Nederlands.* Universiteit Gent, 2008.

Evie Coussé, Mona Arfs, and Gert De Sutter. Variabele werkwoordsvolgorde in de Nederlandse werkwoordelijke eindgroep. Een taalgebruiksgebaseerd perspectief op de synchronie en diachronie van de zgn. rode en groene woordvolgorde. *Taal aan den lijve. Het gebruik van corpora in taalkundig onderzoek en taalonderwijs*, pages 29–47, 2008.

Gert De Sutter. *Rood, groen, corpus! Een taalgebruiksgebaseerde analyse van woordvolgordevariatie in tweeledige werkwoordelijke eindgroepen*. University of Leuven: PhD thesis, 2005.

Gert De Sutter. Towards a multivariate model of grammar: The case of word order variation in Dutch clause final verb clusters. *Describing and modeling variation in grammar*, 204:225–254, 2009.

Hans Den Besten and Jerold A Edmondson. The verbal complex in continental West Germanic. *On the formal syntax of the Westgermania*, 3:155–216, 1983.

Arnold Evers. *The transformational cycle in Dutch and German*, volume 75. Indiana University Linguistics Club Bloomington, 1975.

Arnold Evers. Verbal clusters and cluster creepers. *Amsterdam studies in the theory and history of linguistic science, Series 4*, pages 43–90, 2003.

Stefan T Gries. A multifactorial analysis of syntactic variation: particle movement revisited. *Journal of quantitative linguistics*, 8(1):33–50, 2001.

Liliane Haegeman and Henk van Riemsdijk. Verb projection raising, scope, and the typology of rules affecting verbs. *Linguistic inquiry*, pages 417–466, 1986.

Hans Martin Lehmann and Gerold Schneider. Syntactic variation and lexical preference in the dative-shift alternation. *Language and Computers*, 75(1):65–75, 2012.

Caitlin Meyer and Fred Weerman. Cracking the cluster: The acquisition of verb raising in dutch. *Manuscript in preparation*, 2014.

G de Schutter. De volgorde in tweeledige werkwoordelijke eindgroepen met voltooid deelwoord in spreek-en schrijftaal. *Nederlandse taalkunde*, 1:207–220, 1996.

Jan Stroop. Systeem in gesproken werkwoordsgroepen. *Taal en tongval*, 22:128–147, 1970.

Jan Stroop. Twee- en meerledige werkwoordsgroepen in gesproken Nederlands. *Fons verborum*, pages 459–469, 2009.

Gertjan van Noord. Huge parsed corpora in lassy. *Proceedings of TLT7. LOT, Groningen, The Netherlands*, 2009.

Gertjan van Noord and Gosse Bouma. Parsed corpora for linguistics. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 33–39. Association for Computational Linguistics, 2009.

Gertjan van Noord, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. Large scale syntactic annotation of written Dutch: Lassy. In Peter Spyns and Jan Odijk, editors, *Essential Speech and Language Technology for Dutch*, Theory and Applications of Natural Language Processing, pages 147–164. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-30909-0. doi: 10.1007/978-3-642-30910-6_9. URL `http://dx.doi.org/10.1007/978-3-642-30910-6_9`.

Gertjan van Noord et al. At last parsing is now operational. In *TALN06. Verbum Ex Machina. Actes de la 13e conference sur le traitement automatique des langues naturelles*, pages 20–42, 2006.

Susi Wurmbrand. West Germanic verb clusters: The empirical domain. *Verb clusters: A study of Hungarian, German, and Dutch*, pages 43–85, 2004.

Susi Wurmbrand. Verb clusters, verb raising, and restructuring. *The Blackwell companion to syntax*, pages 229–343, 2006.

Shalom Zuckerman. *The acquisition of "optional" movement*. PhD thesis, University Library Groningen [Host], 2001.

Jan-Wouter Zwart. Verb clusters in continental West Germanic dialects. *Microparametric syntax and dialect variation*, pages 229–258, 1996.

Empirical analysis of exploiting review helpfulness for extractive summarization of online reviews

Wenting Xiong
University of Pittsburgh
Department of Computer Science
wex12@cs.pitt.edu

Diane Litman
University of Pittsburgh
Department of Computer Science & LRDC
litman@cs.pitt.edu

Abstract

We propose a novel unsupervised extractive approach for summarizing online reviews by exploiting review helpfulness ratings. In addition to using the helpfulness ratings for review-level filtering, we suggest using them as the supervision of a topic model for sentence-level content scoring. The proposed method is metadata-driven, requiring no human annotation, and generalizable to different kinds of online reviews. Our experiment based on a widely used multi-document summarization framework shows that our helpfulness-guided review summarizers significantly outperform a traditional content-based summarizer in both human evaluation and automated evaluation.

1 Introduction

Multi-document summarization has great potential in online reviews, as manually reading comments provided by other users is time consuming if not impossible. While extractive techniques are generally preferred over abstractive ones (as abstraction can introduce disfluency), existing extractive summarizers are either supervised or based on heuristics of certain desired characteristics of the summarization result (e.g., maximize n-gram coverage (Nenkova and Vanderwende, 2005), etc.). However, when it comes to online reviews, there are problems with both approaches: the first one requires manual annotation and is thus less generalizable; the second one might not capture the salient information in reviews from different *domains* (*camera* reviews vs. *movie* reviews), because the heuristics are designed for traditional genres (e.g., news articles) while the utility of reviews might vary with the review domain.

We propose to exploit review metadata, that is *review helpfulness ratings*[1], to facilitate review summarization. Because this is user-provided feedback on review helpfulness which naturally reflects users' interest in online review exploration, our approach captures domain-dependent salient information adaptively. Furthermore, as this metadata is widely available online (e.g., Amazon.com, IMDB.com)[2], our approach is unsupervised in the sense that no manual annotation is needed for summarization purposes. Therefore, we hypothesize that summarizers guided by review helpfulness will outperform systems based on textual features/heuristics designed for traditional genres. To build such helpfulness-guided summarizers, we introduce review helpfulness during content selection in two ways: 1) using the review-level helpfulness ratings directly to filter out unhelpful reviews, 2) using sentence-level helpfulness features derived from review-level helpfulness ratings for sentence scoring. As we observe in our pilot study that supervised LDA (sLDA) (Blei and McAuliffe, 2010) trained with review helpfulness ratings has potential in differentiating review helpfulness at the sentence level, we develop features based on the inferred hidden topics from sLDA to capture the helpfulness of a review sentence for summarization purposes. We implement our helpfulness-guided review summarizers based on an widely used open-source multi-document extractive summarization framework (MEAD (Radev et al., 2004)). Both human and

[1]This is the percentage of readers who found the review to be helpful (Kim et al., 2006).

[2]If it is not available, the review helpfulness can be assessed fully automatically (Kim et al., 2006; Liu et al., 2008).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1985–1995, Dublin, Ireland, August 23-29 2014.

automated evaluations show that our helpfulness-guided summarizers outperform a strong baseline that MEAD provides across multiple review domains. Further analysis on the human summaries shows that some effective heuristics proposed for traditional genres might not work well for online reviews, which indirectly supports our use of review metadata as supervision. The presented work also extrinsically demonstrates that the helpfulness-related topics learned from the review-level supervision can capture review helpfulness at the sentence-level.

2 Related Work

In multi-document extractive summarization, various unsupervised approaches have been proposed to avoid manual annotation. A key task in extractive summarization is to identify important text units. Prior successful extractive summarizers score a sentence based on n-grams within the sentence: by the word frequency (Nenkova and Vanderwende, 2005), bigram coverage (Gillick and Favre, 2009), topic signatures (Lin and Hovy, 2000) or latent topic distribution of the sentence (Haghighi and Vanderwende, 2009), which all aim to capture the "core" content of the text input. Other approaches regard the n-gram distribution difference (e.g., Kullback-Lieber (KL) divergence) between the input documents and the summary (Lin et al., 2006), or based on a graph-representation of the document content (Erkan and Radev, 2004; Leskovec13 et al., 2005), with an implicit goal to maximize the output representativeness. In comparison, while our approach follows the same extractive summarization paradigm, it is metadata driven, identifying important text units through the guidance of user-provided review helpfulness assessment.

When it comes to online reviews, the desired characteristics of a review summary are different from traditional text genres (e.g., news articles), and could vary from one review domain to another. Thus different review summarizers have been proposed to focus on different desired properties of review summaries, primarily based on opinion mining and sentiment analysis (Carenini et al., 2006; Lerman et al., 2009; Lerman and McDonald, 2009; Kim and Zhai, 2009). Here the desired property varies from the coverage of product aspects (Carenini et al., 2006; Lerman et al., 2009) to the degree of agreement on aspect-specific sentiment (Lerman et al., 2009; Lerman and McDonald, 2009; Kim and Zhai, 2009). While there is a large overlap between text summarization and review opinion mining, most work focuses on sentiment-oriented aspect extraction and the output is usually a set of topics words plus their representative text units (Hu and Liu, 2004; Zhuang et al., 2006). However, such a topic-based summarization framework is beyond the focus of our work, as we aim to adapt traditional extractive techniques to the review domain by introducing review helpfulness ratings as guidance.

In this paper, we utilize review helpfulness via using sLDA. The idea of using sLDA in text summarization is not new. However, the model is previously applied at the sentence level (Li and Li, 2012), which requires human annotation on the sentence importance. In comparison, our use of sLDA is at the document (review) level, using existing metadata of the document (review helpfulness ratings) as the supervision, and thus requiring no annotation at all. With respect to the use of review helpfulness ratings, early work of review summarization (Liu et al., 2007) only consider it as a filtering criteria during input preprocessing. Other researchers use it as the gold-standard for automated review helpfulness prediction, a predictor of product sales (Ghose and Ipeirotis, 2011), a measurement of reviewers' authority in social network analysis (Lu et al., 2010), etc.

3 Helpfulness features for sentence scoring

While the most straightforward way to utilize review helpfulness for content selection is through filtering (Liu et al., 2007) (further discussed in Section 4.3), we also propose to take into account review helpfulness during sentence scoring by learning helpfulness-related review topics in advance. Because sLDA learns the utility of the topics for predicting review-level helpfulness ratings (decomposing review helpfulness ratings by topics), we develop novel features (*rHelpSum* and *sHelpSum*) based on the inferred topics of the words in a sentence to capture its helpfulness in various perspectives. We later use them for sentence scoring in a helpfulness-guided summarizer (Section 4.3).

Compared with LDA (Blei et al., 2003), sLDA (Blei and McAuliffe, 2010) introduces a response

variable $y_i \in Y$ to each document D_i during topic discovery. The model not only learns the topic assignment $z_{1:N}$ for words $w_{1:N}$ in D_i, it also learns a function from the posterior distribution of z in D to Y. When Y is the review-level helpfulness gold-standard, the model leans a set of topics predictive of review helpfulness, as well as the utility of z in predicting review helpfulness y_i, denoted as η. (Both z and η are K-dimensional.)

At each inference step, sLDA assigns a topic ID to each word in every review. $z_l = k$ means that the topic ID for word at position l in sentence s is k. Given the topic assignments $z_{1:L}$ to words $w_{1:L}$ in a review sentence s, we estimate the contribution of s to the helpfulness of the review it belongs to (Formula 1), as well as the average topic importance in s (Formula 2). While $rHelpSum$ is sensitive to the review length, $sHelpSum$ is sensitive to the sentence length.

$$rHelpSum(s) = \frac{1}{N} \sum_{l=1}^{l=L} \sum_{k} \eta_k p(z_l = k) \tag{1}$$

$$sHelpSum(s) = \frac{1}{L} \sum_{l=1}^{l=L} \sum_{k} \eta_k p(z_l = k) \tag{2}$$

As the topic assignment in each inference iteration might not be the same, Riedl and Biemann (Riedl and Biemann, 2012) proposed the *mode* method in their application of LDA for text segmentation – use the most frequently assigned topic for each word in all iterations as the final topic assignment – to address the instability issue. Inspired by their idea, we also use the *mode* method to infer the topic assignment in our task, but only apply the *mode* method to the last 10 iterations, because the topic distribution might not be well learned at the beginning.

4 Experimental setup

To investigate the utility of exploiting user-provided review helpfulness ratings for content selection in extractive summarization, we develop two helpfulness-guided summarizers based on the MEAD framework (HelpfulFilter and HelpfulSum). We compare our systems' performance against a strong unsupervised extractive summarizer that MEAD supports as our baseline (MEAD+LexRank). To focus on sentence scoring only, we use the same MEAD word-based MMR (Maximal Marginal Relevance) reranker (Carbonell and Goldstein, 1998) for all summarizers, and set the length of the output to be 200 words.

4.1 Data

Our data consists of two kinds of online reviews: 4050 Amazon camera reviews provided by Jindal and Liu (2008) and 280 IMDB movie reviews that we collected by ourselves. Both corpora were used in our prior work of automatically predicting review helpfulness, in which every review has at least three helpfulness votes. On average, the helpfulness of camera reviews is .80 and that of movie reviews is .74.

Summarization test sets. Because the proposed approach method is purely unsupervised, and we do not optimize our summarization parameters during learning, we evaluate our approach based on a subset of review items directly: we randomly sample 18 reviews for each review item (a camera or movie) and randomly select 3 items for each review domain. In total there are 6 summarization test sets (3 items $\times$ 2 domains), where each contains 18 reviews to be summarized (i.e. "summarizing 18 camera reviews for Nikon D3200"). In the summarization test sets, the average number of sentences per review is 9 for camera reviews, and 18 for movie reviews; the average number of words per sentence in the camera reviews and movie reviews are 25 and 27, respectively.

4.2 sLDA training

We implement sLDA based on the topic modeling framework of Mallet (McCallum, 2002) using 20 topics ($K = 20$) and the best hyper-parameters (topic distribution priors α and word distribution priors

β) that we learned in our pilot study on LDA. [3]

Since our summarization approach is unsupervised, we learn the topic assignment for each review word using the corresponding sLDA model trained on all reviews of that domain (4050 reviews for camera and 280 reviews for movie).[4]

4.3 Three summarizers

Baseline (MEAD+LexRank): The default feature set of MEAD includes *Position*, *Length*, and *Centroid*. Here *Length* is a word-count threshold, which gives score 0 to sentences shorter than the threshold. As we observe that short review sentences sometimes can be very informative as well (e.g., "This camera is so amazing!", "The best film I have ever seen!"), we adjust $Length$ to 5 from its default value 9. MEAD also provides scripts to compute *LexRank* (Erkan and Radev, 2004), which is a more advanced feature using graph-based algorithm for computing relative importance of textual units. We supplement the default feature set with *LexRank* to get the best summarizer from MEAD, yielding the sentence scoring function $F_{baseline}(s)$, in which s is a given sentence and all features are assigned equal weights (same as in the other two summarizers).

$$F_{baseline}(s) = \begin{cases} Position + Centroid + LexRank & \text{if } Length \geq 5 \\ 0 & \text{if } Length < 5 \end{cases} \quad (3)$$

HelpfulFilter: This summarizer is a direct extension of the baseline, which considers review-level helpfulness ratings ($hRating$) as an additional filtering criteria in its sentence scoring function $F_{HelpfulFilter}$. (In our study, we omit the automated prediction (Kim et al., 2006; Liu et al., 2008) and filter reviews by their helpfulness gold-standard directly.) We set the cutting threshold to be the average helpfulness rating of all the reviews that we used to train the topic model for the corresponding domain ($hRatingAve(domain)$).

$$F_{HelpfulFilter}(s) = \begin{cases} F_{baseline}(s) & \text{if } hRating(s) \geq hRatingAve(domain) \\ 0 & \text{if } hRating(s) < hRatingAve(domain) \end{cases} \quad (4)$$

HelpfulSum: To isolate the contribution of review helpfulness, the second summarizer only uses helpfulness related features in its sentence scoring function $F_{HelpfulSum}$. The features are ***rHelpSum*** – the contribution of a sentence to the overall helpfulness of its corresponding review, ***sHelpSum*** – the average topic weight in a sentence for predicting the overall helpfulness of the review (Formula 1 and 2), plus $hRating$ for filtering. Note that there is no overlap between features used in the baseline and HelpfulSum, as we wonder if the helpfulness information alone is good enough for discovering salient review sentences.

$$F_{HelpfulSum}(s) = \begin{cases} rHelpSum(s) + sHelpSum(s) & \text{if } hRating(s) \geq hRatingAve(domain) \\ 0 & \text{if } hRating(s) < hRatingAve(domain) \end{cases} \quad (5)$$

5 Evaluation

For evaluation, we will first present our human evaluation user study and then present the automated evaluation result based on human summaries collected from the user study.

[3] In our pilot study, we experimented with various hyper-parameter settings, and trained the model with 100 sampling iterations in both the **E**stimation and the **M**aximization steps. As we found the best results are more likely to be achieved when $\alpha = 0.5$, $\beta = 0.1$, we use this setting to train the sLDA model in our summarization experiment.

[4] In practice, this means that we need to (re)train the topic model after given the summarization test set.

5.1 Human evaluation

The goal of our human evaluation is to compare the effectiveness of 1) using a traditional content selection method (MEAD+LexRank), 2) using the traditional method enhanced by review-level helpfulness filtering (HelpfulFilter), and 3) using sentence helpfulness features estimated by sLDA plus review-level helpfulness filtering (HelpfulSum) for building an extractive multi-document summarization system for online reviews. Therefore, we use a within-subject design in our user study for each review domain, considering the *summarizer* as the main effect on human evaluation results.

The user study is carried out in the form of online surveys (one survey per domain) hosted by Quadrics. In total, 36 valid users participated in our online-surveys.[5] We randomly assigned 18 of them to the camera reviews, and the rest 18 to the movie reviews.

5.1.1 Experimental procedures

Each online survey contains three summarization sets. The human evaluation on each one is taken in three steps:

Step 1: We first require users to perform **manual summarization**, by selecting 10 sentences from the input reviews (displayed in random order for each visit). This ensures that users are familiar with the input text so that they can have fair judgement on machine-generated results. To help users select the sentences, we provide an introductory scenario at the beginning of the survey to illustrate the potential application in accordance with the domain (e.g., Figure 1).

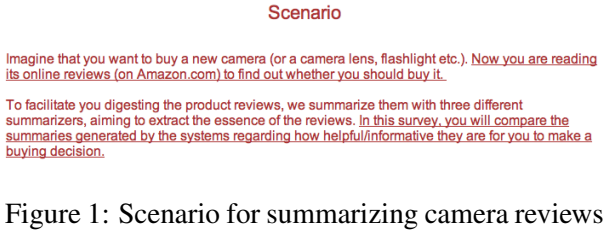


Figure 1: Scenario for summarizing camera reviews

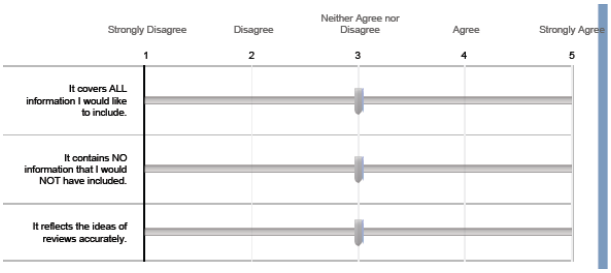


Figure 2: Content evaluation

Step 2: We then ask users to perform **pairwise comparison** on summaries generated by the three systems. The three pairs are generated in random order; and the left-or-right display position (in Figure 3) of the two summaries in each pair is also randomly selected. Here we use the same 5-level preference ratings used in (Lerman et al., 2009), and translate them into integers from -2 to 2 in our result analysis.
Step 3: Finally, we ask users to evaluate the three summaries in isolation regarding the summary quality in three content-related aspects: *recall*, *precision* and *accuracy* (top, middle and bottom in Figure 2, respectively), which were used in (Carenini et al., 2006). In this **content evaluation**, the three summaries are randomly visited and the users rate the proposed statements (one for each aspect) on a 5-point scale.

5.1.2 Results

Pairwise comparison. We use a mixed linear model to analyze user preference over the three summary pairs separately, in which "summarizer" is a between-subject factor, "review item" is the repeated factor, and "user" is a random effect. Results are summarized in Table 1. (Positive preference ratings on "A over B" means A is preferred over B; negative ratings means B is preferred over A.) As we can see, **HelpfulSum** is the best: it is consistently preferred over the other two summarizers across domains and the preference is significant throughout conditions except when compared with HelpfulFilter on movie reviews. **HelpfulFilter** is significantly preferred over the baseline (MEAD+LexRank) for movie reviews, while it does not outperform the baseline on camera reviews. A further look at the compression rate (cRate) of the three systems (Table 2) shows that on average HelpfulFilter generates shortest summaries

[5] All participants are older than eighteen, recruited via university mailing lists, on-campus flyers as well as social networks online. While we also considered educational peer reviews as a third domain, about half of the participants dropped out in the middle of the survey. Thus we only consider the two e-commerce domains in this paper.

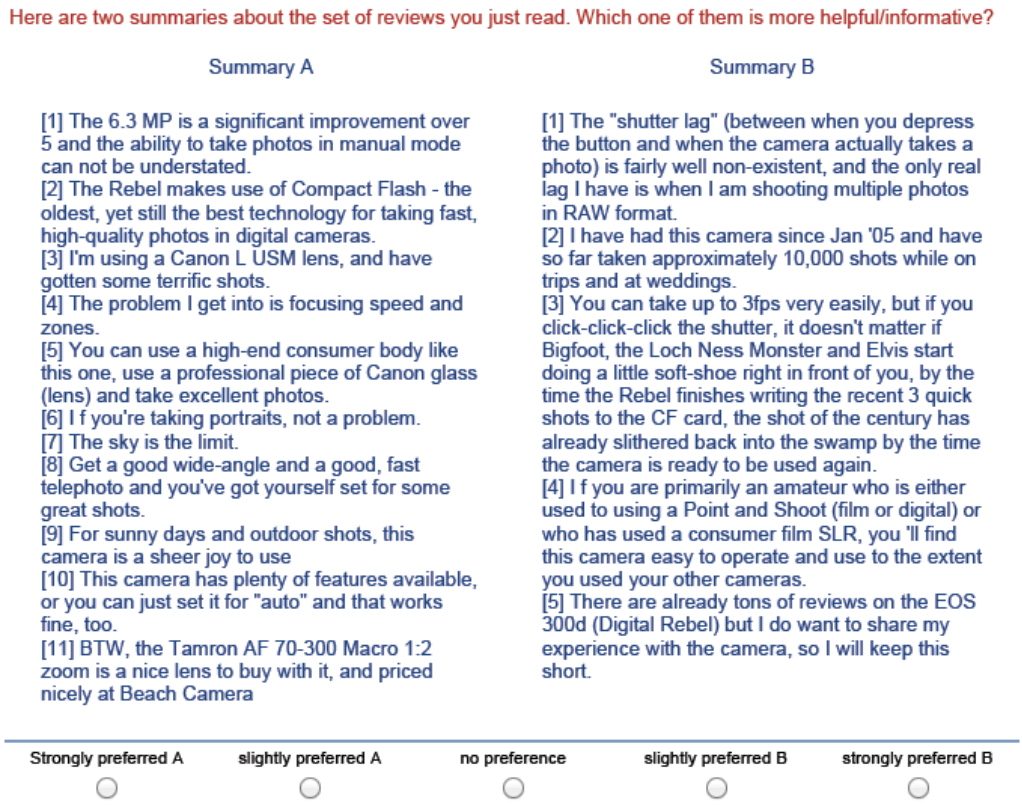


Figure 3: Example of pairwise comparison for summarizing camera reviews (left:HelpfulSum, right: the baseline).

among the three summarizers on camera reviews[6], which makes it naturally harder for **HelpfulFilter** to beat the other two (Napoles et al., 2011).

Pair	Domain	Est. Mean	Sig.
HelpfulFilter	Camera	-.602	.001
over MEAD+LexRank	Movie	.621	.000
HelpfulSum	Camera	.424	.011
over MEAD+LexRank	Movie	.601	.000
HelpfulSum	Camera	1.18	.000
over HelpfulFilter	Movie	.160	.310

Table 1: Mixed-model analysis of user preference ratings in pairwise comparison across domains. Confidence interval = 95%. The preference rating is ranged from -2 to 2.

Summarizer	Camera	Movie
MEAD+LexRank	6.07%	2.64%
HelpfulFilter	3.25%	2.39%
HelpfulSum	5.94%	2.69%
Human (Ave.)	6.11%	2.94%

Table 2: Compression rate of the three systems across domains.

Content evaluation. We summarize the average quality ratings (Figure 2) received by each summarizer across review items and users for each review domain in Table 3. We carry out paired T-tests for every pair of summarizers on each quality metric. While no significant difference is found among the three summarizers on any quality metric for movie reviews, there are differences for camera reviews. In terms of both accuracy and recall, HelpfulSum is significantly better than HelpfulFilter (p=.008 for accuracy, p=.034 for recall) and the baseline is significantly better than HelpfulFilter (p=.005 for accuracy, p=.005 for recall), but there is no difference between HelpfulSum and the baseline. For precision, no significant

[6]While we limit the summarization output to be 200 words in MEAD, as the content selection is at the sentence level, the summaries can have different number of words in practice. Considering that word-based MMR controls the redundancy in the selected summary sentences ($\lambda = 0.5$ as suggested), there might be enough content to select using $F_{HelpfulFilter}$.

difference is observed in either domain.

Summarizer	Camera			Movie		
Metric	Precision	Recall	Accuracy	Precision	Recall	Accuracy
MEAD+LexRank	2.63	**3.24**	3.57	2.50	2.59	2.93
HelpfulFilter	**2.78**	2.74	3.11	2.44	2.61	2.96
HelpfulSum	2.41	3.19	**3.69**	**2.52**	**2.67**	**3.02**

Table 3: Human ratings for content evaluation. The best result on each metric is bolded for every review domain (the higher the better).

With respect to pairwise evaluation, content evaluation yields consistent results on camera reviews between HelpfulFilter vs. the baseline and HelpfulSum vs. HelpfulFilter. However, only pairwise comparison (preference ratings) shows significant difference between HelpfulSum vs. the baseline and the difference in the summarizers' performance on movie reviews. This confirms that pairwise comparison is more suitable than content evaluation for human evaluation (Lerman et al., 2009).

5.2 Automated evaluation based on ROUGE metrics

Although human evaluation is generally preferred over automated metrics for summarization evaluation, we report our automated evaluation results based on ROUGE scores (Lin, 2004) using references collected from the user study. For each summarization test set, we have 3 machine generated summaries and 18 human summaries. We compute the ROUGE scores in a leave-1-out fashion: for each machine generated summary, we compare it against 17 out of the 18 human summaries and report the score average across the 17 runs; for each human summary, we compute the score using the other 17 as references, and report the average human summarization performance.

Evaluation results are summarized in Table 4 and Table 5, in which we report the F-measure for R-1 (unigram), R-2 (bigram) and R-SU4 (skip-bigram with maximum gap length of 4)[7], following the convention in the summarization community. Here we observe slightly different results with respect to human evaluation: for camera reviews, no significant result is observed, while HelpfulSum achieves the best R-1 score and HelpfulFilter works best regarding R-2 and R-SU4. In both cases the baseline is never the best. For movie reviews, HelpfulSum significantly outperforms the other summarizers on all ROUGE measurements, and the improvement is over 100% on R-2 and R-SU4, almost the same as human does. This is consistent with the result of pairwise comparison in that HelpfulSum works better than both HelpfulFilter and the baseline on movie reviews.

Summarizer	R-1	R-2	R-SU4
MEAD+LexRank	.333	.117	.110
HelpfulFilter	.346	**.121**	**.111**
HelpfulSum	**.350**	.110	.101
Human	.360	.138	.126

Table 4: ROUGE evaluation on camera reviews

Summarizer	R-1	R-2	R-SU4
MEAD+LexRank	.281	.044	.047
HelpfulFilter	.273	.040	.041
HelpfulSum	**.325**	**.095**	**.090**
Human	.339	.093	.093

Table 5: ROUGE evaluation on movie reviews

6 Human summary analysis

To get a comprehensive understanding of the challenges in extractive review summarization, we analyze the agreement in human summaries collected in our user study at different levels of granularity, regarding heuristics that are widely used in existing extractive summarizers.

Average word/sentence counts. Figure 4 illustrates the trend of average number of words and sentences shared by different number of users across review items for each domain. As it shows, no sentence is

[7]Because ROUGE requires all summaries to have equal length (word counts), we only consider the first 100 words in every summary.

agreed by over 10 users, which suggests that it is hard to make humans agree on the informativeness of review sentences.

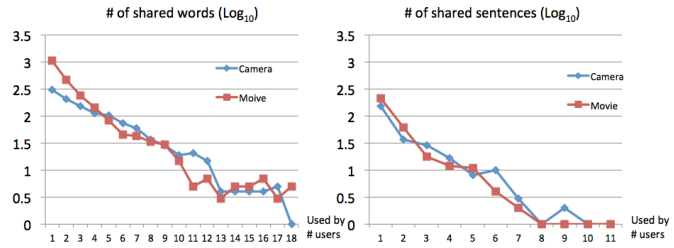


Figure 4: Average number of words (w) and sentences (s) in agreed human summaries

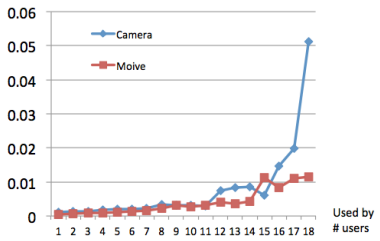


Figure 5: Average probability of words used in human summaries

Word frequency. We then compute the average probability of word (in the input) used by different number of human summarizers to see if the word frequency pattern found in news articles (words that human summarizers agreed to use in their summaries are of high frequency in the input text (Nenkova and Vanderwende, 2005)) holds for online reviews. Figure 5 confirms this. However, the average word probability is below 0.01 in those shared by 14 out of 18 summaries[8] ; the flatness of the curve seems to suggest that word frequency alone is not enough for capturing the salient information in input reviews.

KL-divergence. Another widely used heuristic in multi-document summarization is minimizing the distance of unigram distribution between the summary and the input text (Lin et al., 2006). We wonder if this applies to online review summarization. For each testing set, we group review sentences by the number of users who selected them in their summaries, and compute the KL-divergence (KLD) between each sentence group and the input. The average KL-divergence of each group across review items are visualized in Figure 6, showing that this intuition is incorrect for our review domains. Actually, the pattern is quite the opposite, especially when the number of users who share the sentences is less than 8. Thus traditional methods that aim to minimize KL-divergence might not work well for online reviews.

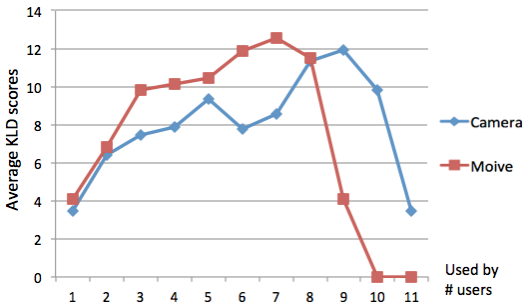


Figure 6: Average KL-Divergence between input and sentences used in human summaries

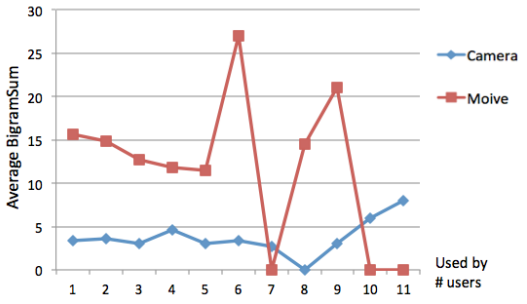


Figure 7: Average BigramSum of sentences used in human summaries

Bigram coverage. Recent studies proposed a simple but effective criteria for extractive summarization based on bigram coverage (Nenkova and Vanderwende, 2005; Gillick and Favre, 2009). The coverage of a given bigram in a summary is defined as the number of input documents the bigram appears in, and presumably good summaries should have larger sum of bigram coverage (BigramSum). However, as shown in Figure 7, this criteria might not work well in our case either. For instance, the BigramSum of the sentences that are shared by 3 human judges is smaller than those shared by 1 or 2 judges.

[8]The average probability of words used by all 4 human summarizers are 0.01 across the 30 DUC03 sets (Nenkova and Vanderwende, 2005).

7 Conclusion and future work

We propose a novel unsupervised extractive approach for summarizing online reviews by exploiting review helpfulness ratings for content selection. We demonstrate that the helpfulness metadata can not only be directly used for review-level filtering, but also be used as the supervision of sLDA for sentence scoring. This approach leverages the existing metadata of online reviews, requiring no annotation and generalizable to multiple review domains. Our experiment based on the MEAD framework shows that HelpfulFilter is preferred over the baseline (MEAD+LexRank) on camera reviews in human evaluation. HelpfulSum, which utilizes review helpfulness at both the review and sentence level, significantly outperforms the baseline in both human and automated evaluation. Our analysis on the collected human summaries reveals the limitation of traditional summarization heuristics (proposed for news articles) for being used in review domains.

In this study, we consider the ground truth of review helpfulness as the percentage of helpful votes over all votes, where the helpfulness votes could be biased in various ways (Danescu-Niculescu-Mizil et al., 2009). In the future, we would like to explore more sophisticated models of review helpfulness to eliminate such biases, or even automatic review helpfulness predictions based on just review text. We also would like to build a fully automated summarizer by replacing the review helpfulness gold-standard with automated predictions as the filtering criteria. Given the collected human summaries, we will experiment with different feature combinations for sentence scoring and we will compare our helpfulness features with other content features as well. Finally, we want to further analyze the impact of the number of human judges on our automated evaluation results based on ROUGE scores.

Acknowledgements

This research is supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A120370 to the University of Pittsburgh. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute or the U.S. Department of Education. We thank Dr. Jingtao Wang and Dr. Christian Schunn for giving us suggestions on the user study design.

References

David M Blei and Jon D McAuliffe. 2010. Supervised topic models. *arXiv preprint arXiv:1003.0783*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM.

Giuseppe Carenini, Raymond T Ng, and Adam Pauls. 2006. Multi-document summarization of evaluative text. In *In Proceedings of the 11st Conference of the European Chapter of the Association for Computational Linguistics*.

Cristian Danescu-Niculescu-Mizil, Gueorgi Kossinets, Jon Kleinber g, and Lillian Lee. 2009. How opinions are received by online communities: A case study on Amazon .com helpfulness votes. In *Proceedings of the 18th International Conference on World Wide Web*, pages 141–150.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.(JAIR)*, 22(1):457–479.

Anindya Ghose and Panagiotis G Ipeirotis. 2011. Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics. *IEEE Transactions on Knowledge and Data Engineering*, 23(10):1498–1512.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pages 10–18. Association for Computational Linguistics.

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

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, pages 219–230.

Hyun Duk Kim and ChengXiang Zhai. 2009. Generating comparative summaries of contradictory opinions in text. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 385–394. ACM.

Soo-Min Kim, Patrick Pantel, Tim Chklovski, and Marco Pennacchiotti. 2006. Automatically assessing review helpfulness. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 423–430. Association for Computational Linguistics.

Kevin Lerman and Ryan McDonald. 2009. Contrastive summarization: an experiment with consumer reviews. In *Proceedings of human language technologies: The 2009 annual conference of the North American chapter of the association for computational linguistics, companion volume: Short papers*, pages 113–116. Association for Computational Linguistics.

Kevin Lerman, Sasha Blair-Goldensohn, and Ryan McDonald. 2009. Sentiment summarization: Evaluating and learning user preferences. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 514–522.

Jure Leskovec13, Natasa Milic-Frayling, and Marko Grobelnik. 2005. Impact of linguistic analysis on the semantic graph coverage and learning of document extracts.

Jiwei Li and Sujian Li. 2012. A novel feature-based bayesian model for query focused multi-document summarization. *arXiv preprint arXiv:1212.2006*.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th conference on Computational linguistics*, volume 1 of *COLING '00*, pages 495–501.

Chin-Yew Lin, Guihong Cao, Jianfeng Gao, and Jian-Yun Nie. 2006. An information-theoretic approach to automatic evaluation of summaries. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 463–470. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.

Jingjing Liu, Yunbo Cao, Chin yew Lin, Yalou Huang, and Ming zhou. 2007. Low-quality product review detection in opinion summarization. In *Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing*.

Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. 2008. Modeling and predicting the helpfulness of online reviews. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 443–452. IEEE.

Yue Lu, Panayiotis Tsaparas, Alexandros Ntoulas, and Livia Polanyi. 2010. Exploiting social context for review quality prediction. In *Proceedings of the 19th international conference on World wide web*, pages 691–700.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Courtney Napoles, Benjamin Van Durme, and Chris Callison-Burch. 2011. Evaluating sentence compression: Pitfalls and suggested remedies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 91–97. Association for Computational Linguistics.

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005-101*.

Dragomir Radev, Timothy Allison, Sasha Blair-Goldensohn, John Blitzer, Arda Celebi, Stanko Dimitrov, Elliott Drabek, Ali Hakim, Wai Lam, Danyu Liu, et al. 2004. Mead-a platform for multidocument multilingual text summarization. Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004).

Martin Riedl and Chris Biemann. 2012. How text segmentation algorithms gain from topic models. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 553–557. Association for Computational Linguistics.

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

Lexico-syntactic text simplification and compression with typed dependencies

Mandya Angrosh
Computing Science,
University of Aberdeen,
Aberdeen, UK.
angroshmandya@abdn.ac.uk

Tadashi Nomoto
National Institute of
Japanese Literature,
Tokyo, Japan.
nomoto@acm.org

Advaith Siddharthan
Computing Science,
University of Aberdeen,
Aberdeen, UK.
advaith@abdn.ac.uk

Abstract

We describe two systems for text simplification using typed dependency structures, one that performs lexical and syntactic simplification, and another that performs sentence compression optimised to satisfy global text constraints such as lexical density, the ratio of difficult words, and text length. We report a substantial evaluation that demonstrates the superiority of our systems, individually and in combination, over the state of the art, and also report a comprehension based evaluation of contemporary automatic text simplification systems with target non-native readers.

1 Introduction

Text simplification has often been defined as the process of reducing the grammatical and lexical complexity of a text, while still retaining the original information content and meaning. However, text can also be simplified in other ways; for instance, by removing peripheral information to reduce text length, through sentence compression or summarisation. A key goal of automatic text simplification is to make information more accessible to the large numbers of people with reduced literacy, motivated by a large body of evidence that manual text simplification is an effective intervention (Anderson and Freebody, 1981; L'Allier, 1980; Beck et al., 1991; Anderson and Davison, 1988; Linderholm et al., 2000; Kamalski et al., 2008). However automatic text simplification systems have rarely been evaluated in a manner that sheds light on whether they can facilitate target users.

To date, evaluations of automatic text simplification have been (a) performed on a small scale, as few as 20–25 sentences in some cases (Wubben et al., 2012; Siddharthan and Mandya, 2014; Narayan and Gardent, 2014), (b) performed on sentences in isolation, thus not measuring incoherence caused at the inter-sentential level that can make text more difficult (Siddharthan (2003a) being the exception), and (c) performed using either automatic metrics (Zhu et al., 2010; Coster and Kauchak, 2011; Woodsend and Lapata, 2011; Wubben et al., 2012; Paetzold and Specia, 2013) or using ratings by fluent readers for fluency, simplicity and meaning preservation (Siddharthan, 2006; Woodsend and Lapata, 2011; Wubben et al., 2012; Paetzold and Specia, 2013; Siddharthan and Mandya, 2014; Narayan and Gardent, 2014; Mandya and Siddharthan, 2014). As such, none of these evaluations can help us answer the basic question: How good is automatic text simplification; i.e., would it facilitate poor readers?

Our goals in this paper are twofold. First, we want to evaluate text simplification systems more systematically than has been attempted before, using both human judgements on a larger scale, and directly testing comprehension on longer passages for target reader populations. Second, we want to compare two different approaches to text simplification. In this paper, we present a text simplification system that can perform lexical and syntactic simplification (§3), as well as a novel sentence compression system designed specifically for the text simplification task (§4), in that it favours compressions with fewer difficult words and with more function words such as connectives that are known to improve readability. We evaluate both, as well as a hybrid system that performs both text simplification and compression (§5, 6).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1996–2006, Dublin, Ireland, August 23-29 2014.

2 Related Work

Text simplification systems differ primarily in the level of linguistic knowledge they encode. Phrase Based Machine Translation (PBMT) systems (Specia, 2010; Wubben et al., 2012; Coster and Kauchak, 2011) use the least knowledge, and as such are ill equipped to handle simplifications that require morphological changes, syntactic reordering, sentence splitting or insertions. While syntax based MT approaches use syntactic knowledge, existing systems do not offer a treatment of morphology (Zhu et al., 2010; Woodsend and Lapata, 2011; Paetzold and Specia, 2013). This means that while some syntactic reordering operations can be performed well, others requiring morphological changes cannot. Consider converting passive to active voice (e.g., from "trains are liked by John" to "John likes trains"). Besides deleting auxiliaries and reordering the arguments of the verb, there is also a requirement to modify the verb to make it agree in number with the new subject "John", and take the tense of the auxiliary "are".

Hand crafted systems such as Siddharthan (2010) and Siddharthan (2011) use transformation rules that encode morphological changes as well as deletions, re-orderings, substitutions and sentence splitting, and can handle voice change correctly. However, hand crafted systems are limited in scope to syntactic simplification as there are too many lexico-syntactic and lexical simplifications to enumerate manually.

Some contemporary work in text simplification has evolved from research in sentence compression, a related research area that aims to shorten sentences for the purpose of summarising the main content. Sentence compression has historically been addressed in a generative framework, where transformation rules are learnt from parsed corpora of sentences aligned with manually compressed versions, using ideas adapted from statistical machine translation. The compression rules learnt are typically syntactic tree-to-tree transformations (Knight and Marcu, 2000; Galley and McKeown, 2007; Riezler et al., 2003; Cohn and Lapata, 2009; Nomoto, 2008) of some variety. Indeed, Woodsend and Lapata (2011) develop this line of research. Their model is based on quasi-synchronous tree substitution grammar (QTSG) (Smith and Eisner, 2006) and integer linear programming. Quasi-synchronous grammars aim to relax the isomorphism constraints of synchronous grammars, in this case by generating a loose alignment between parse trees. Woodsend and Lapata (2011) use QTSG to generate all possible rewrite operations for a source tree, and then integer linear programming to select the most appropriate simplification. Their system performs lexical and syntactic simplification as well as compression.

Recently, there have been attempts to combine approaches. Narayan and Gardent (2014) use an approach based on semantics to perform syntactic simplification, and PBMT for lexical simplifications. We have also created a hybrid system, but one using linguistically sound hand written rules for syntactic simplification and automatically acquired rules for lexicalised constructs (Siddharthan and Mandya, 2014; Mandya and Siddharthan, 2014). In this paper we combine this work (summarised in §3) with a new method for sentence compression (described in §4).

3 Text Simplification with Synchronous Dependency Grammars

We use the RegenT text simplification (Siddharthan, 2011), augmented with automatically acquired rules, as described in detail elsewhere (Mandya and Siddharthan, 2014; Siddharthan and Mandya, 2014). In this section, we will restrict ourselves to summarising the key features of the system.

Our text simplification system follows the architecture proposed in Ding and Palmer (2005) for Synchronous Dependency Insertion Grammars, reproduced in Fig. 1. It uses the same dataset[1] as Woodsend and Lapata (2011) for learning lexicalised rules. The rules are acquired in the format required by the RegenT text simplification system (Siddharthan, 2011), which is used to implement the simplification. This

[1] consisting of ∼140K aligned simplified and original sentence pairs obtained from Simple English Wikipedia and English Wikipedia.

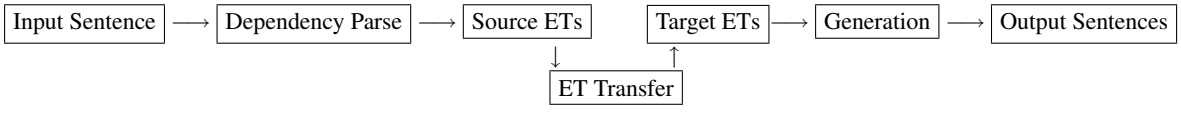


Figure 1: System Architecture

RULE 1: MOST_INTENSIVE2STRONGEST

1. DELETE
   (a) advmod(?X0[intensive], ?X1[most])
   (b) advmod(?X2[storm], ?X0[intensive])
2. INSERT
   (a) advmod(?X2, ?X3[strongest])

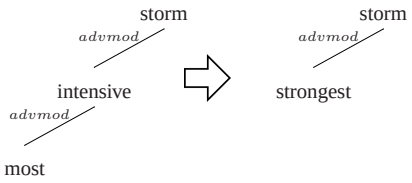


Figure 2: Simplification as a Transfer rule and a transduction of Elementary Trees (ETs)

requires dependency parses from the Stanford Parser, and generates output sentences from dependency parses using the generation-light approach described in (Siddharthan, 2011).

In short, we extract a synchronous grammar from dependency parses of aligned English and simple English sentences, starting from the differences in the parses. For example, consider two aligned sentences from the aligned corpus described in Woodsend and Lapata (2011):

1. (a) It was the second most intensive storm on the planet in 1989.
   (b) It was the second strongest storm on the planet in 1989.

An automatic comparison of the dependency parses for the two sentences reveals that there are two typed dependencies that occur only in the parse of the first sentence, and one that occurs only in the parse of the second. Thus, to convert the first sentence into the second, two dependencies need to be deleted and one inserted. From this example, the rule shown in Fig. 2 is extracted. The rule contains variables (?Xn), which can be forced to match certain words in square brackets.

Such deletion and insertion operations are central to text simplification, but a few other operations are also needed to handle morphology and to avoid broken dependency links in the Target ETs. These are enumerated in (Siddharthan, 2011). By collecting such rules, a meta-grammar is produced that can translate dependency parses in one language (English) into the other (simplified English). The rule above will translate "most intensive" to "strongest", in the immediate lexical context of "storm". The ET Transfer component can be presented either as transformation rules or as a transduction of ETs, as shown in Fig. 2. In Mandya and Siddharthan (2014), we describe how such automatically acquired rules can be generalised to apply in new contexts; for instance, by expanding lexical context to include related words derived from WordNet, or by removing the lexical context for lexical simplifications that are not context dependent.

Learning paraphrase with typed dependency representations has certain advantages to PBMT; for example, consider the rule that simplifies "described as" to "called":

RULE: DESCRIBED_AS2CALLED

1. DELETE:
   (a) prep_as(?X0[described], ?X1)
2. INSERT:
   (a) dobj(?X2[called], ?X1)

This single rule can simplify "*Coulter was described as a polemicist*" to "*Coulter was called a polemicist*" as well as cases where the words are not adjacent, such as "*Coulter has described herself as a polemicist*" to "*Coulter has called herself a polemicist*".

Our text simplification system, as evaluated in this paper, combines a set of 278 hand crafted grammar for syntactic simplification (from the original RegenT system) and 5172 automatically acquired rules, based on the principles described above.

4 Sentence Compression with Reluctant Trimmer

This section describes the mechanics of the reluctant trimmer (RT), or how it works to create a simplified form of sentence. We will explain later where the word 'reluctant' comes from. Broadly, RT comes in two parts: generation and selection. For a given sentence it takes as input, it generates a number of

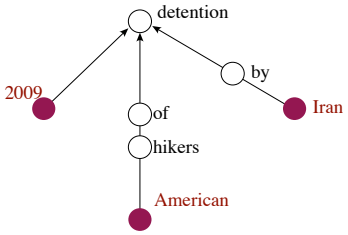


Figure 3: Dependency structure for *"2009 detention of American hikers by Iran"*

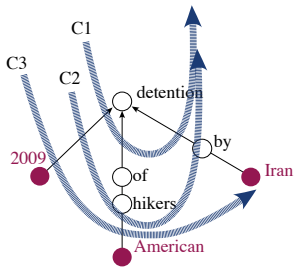


Figure 4: Cropping dependency tree

truncations of the sentence, each of which has some elements removed in a way that largely complies with English syntax. It does this by first parsing the sentence into a a dependency representation, and creating what we call terminating dependency paths out of the representation. After placing them in a lattice format, we run a K-best search over the lattice to generate K best truncations of the sentence. We repeat the process for each sentence found in the text, which will produce a collection of sets of truncation candidates. We then run integer linear programming over the collection, selecting one sentence for each set in a way that satisfies global constraints such as lexical density, the ratio of hard words, and text length. In particular, we regard RT not as an operation that works sentence by sentence, but one that works with text as a whole. We argue that how the sentence is to be compressed is not only dictated by the sentence itself, but also by the text in which it appears.

We start off with an example shown in Figure 3, where we have a phrase "*2009 detention of American hikers by Iran*." Our goal here is to develop a systematic method that will prune the dependency tree so as to generate shorter versions of the sentence largely in compliance with the English grammar. Figure 4 provides an intuitive picture of how this could be done: by cropping the tree along the arrows. We implement the idea by borrowing the notion of *Terminating Dependency Path* (TDP) (Nomoto, 2008), which gives us a way to translate a dependency tree into a trellis of nodes, which in turn allows us to find truncations through dynamic programming.

Figure 5 shows a TDP lattice derived from the dependency tree given in Figure 3. TDPs are depicted as solid blue lines in the figure. It is easy to see that each TDP corresponds to a path in the dependency tree that runs from a leaf to the root. The conversion from dependency tree to TDP lattice is thus straightforward. We perform A* search over the TDP lattice to find the best compression. Assume that we have a path or a sequence of nodes, $\langle n[1], n[2] \ldots, n[j], \ldots, n[z-1], n[z] \rangle$, that takes you from the starting node, $n[1]$, to the goal, $n[z]$, on the TDP lattice. Define the cost C of node $n[x]$ by: $C(x) = g(x) + h(x)$ where $g(x)$ is the cost incurred for the travel from the starting node to $n[x]$ and $h(x)$ the future estimate for the cost of travelling from $n[x]$ to the goal. Let $g(x) = -\sum_{j \in V(1,x)} \text{backward}(j)$ and $h(x) = -\sum_{j \in W(x,z-1)} \text{forward}(j)$, with:

$$\text{backward}(x) = \text{tfidf}(n[x]) + pr(\text{seq}(n[x-1], n[x]) | M), \quad (1)$$

$$\text{forward}(x) = \text{backward}(x+1) \quad (2)$$

$V(1,x)$ is a sequence of nodes that appeared on the path we took to reach $n[x]$ from the starting node, $W(x, z-1)$ a sequence of nodes that gives the shortest possible path (i.e. the path that incurs least cost) from $n[x]$ to the goal. tfidf(n) represents a tfidf score for a word associated with the node n, with tfidf$(n[1]) = 0$ and tfidf$(n[g]) = 0$, and is normalised so that it falls between 1 and 0.[2] seq(n, m) refers to an uninterrupted sequence of words you find on the path that extends from n to m via the root, ignoring duplicates. Figure 6 gives an intuitive sense of how this works. seq (2009, hiker), for instance, can be found by following the blue line in the figure, which results in "*2009 detention of hikers.*" 'M' refers to a language model.[3] $pr(\text{seq}(n, m)|M)$ is the probability of sequence 'seq(n,m)' under language

[2]Document frequencies (df) we used for present purposes are based on those given in the British National Corpus (`www.kilgarriff.co.uk/bnc-readme.html`), which keeps record of the number of files a particular word occurred.

[3]The language model is built here by running SRLM (`www.speech.sri.com/projects/srilm`) on the English

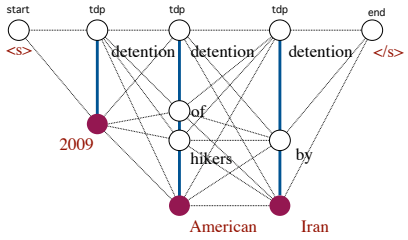


Figure 5: TDP lattice. '$\langle s \rangle$' is a label for the starting node, '$\langle /s \rangle$' that for the goal.

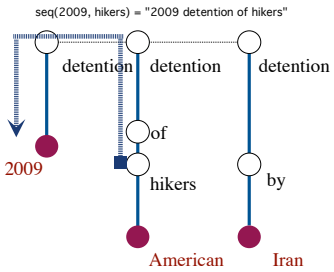


Figure 6: seq(2009,hiker)

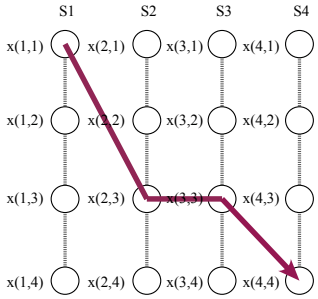


Figure 7: Decoding with ILP

model M.[4] Traversing over the TDP lattice while picking nodes with least costs will produce the best compression, to which we apply Yen (1971)'s algorithm to find K-best alternatives (where K is set to 10 in the experiments reported below).

We now turn to the second part of the story, which is about choosing from each pool of K-best candidates, to create a simplified version of the text. (Recall that we keep a pool of K-best compressions for each of the sentences in a text, and create a simplification by choosing a compression from each pool.) In this paper, we build on a particular simplification approach based on integer linear programming (ILP), by (Dras, 1999), which he dubbed 'reluctant paraphrasing.' In a nutshell, Dras claims that we should make as little change to the text as possible, arguing that any change may run the risk of muddling the meaning of the original text: hence the name 'reluctant paraphrasing.' The following linear program (LP) represents our adaptation of Dras's method. Formula 3 represents the objective function, with 4 through 7 expressing constraints:

$$\min \quad z = \sum c_{i,j} x_{i,j} \qquad (3)$$

subject to:

$$\forall i \sum_j x_{ij} = 1, \quad x_{ij} \in \{0,1\}, \forall ij \qquad (4)$$

$$\frac{W + \sum w_{ij} \cdot x_{ij}}{S} \leq k_1 \qquad (5)$$

$$\frac{H + \sum h_{ij} \cdot x_{ij}}{W + \sum w_{ij} \cdot x_{ij}} \leq k_2 \qquad (6)$$

$$\frac{F + \sum f_{ij} \cdot x_{ij}}{W + \sum w_{ij} \cdot x_{ij}} \geq k_3 \qquad (7)$$

$x_{i,j}$ denotes a candidate for which we are to make a decision on whether to include it in the simplification of a given text d. In particular we mean $x_{i,j}$ to represent the j-th best compression for the i-th sentence in d. Constraint 4 dictates that we have exactly one compression candidate for each sentence in d. w_{ij} indicates the number of changes or deletions we performed on the original sentence to create x_{ij}: -1 if x_{ij} has one less term than the original sentence it is a compression of; 0 if there is no change. W is the number of terms in d, S the number of sentences in d. Constraint 5 states that proportion of the number of terms to that of sentences should be less than or equal to k_1; in other words, changes made to the text should not exceed k_1. H in constraint 6 denotes the total number of 'hard' or difficult words in the original text; h_{ij} the number of changes made to hard words in x_{ij}, namely how many less or more words there remain that are deemed 'hard,' compared to the sentence it comes from.[5]: $h_{ij} = -3$, for example, means that we have three less hard words in x_{ij} than in the original sentence.

Constraint 6 is included here to keep the proportion of hard words in text from growing beyond a

portion of TDT5 corpus and TDT Pilot Study Corpus (both available at Linguistic Data Consortium), the total number of sentences combined reaching 293,971.

[4]We note here that we did not compensate the probability for the length of a word sequence, as we were unable to find an empirical evidence that suggested we should do otherwise.

[5]'Hard words' are defined here as those that fall off of the New General Service List (www.newgeneralservicelist.org) which currently contains 2,881most frequently used words.

particular threshold k_2. The values of k_1, k_2 and k_3 were determined based on the Breaking News English (BNE) corpus (described later), which provides for each story, simplified versions at two levels of difficulty, one being called 'easy' and the other 'hard.' If we take the 'easy' as a gold standard simplification for the 'hard,' we will be able to get estimates of k_1 through k_3. None of the data we used for this purpose, however, is part of the BNE reading test discussed below.

F in constraint 7 represents the total number of function words (those that are not of JJ, MD, NN, RB, or VB in the Penn scheme) while f_{ij} indicates that of changes to function words (the way it works is analogous to h_{ij}). The motivation for the constraint is to prevent function words from being eliminated excessively, which Dras argues, reduces the readability of text. The objective function includes parameters $c_{i,j}$ which serve to indicate the cost of transforming the sentence. In this paper, we define c_{ij} as Levenshtein edit distance between compression and original sentence. In ordinary language, the linear program may read like "Keep changes to a minimum. Accept compressions that look much like the original sentences from which they arise, with less of hard words and content terms and more of function words." Further, we made use of an array of hand-coded constraints in addition to a language model, to ensure that a compression we generate remains as grammatical as possible. Included were those that prohibit the generation of a compression that involves a dangling preposition or breaks apart multi-word prepositions (MWPs) such as *according to*, *compared to*, *in front of*, etc. (the complete list of MWPs we used for this purpose can be found in de Marneffe and Manning (2008)). Added to these were some "don't drop" rules that demanded we keep intact subjects and verbs as well.

Figure 7 illustrates how compression variables $x_{i,j}$ are organised (each of which is depicted as "$x(i, j)$" in the figure). Each vertical line represents a pool of K-best compressions generated for a particular sentence s_i. LP seeks to find a candidate from each pool so that the resulting set of compressions best meets the objective function and conditions it dictates.[6]

5 Evaluation of Fluency, Simplicity and Meaning Preservation

We performed a manual evaluation of how fluent and simple the text produced by our simplification system is, and the extent to which it preserves meaning. We evaluate 3 systems:

TS: The Text Simplification system based on synchronous dependency grammars (§3).

RT: The Reluctant Trimmer for sentence compression (§4).

HYB: A hybrid text simplification system that applies RT to the output of TS.

We used as a baseline Woodsend and Lapata (2011)'s QTSG system that learns a quasi-synchronous tree substitution grammar from the same EW-SEW dataset used by TS. QTSG is the best performing system in the literature with a similar scope to ours in terms of the syntactic, lexical and compression operations performed [7]. QTSG relies entirely on an automatically acquired grammar of 1431 rules, for lexical and syntactic simplification as well as sentence compression. Our TS system has an automatically extracted grammar with 5172 lexicalised rules to augment the existing 278 manually written syntactic rules in RegenT. The RT system is not trained on simplified text. We also compare against the manual simplification (SEW), and the original EW sentences.

Data: We use an evaluation set consisting of 100 sentences from English Wikipedia (EW) aligned with Simple English Wikipedia (SEW) sentences, following recent work (Woodsend and Lapata, 2011; Wubben et al., 2012; Zhu et al., 2010; Mandya and Siddharthan, 2014; Siddharthan and Mandya, 2014). These 100 sentences have been excluded from our training data for rule acquisition, as is standard. Following Wubben et al. (2012), we used all the sentences from the evaluation set for which each of the four systems had performed at least one simplification (as selecting sentences where no simplification is performed by one system is likely to boost its fluency and meaning preservation ratings). This gave us a test set of 50 sentences from the original 100.

[6] As an LP solver, we used lp_solve 5.5.2.0, a mixed integer programming solver, available under public license at SourceForge (`lpsolve.sourceforge.net/5.5`).

[7] The PBMT system of Wubben et al. (2012) reports better results than QTSG, but is not directly comparable because it does not perform syntactic simplifications such as sentence splitting.

	FLUENCY						SIMPLICITY						MEANING					
	EW	SEW	QTSG	TS	RT	HYB	EW	SEW	QTSG	TS	RT	HYB	EW	SEW	QTSG	TS	RT	HYB
Mean	3.97	4.09	2.20	3.53	3.19	3.01	3.40	3.54	2.41	3.79	3.15	2.83	-	4.14	2.52	3.44	3.43	3.28
SD	0.92	0.90	1.35	1.12	1.22	1.22	1.08	1.15	1.28	1.18	1.21	1.23	-	0.89	1.31	1.08	1.15	1.14
Median	4	4	2	4	3	3	3	4	2	4	3	3	-	4	2	4	4	3

Table 1: Results of human evaluation of different versions of simplified text

Method: We recruited participants on Amazon Mechanical Turk, filtered to live in the US and have an approval rating of 80%, and paid $3 for a HIT (Human Intelligence Task). Each HIT contained 10 sentences from Wikipedia (EW), each alongside 5 simplified versions: QTSG, TS, RT, HYB and SEW in a randomised manner. For each of these 10 sets, participants were asked to rate each simplified version for fluency, simplicity and the extent to which it preserved the meaning of the original EW sentence. Participants were also asked to rate the fluency and simplicity of the original EW sentence. We used a Likert scale of 1–5, where 1 is totally unusable output, and 5 is output that is perfectly usable.

Results: The results are shown in Table 1. As seen, our HYB system, and the individual components TS and RT all outperform QTSG with all three metrics. In particular, TS is comparable to the SEW version when one looks at the median scores. Interestingly, TS performs better than SEW with respect to simplicity, suggesting that the system is indeed capable of a wide range of simplification operations. The ANOVA tests carried out to measure significant differences between versions is presented below. Table 3 (Row 1) shows the average number of words in the original and each simplified version.

Fluency: A one-way ANOVA was conducted with *fluency* as the dependent variable and text *version* as the fixed effect. We report a significant effect of version (EW, SEW, QTSG, HYB, TS, RT) on the fluency score (F=173.1, p<10^{-16}). A Tukey's pairwise comparison test (Tukey's HSD, overall $\alpha = 0.05$) indicated significant differences between all pairs, except SEW-EW at $p < 0.05$.

Simplicity: A one-way ANOVA was conducted with *simplicity* as the dependent variable and text *version* as the fixed effect. We report a significant effect of version on the simplicity score (F=29.9, p<10^{-16}). A Tukey's pairwise comparison test (Tukey's HSD, overall $\alpha = 0.05$) indicated significant differences between all pairs except: EW-SEW, RT-EW, and SEW-TS at $p < 0.05$.

Meaning: A one-way ANOVA was conducted with *meaning preservation* as the dependent variable and text *version* as the fixed effect. We report a significant effect of version on the meaning preservation score (F=130.12, p=2x10^{-16}). A Tukey's pairwise comparison test (Tukey's HSD, overall $\alpha = 0.05$) indicated significant differences between all pairs except: RT-TS, RT-HYB and HYB-TS at $p < 0.05$.

Error Analysis: We manually examined sentences that had average ratings below 2. The main cause of error for TS was misparsing, particularly errorful relative clause attachment and the parsing of comma separated lists as apposition. TS fails badly in such cases, and it is possible that methods such as those described in Siddharthan (2003b) are still relevant for correcting parser output. RT suffers mainly when it removes punctuation, which make reading difficult, or names that contain meaning (e.g., "*Seven volumes in length , it was composed by Buddhist priest Jien of the Tendai sect c. 1220.*" got compressed to "*Seven volumes in length it was composed by Jien of the sect c. 1220.*"). The hybrid system can create inconsistencies when TS has split a sentence and RT removes names from only one part ("*Moles can be found in most parts of North America, Asia, and Europe, although there are no moles in Ireland.*" got simplified to "*Moles can be found in parts of America, and Asia and Europe. But, there are no moles.*").

6 Evaluation of Reading Comprehension

We also investigate, for the first time, the effect of contemporary text simplification systems on reading comprehension for non-native speakers with a range of English skills.

Method: The test was conducted on Amazon Mechanical Turk with participants chosen from India and paid $0.75 each. There is no method to selectively recruit low reading skill participants on Turk, so these setting were selected to recruit non-native speakers (India) and minimise participants with postgraduate

degrees (low pay). The test comprised of two components - (a) pre-test for English vocabulary skills; and (b) a reading comprehension test to measure the effect of text simplification.

Pre-test: Reading skills are multifaceted and typically assessed through test batteries that test a range of skills. As such there is no comprehensive assessment possible using a single short online test. As we are recruiting non-native speakers, we chose to use the vocabulary size test (Nation and Beglar, 2007), designed to estimate both first language and second language learners' written receptive vocabulary size in English. The test ranks words based on their corpus frequency, and creates 14 levels, each with 1000 words, so that level 10 for example would contain the 9001^{th} to 10000^{th} most frequent words in English. We designed our vocabulary test by using 28 items, 2 at each level[8]. Each word is tested by showing a short sentence containing it and asking the participant to select the meaning of the word from four options. An estimate of vocabulary size can be got by multiplying the score on this test by 500, so the maximum vocabulary size estimate is 28*500=14,000. Nation and Beglar (2007) spell out three important milestones in terms of word family vocabulary size:

> 5000: Minimum for Non-native speakers of non-European backgrounds to cope at English speaking Universities
>
> 8000: Critical goal for language learners to deal with a range of unsimplified language (98% coverage for newspapers)
>
> 9000: Level of non-native English speaking PhD students (98% coverage for English novels)

In addition, we asked participants to self-report their English language skills by selecting from following options: (a) native; (b) fluent (non-native); (c) good (non-native); and (d) basic (non-native).

Main test: The reading comprehension tests were conducted using 5 news summaries chosen from the Breaking News English[9] (BNE) website, with the permission of its creator and maintainer. The BNE website is a resource that provides high quality news summaries at various levels of simplification for second language learners, and has recently been nominated by the British Council for the 2014 ELTons award for Innovation in Learner Resources. We selected five news stories which had manually constructed summaries at reading levels 6 (hard) and 4 (easy). The website provides a range of exercises following each summary at level 6. We chose to use the multiple choice test to assess reading comprehension. For each of these summaries, we created automatically simplified texts by running our systems on the level 6 text. This resulted in a total of five versions for each news summary - L6 (original); L4 (manual simplification); TS (automatic simplification of L6); RT (compression of L6); and HYB (RT applied to output of TS applied to L6).

We used a balanced design where each participant would (after taking the vocabulary pre-test described above) see each of the 5 news stories in exactly one of the 5 versions in a Latin square design. For each comprehension test, the news summary was shown for a maximum of 150 seconds, after which it was removed and 5 multiple choice comprehension questions presented, which was available for another 150 seconds (2.5 minutes). Participants could finish before the 150 seconds by clicking a "finished" button. Table 3 shows the average length of text in each version.

Results: The first row in Table 2 shows the accuracy (proportion of comprehension questions answered correctly) on the main comprehension test for participants divided into four categories based on their estimated vocabulary from the pre-test. We do not find any significant differences, but it appears that the main benefits of automatic text simplification are for moderate readers (vocabulary between 5K and 8K).

We found a very poor correlation between participants' self reported English language skills and their performance on the vocabulary test ($\rho = -0.01$; $p = 0.55$). The poor correlation was due to certain participants over-estimating their skills. Out of 50 participants, 3 rated themselves as native. However, they could get only about 28% of the answers correct, showing the fact that the participants had over-estimated themselves.

This caused us to doubt the reliability of our version of the vocabulary test[10]. We therefore also attempted to categorise participants based on their overall accuracy over all 25 questions in the com-

[8]The original test uses 10 words from each level, but we required a shorter version.

[9]`www.breakingnewsenglish.com`

[10]The published results are for a 140 question test taking 40 minutes, which we have had to reduce to 28 questions for practical reasons.

	L4	L6	TS	RT	HYB	L4	L6	TS	RT	HYB	L4	L6	TS	RT	HYB	L4	L6	TS	RT	HYB
Skills	Excellent (Vocab≥9000)					Good (9000>Vocab≥8000)					Mod (8000>Vocab≥5000)					Poor (Vocab<5000)				
Accuracy	0.69	0.92	0.94	0.85	0.78	0.84	0.87	0.80	0.84	0.77	0.74	0.78	0.80	0.82	0.80	0.64	0.77	0.72	0.58	0.55
Size	13 Participants					10 Participants					14 Participants					13 Participants				
Skills	Excellent (acc≥.9)					Good (.9>acc≥.8)					Mod (.8>acc≥.5)					Poor (acc<.5)				
Accuracy	0.88	0.98	0.95	0.90	0.83	0.75	0.87	0.84	0.82	0.77	0.60	0.70	0.75	0.63	0.58	0.53	0.53	0.40	0.33	0.33
Size	8 Participants					31 Participants					8 Participants					3 Participants				

Table 2: Results of comprehension tests: Mean accuracy (proportion of comprehension questions answered correctly) by reading comprehension skills. Row 1: Participants categorised by estimated vocabulary from pretest. Row 2: Participants categorised based on accuracy on comprehension tests.

	Dataset	Original	Simplified	TS	RT	HYB	QTSG
Average words per text	Wikipedia Evaluation Set	27.0 (EW)	20.4 (SEW)	25.3	22.0	20.6	24.0
Average words per text	Breaking News Evaluation Set	172.6 (L6)	152.8 (L4)	184.4	149.2	151.4	-

Table 3: Effect of simplification of sentence and document lengths

prehension test. While the thresholds of 5000, 8000 and 9000 for vocabulary size are derived from the literature, we had to set these threshold for comprehension scores. To do this in an objective (though still arbitrary) manner, we selected thresholds numerically similar to the vocabulary size thresholds: Excellent ($acc \geq 0.9$), Good($0.9 > acc >= 0.8$), Moderate ($0.8 > acc \geq 0.5$) and Poor ($acc < 0.5$).

The second row in Table 2 shows the accuracy of participants when categorised by average accuracy on the comprehension questions. Note that this categorisation is posthoc (though we have used thresholds derived from the vocabulary test to be objective), and the results pertaining to this categorisation should be regarded as preliminary. This new categorisation based on observed reading ability, rather than predicted language skills, throws up more definitive results. We fitted a Generalised Linear Mixed Model (GLMM), with "correct" answer as the (binary) dependent variable, text "version" (L4, L6, TS, RT, HYB) and "comprehension" (Excellent, Good, Moderate, Poor) as the fixed effects and participant and question as the random effects. We found a strong main effect of comprehension (comprehension=moderate, $z = -3.178$, $p = 0.001$; comprehension=poor, $z = -4.858$, $p < 0.0001$) and a weak effect of version (version=L4, $z = -1.797$, $p = 0.073$); i.e., these three conditions predict a reduced accuracy on the test. We also found a weak interaction between comprehension and version (comprehension=moderate:version=TS, $z = 1.78$, $p = 0.075$); i.e., that TS increases correct answers for readers with moderate reading skills ($p = 0.075$).

Note that L4, RT and HYB all omit information through compression (Table 3 shows text lengths). This explains the drop in comprehension for these versions, as some information needed to answer a question might have been omitted from the summary. Note also that RT and the HYB systems are competitive with the manual simplification L4 for moderate and good readers. Table 4 provides sample texts to illustrate differences.

L6	The United Nations has warned that the Central African Republic (CAR) needs urgent help. The UN Deputy Secretary-General Jan Eliasson said it was 'descending into complete chaos before our eyes'. The landlocked nation has been slowly moving towards a state of total anarchy since rebels seized power in March.
L4	The U.N. has asked for urgent help for the Central African Republic. The UN's Jan Eliasson said it was 'descending into complete chaos'. There is almost a state of anarchy after rebels took power in March.
TS	The United Nations has warned that the Central African Republic , CAR , needs urgent help. The UN Deputy Secretary-General Jan Eliasson said: It was ' descending into complete chaos before our eyes '. The landlocked nation has been slowly moving towards a state of total anarchy. This happened since rebels seized power in March.
RT	The Nations has warned that the Republic needs help. The Deputy Secretary-General Jan Eliasson said it was descending into complete chaos before our eyes. The nation has been slowly moving towards a state of anarchy since rebels seized power in March.
HYB	The Nations has warned that the Central African Republic CAR needs urgent help. The Deputy Secretary-General Jan Eliasson said It was descending into complete chaos before our eyes. The nation has been moving towards a state This happened since rebels seized power.

Table 4: Example of system output to illustrate differences (Beginning of comprehension story 3).

7 Conclusions

We have described and evaluated two different text simplification systems, one that performs lexical and syntactic simplification, and another that performs sentence compression, optimised for the text simplification task. Both systems and their combination outperform a leading contemporary system. The evaluation of reading comprehension with non-native speakers provides preliminary results that automatic text simplification can facilitate comprehension for moderate readers, but not for good ones. A larger evaluation with moderate readers in necessary to confirm this. Finally we plan to make the TS and RT systems available to the public under the Creative Commons license.[11]

Acknowledgements

This research is supported by an award made by the EPSRC; award reference: EP/J018805/1.

References

Richard C Anderson and Alice Davison. 1988. *Conceptual and empirical bases of readability formulas.* Lawrence Erlbaum Associates, Inc.

Richard Anderson and Peter Freebody. 1981. Vocabulary knowledge. In John Guthrie, editor, *Comprehension and Teaching: Research Reviews*, pages 77–117. International Reading Association, Newark, DE.

Isabel L. Beck, Margaret G. McKeown, Gale M. Sinatra, and Jane A. Loxterman. 1991. Revising social studies text from a text-processing perspective: Evidence of improved comprehensibility. *Reading Research Quarterly*, 26(3):251–276.

T. Cohn and M. Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34(1):637–674.

William Coster and David Kauchak. 2011. Learning to simplify sentences using wikipedia. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 1–9. Association for Computational Linguistics.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. Stanford typed dependencies manual. http://nlp.stanford.edu/software/dependencies_manual.pdf.

Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 541–548. Association for Computational Linguistics.

Mark Dras. 1999. *Tree adjoining grammar and the reluctant paraphrasing of text*. Ph.D. thesis, Macquarie University NSW 2109 Australia.

Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 180–187, Rochester, New York, April. Association for Computational Linguistics.

J. Kamalski, T. Sanders, and L. Lentz. 2008. Coherence marking, prior knowledge, and comprehension of informative and persuasive texts: Sorting things out. *Discourse Processes*, 45(4):323–345.

K. Knight and D. Marcu. 2000. Statistics-based summarization — step one: Sentence compression. In *Proceeding of The American Association for Artificial Intelligence Conference (AAAI-2000)*, pages 703–710.

J.J. L'Allier. 1980. *An evaluation study of a computer-based lesson that adjusts reading level by monitoring on task reader characteristics*. Ph.D. thesis, University of Minnesota, Minneapolis, MN.

T. Linderholm, M.G. Everson, P. van den Broek, M. Mischinski, A. Crittenden, and J. Samuels. 2000. Effects of Causal Text Revisions on More-and Less-Skilled Readers' Comprehension of Easy and Difficult Texts. *Cognition and Instruction*, 18(4):525–556.

Angrosh Mandya and Advaith Siddharthan. 2014. Text simplification using synchronous dependency grammars: Generalising automatically harvested rules. In *INLG 2014 Proceedings of the Eighth International Natural Language Generation Conference*, pages 16–25, Philadelphia, PA, June. Association for Computational Linguistics.

[11] For information on the availability of systems, visit us at: `www.quantmedia.org/coling2014/`.

Shashi Narayan and Claire Gardent. 2014. Hybrid simplification using deep semantics and machine translation. In *Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics.*, pages 435–445, Baltimore, MD. Association for Computational Linguistics.

I. S. P. Nation and David Beglar. 2007. A vocabulary size test. *The Language Teacher*, 31(7):9–13.

Tadashi Nomoto. 2008. A generic sentence trimmer with CRFs. In *Proceedings of ACL-08: HLT*, pages 299–307, Columbus, Ohio, June. Association for Computational Linguistics.

Gustavo H. Paetzold and Lucia Specia. 2013. Text simplification as tree transduction. In *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*, pages 116–125.

Stefan Riezler, Tracy H. King, Richard Crouch, and Annie Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of the Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'03)*, Edmonton, Canada.

Advaith Siddharthan and Angrosh Mandya. 2014. Hybrid text simplification using synchronous dependency grammars with hand-written and automatically harvested rules. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 722–731, Gothenburg, Sweden, April. Association for Computational Linguistics.

Advaith Siddharthan. 2003a. Preserving discourse structure when simplifying text. In *Proceedings of the European Natural Language Generation Workshop (ENLG), 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*, pages 103–110, Budapest, Hungary.

Advaith Siddharthan. 2003b. Resolving pronouns robustly: Plumbing the depths of shallowness. In *Proceedings of the Workshop on Computational Treatments of Anaphora, 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*, pages 7–14, Budapest, Hungary.

Advaith Siddharthan. 2006. Syntactic simplification and text cohesion. *Research on Language and Computation*, 4(1):77–109.

Advaith Siddharthan. 2010. Complex lexico-syntactic reformulation of sentences using typed dependency representations. In *Proceedings of the 6th International Natural Language Generation Conference (INLG 2010)*, pages 125–133, Dublin, Ireland.

Advaith Siddharthan. 2011. Text simplification using typed dependencies: a comparison of the robustness of different generation strategies. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 2–11. Association for Computational Linguistics.

David A Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 23–30. Association for Computational Linguistics.

Lucia Specia. 2010. Translating from complex to simplified sentences. In *Proceedings of the Conference on Computational Processing of the Portuguese Language*, pages 30–39. Springer.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 409–420. Association for Computational Linguistics.

Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1015–1024. Association for Computational Linguistics.

Jin Y. Yen. 1971. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712–716, July.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd international conference on computational linguistics*, pages 1353–1361. Association for Computational Linguistics.

Learning when to point: A data-driven approach

Albert Gatt
Institute of Linguistics
University of Malta
albert.gatt@um.edu.mt

Patrizia Paggio
Institute of Linguistics, Uni of Malta
Centre for Language Technology, Uni of Copenhagen
patrizia.paggio@um.edu.mt

Abstract

The relationship between how people describe objects and when they choose to point is complex and likely to be influenced by factors related to both perceptual and discourse context. In this paper, we explore these interactions using machine-learning on a dialogue corpus, to identify multimodal referential strategies that can be used in automatic multimodal generation. We show that the decision to use a pointing gesture depends on features of the accompanying description (especially whether it contains spatial information), and on visual properties, especially distance or separation of a referent from its previous referent.

1 Introduction

The automatic generation of multimodal referring actions is a relatively under-studied phenomenon in Natural Language Generation (NLG). While there has been extensive research on Referring Expression Generation (REG) focusing on the choice of content in expressions such as (1) below (Dale, 1989; Dale and Reiter, 1995; Krahmer and van Deemter, 2012), their multimodal counterpart – exemplified in (2) – raises questions that go beyond these choices.

(1) the group of five large red circles

(2) there's a group of five large red ones [+pointing gesture with arm extended]

One important question concerns the appropriateness of a pointing gesture under different conditions. The relevant conditions here include both the physical or perceptual common ground shared by interlocutors (for example, what other objects are in the vicinity of the target referent, and therefore potentially confusable with it), the discursive common ground (for example, whether this object has been referred to before) and the content of the interlocutor's speech act, that is, what she chooses to say in addition to pointing. For example in (2), the speaker, who is engaged in a dialogue in which she needs to guide her interlocutor through a route on an abstract map (see Section 3 below), has chosen to use the cardinality of the referent (it is a group made up of five circles), its size, and its colour. Her choice of properties may be sufficient to distinguish it from all its distractors in the current context. However, unlike (1), (2) is a *composite utterance* consisting of two communicative modalities, each of which contributes to the communicative intention (Enfield, 2009).

This paper addresses the question of when a pointing gesture is appropriate as part of a composite, multimodal referring action. This is an important component of many multimodal generation systems, including those that communicate through embodied agents. We address this question in a data-driven manner, using a corpus of dialogues in which references have been annotated at both the level of speech and gesture. Our aim is to learn strategies for combinations of pointing and describing, as a function of perceptual and discursive features. We first summarise some relevant psycholinguistic and computational work (Section 2), before describing our corpus data (Section 3) and reporting on the machine-learning experiments conducted (Section 4). Section 5 concludes with some remarks on future work.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2007–2017, Dublin, Ireland, August 23-29 2014.

2 Pointing and reference

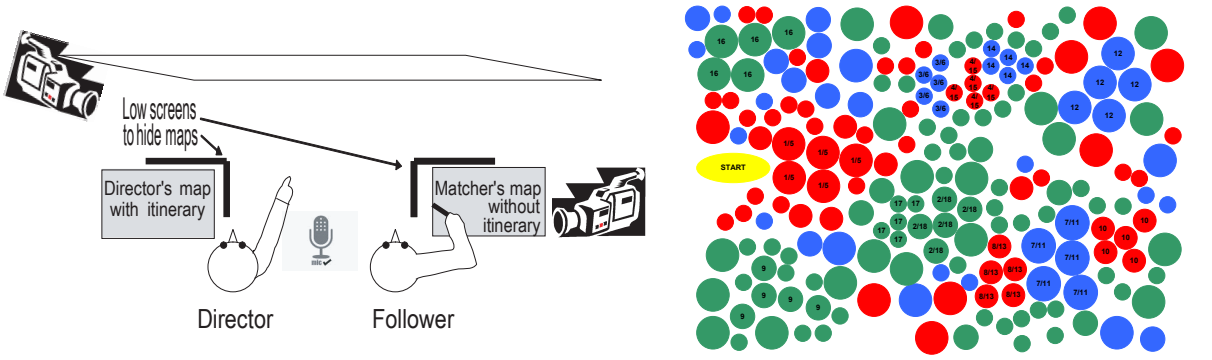
The idea that gesture and speech are planned separately, incorporated in early work on multimodal generation (André and Rist, 1996) is contradicted by more recent psycholinguistic research, in which gesture and language are increasingly viewed as tightly coupled (Kita and Özyürek, 2003; McNeill, 1985; McNeill and Duncan, 2000), contributing jointly to the composite utterances (Enfield, 2009). This view has also influenced recent work in multimodal NLG. For example, Kopp et al. (2008) use 'multimodal concepts', combining propositional and gestural or perceptual information.

In the case of referring expressions, pointing has been treated as a property, on a par with an object's colour or size. Thus, van der Sluis and Krahmer (2007) propose an algorithm in a graph-based framework (Krahmer et al., 2003) which selects pointing gestures of varying degrees of precision based on their cost when compared to other linguistically realisable features. Similarly, Kranstedt and Wachsmuth (2005) propose an extension of Dale and Reiter's (1995) Incremental Algorithm, which initially considers the possibility of producing an unambiguous pointing gesture. If this fails, a pointing gesture that is less precise may be generated, together with descriptive features of an object.

Both of these approaches assume that the choice of modality in a referring action ultimately hinges on a trade-off between what can be said and what is easiest to produce, a view that has some empirical support (Beun and Cremers, 1998; Bangerter, 2004; Piwek, 2007). On the other hand de Ruiter et al. (2012) found that likelihood of pointing was unaffected by the difficulty of using descriptive features. From a computational perspective, our earlier work (Gatt and Paggio, 2013) also found evidence, based on a machine-learning study on dialogue data, for the co-occurrence of pointing with descriptive (especially spatial) features, suggesting that pointing gestures may be planned in tandem (and not in competition) with these features.

The present paper uses the same corpus data as Gatt and Paggio (2013); however, that paper focused on the relationship between descriptive features (in the spoken part of the utterance) and pointing. In contrast, here we take a much broader view, also addressing the impact of the physical/perceptual features of the objects under discussion, and aspects of the discourse history.

3 Data used in this study



(a) Experiment setup

(b) Group circles map (numbers indicate the order in which landmarks are visited along the itinerary)

Figure 1: MREDI dialogue setup (reproduced from Gatt and Paggio (2013)).

We use the MREDI (Multimodal REference in DIalogue) corpus (van der Sluis et al., 2008; Gatt and Paggio, 2013), a collection of MapTask-like dialogues (Anderson et al., 1991). Dialogues in MREDI were conducted by dyads consisting of a Director and a Follower. The Director's task was to guide the Follower along a route through a visually shared 'map', located approximately one metre away, directly in front of them, blown up to roughly A0 size. The Director also had a private map on which the route was indicated, while the Follower's private map was used to mark the route as it unfolded in the course of the conversation. Figure 1a displays the basic setup.

There were no restrictions on what interlocutors could say. Participants in the study were told in advance that they could use both speech and gestures, but were not explicitly instructed to point. The maps consisted of collections of shapes of different colours and sizes and were very densely populated (see Figure 1b). Four maps were used in the study: in two of these, landmarks consisted of individual circles or squares, while in the other two they consisted of groups or clusters of five circles or squares (Figure 1b is a group circle map). In the group maps, all elements of a group of five were of the same colour and size.

On each map, there were 18 'landmarks'; these were the milestones along the itinerary and were marked on the Director's private map, but not visible on the large map that constituted the common ground. For example, the landmarks (groups of 5 circles) in Figure 1b are numbered from 1 to 18. Each dyad did all four maps; the order was randomised for each pair of participants. Participants switched roles between one map and another. In addition to the difference between group and individual landmarks, the maps were designed to manipulate a number of independent variables:

1. **Distinguishing Properties (DistProps):** Landmarks on the itinerary differed from their distractors – the objects in their immediate vicinity (the *focus area*) – in colour, or in size, or in both colour and size. The focus area was defined as the set of objects immediately surrounding a target. This means that different landmarks required different combinations of properties to ensure that they could be unambiguously identified by a description. For example, in Figure 1b, the group marked 17 consists of a landmark where size is the distinguishing feature, since all five circles in the group are small, and the objects in their immediate vicinity are either large or medium-sized. There were equal numbers of landmarks on each map that could be distinguished by colour only, size only, or both.

2. **Prior reference (Discourse):** Some of the landmarks were visited twice in the itinerary; these are indicated using two numbers in Figure 1b. Thus, landmark 8 in this map was also visited later as landmark 13. There were 6 landmarks on each map that were revisited in this way. This is the primary manipulation related to discourse history.

3. **Shift of domain focus (Distance):** Landmarks were located either near to or far away from the previous target. For example, in Figure 1b, landmark 17 and landmark 18 are adjacent ('near' condition), but landmark 17 is far from the preceding landmark 16.

In what follows, we use data from 8 dyads. Similar to Gatt and Paggio (2013), we only consider utterances by Directors. These were transcribed and split up according to the landmark to which they corresponded. In case a landmark was described over multiple turns in the dialogue, each turn was annotated as a separate utterance. Our dataset consists of a total of 2255 such utterances, of which 370 (16.4%) contain a pointing gesture. This is a relatively low proportion of such gestures, compared to some previous studies, such as Beun and Cremers (1998), who found that 48% of referential acts in their task-oriented dialogue corpus included a pointing gesture. However, Beun and Cremers focussed exclusively on first-mention referring expressions. Furthermore, the low proportion of pointing gestures in MREDI may be due to the fact that under our definition, the identification of a landmark may be spread over several turns, with possible interruptions by the Follower. Each such turn constitutes a separate utterance. This raises the likelihood that certain features of the composite utterance, including pointing, will only occur on some of the turns.

3.1 Features

Utterances in MREDI were annotated with the features displayed in Table 1. These codify aspects of the descriptive content of a referential act, as well as the presence or absence of a pointing gesture.

The features originally encoded in the MREDI corpus had frequency values; Gatt and Paggio (2013) used these frequencies in their study. However, for our experiments, we collapsed the features related to descriptive content – hereafter referred to as *descriptive* features – into boolean features. This significantly reduces the feature set and makes the rules acquired in our machine-learning experiments easier

	Feature	Name	Definition	Example
Visual	S	Size	mention of the target size	*the group of small circles*
	Sh	Shape	mention of the target shape	*the circles at the bottom*
	C	Colour	mention of the target colour	*The blue square near the red square*
Deictic/anaphoric	ID	Identity	Statement of identity between the current and a previous or later target	*the red square, the same one we saw at number 5*
	D	Deixis	Use of a deictic reference	*those squares*
Locative	RP	Relative position	Position of the target landmark relative to another object on the map	*the blue square just below the red square*
	AP	Absolute position	Target position based on absolute frame of reference	*The blue circle down at the bottom*
	FP	Path references	References to non-targets on the path leading to the target.	*go east to the first tiny square, past the blue one*
	DIR	Directions	Direction-giving.	*take a right, go across and straight down*
Action	GZ	Gaze	Gaze at the shared map (boolean).	
	Point	Pointing	Use of a pointing gesture (boolean).[1]	

Table 1: Features annotated in the dialogues. All features have frequency values, except for the Action features, which are boolean.

to interpret. Further, it enables us to test our hypothesis that the presence or absence of a *type* of feature (descriptive, physical or discursive) impacts the decision to point. The boolean descriptive features are as follows:

1. **Deixis**: this has the value `true` if the utterance contains a demonstrative pronoun (such as *that*), or a reference to the landmark that identifies it with the previous landmark. Thus, this feature is `true` if $ID > 0$ or $D > 0$ in Table 1;

2. **Locative**: this has the value `true` if the utterance contains any of the spatial properties in Table 1. Thus, the feature is `true` if $AP > 0$ or $RP > 0$ or $FP > 0$ or $DIR > 0$.;

3. **Visual**: this has the value `true` if the utterance contains at least one mention of the landmark's visual properties. This, the feature is `true` if $C > 0$ or $Sh > 0$ or $S > 0$.

In addition to these features, our experiments also made use of the *physical* features (**Distance** and **DistProps**) manipulated as part of the MREDI data collection study (see above), as well as the feature **Discourse**, which encodes prior reference.

Finally, we added a new feature to the dataset, **MapConfl**, which indicates the type of map on which utterances were produced, namely, individual or group circles or squares. This feature was included because the larger size of group landmarks, compared to individuals, may have influenced the decision to point, since groups are more visually salient.

The feature *Gaze* is present whenever a pointing gesture is made; hence, it is not used in the machine learning experiments reported below.

4 Experiments

In our earlier study on the MREDI corpus (Gatt and Paggio, 2013), investigating the relationship between pointing and descriptive features, we found that the latter could indeed be used as predictors of pointing gestures with an accuracy of 0.833 (F-score). The study also concluded that among the descriptive features it was locative properties that were most useful in guiding the decision of whether or not to point, compared to features describing visual characteristics of the objects.

However, in much of the work reviewed in Section 2, especially work arguing in favour of a trade-off in cost between pointing and describing, the occurrence of pointing is made to depend on the physical properties of referents. Therefore, in the present study we want to test whether the occurrence of pointing

gestures can be predicted more accurately as a function of (i) the descriptive features that speakers use to refer to landmarks; (ii) the physical/perceptual context in which they are found and (iii) whether or not they have been referred to earlier in the discourse. Furthermore, we want to investigate which combinations of physical and descriptive features provide the best results.

Two sets of experiments were conducted on different versions of the MREDI dataset. The first dataset (referred to as the complete dataset) is the same one used in the Gatt and Paggio (2013) study. It includes all of the 2255 Director's utterances from the eight dyads in the corpus, including those that did not contain any references at all, linguistic or gestural. Such utterances might, for example, be confirmations or feedback produced in the course of the dialogue.

We also report results on a second dataset (referred to as the referential dataset), consisting of all utterances that contain a reference, either using descriptive features, pointing, or both. This dataset consisted of 1542 utterances. Note that the number of utterances with a pointing gesture is still 370 in the pruned dataset.

The task in the experiments was to classify the binary feature *Point*. As mentioned earlier, 370 of these utterances contain a pointing gesture. In other words, there are 370 occurrences of *Point=1*.

All the experiments were run using the Weka tool (Witten and Frank, 2005), which gives access to many different algorithms, using 10-fold cross-validation throughout. In the experiments with the complete dataset, the ZeroR and OneR classifiers were first run on the data to establish a baseline. ZeroR always chooses the most frequent value of the class that is being predicted; in the present case, it consistently classifies all utterances as *Point = 0*, since the majority of utterances do not containg pointing gestures. OneR identifies a single feature, on the basis of which all classifications are made. On the MREDI data, OneR always assigned *Point = 0* to all utterances, based on a single rule using the *MapConfl* feature (i.e. the type of map or domain in which the dialogue was being carried out). Note that both baseline classifiers were trained using all features.

Various combinations of descriptive and physical features were then tested using different classifiers in Weka, including NaiveBayes, Support Vector Machines, Maximum Entropy (Logistic in Weka) and the J48 Decision Tree classifier. The present paper will report results for the last two of these, for the following reasons. First, these were the ones which performed best. In addition, the decision trees built by J48 provide an analysis tool to understand how the various features interact, given their transparency; on the other hand, MaxEnt sometimes outperforms J48 and provides a 'ceiling', in addition to the baselines described above.

The strategy used in testing feature combinations was essentially ablative. We tested first using all features, and then compared the performance of the classifiers when they use only descriptive features (Visual, Locative and Deixis), or only Discourse together with the physical features (DistProps and Distance). Omitting descriptive features and using only physical features with Discourse invariably performed near or below baseline (see below). Thus, we experimented with combinations of descriptive features and each physical feature, as well as Discourse, individually.

4.1 Results on the complete dataset

The results for the complete dataset are shown in Table 2 in terms of Precision, Recall and F-measure for each of the classifiers. The top rows display the results using all features, while the baseline results are in the bottom rows. The remaining results for different combinations of features are in descending order of F-score.

Interestingly, using all features – i.e. MapConfl, DistProps, Discourse, Distance, Visual, Locative and Deictic – with or without MapConfl, results in worse overall performance than using a combination of descriptive features (Locative, Deictic and Visual) with Distance. This combination is closely matched for accuracy by the combination involving descriptive features, Distance and DistProps. However, dropping Distance (using only descriptive features and DistProps) results in worse performance.

The addition of Distance and/or DistProps clearly improves the predictive accuracy of a classifier that uses descriptive features. However, the worst combination is found when the descriptive features are excluded. This is in line with the results reported by Gatt and Paggio (2013), who found that features of

Classifier	P	R	F	Features
J48	0.827	0.847	0.832	All
Logistic	0.831	0.854	0.828	All
J48	0.833	0.851	0.838	All - MapConfl
Logistic	0.832	0.851	0.837	All - MapConfl
Logistic	0.839	0.853	0.844	Descriptive + Distance
J48	0.839	0.853	0.844	Descriptive minus Deictic + Distance
Logistic	0.839	0.853	0.844	Descriptive minus Deictic + Distance
J48	0.836	0.851	0.84	Descriptive+DistProps + Distance
J48	0.839	0.853	0.84	Descriptive+Distance
Logistic	0.833	0.851	0.838	Descriptive+DistProps + Distance
J48	0.821	0.847	0.824	Descriptive+DistProps
Logistic	0.809	0.842	0.794	Only Descriptive
Logistic	0.809	0.842	0.794	Descriptive + DistProps
Logistic	0.809	0.842	0.793	Descriptive + Discourse
J48	0.803	0.84	0.787	Only Descriptive
J48	0.795	0.838	0.781	Descriptive + Discourse
J48	0.699	0.836	0.761	Physical + Discourse
Logistic	0.699	0.836	0.761	Physical + and Discourse
ZeroR	0.699	0.836	0.761	All
OneR	0.699	0.836	0.761	All

Table 2: Predicting pointing gestures with different feature combinations in the complete MREDI dataset.

the descriptions produced by speakers were good predictors of pointing.

Adding only DistProps to the descriptive features improved the accuracy of the Logistic classifier somewhat, though it had a greater impact on J48. However, Distance seems to have the greatest impact of the two physical features. Discourse does not appear to play an important role: incorporating this feature does not result in much improvement over using only descriptive features; indeed, in the case of J48, it decreases accuracy.

We also tested one of the best combinations involving descriptive features and Distance but excluding the Deictic feature from the set of descriptive features. This was done because pointing in referential acts is frequently viewed on a par with deictic expressions, insofar as they are both indexical (Bangerter, 2004). This raises the question whether, out of all the descriptive features, Deixis could be considered a somewhat redundant predictor. The results suggest that removing Deixis from the descriptive features does not alter the accuracy of the classifier. We return to the role of Deixis in the discussion in Section 4.3.

4.2 Results on the referential dataset

Exactly the same combinations of features were tested, using 10-fold cross-validation, in separate experiments on the referential dataset. This was done in order to compare the results on a dataset which contains less 'noise', that is, fewer utterances which were non-referential. Such utterances may compromise the predictive validity of certain features, as they inflate the number of utterances in which *Point=0*.

Table 3 contains the results obtained on the reduced dataset. The accuracy is in general lower due to the fact that predicting the absence of pointing is easier in the complete dataset, where many utterances contain no reference at all, descriptive or gestural.

Contrary to the findings on the complete dataset, using the complete set of features as predictors of pointing gives slightly better results than using either descriptive or physical features alone, at least in

Classifier	P	R	F	Features
J48	0.783	0.799	0.785	All - MapConfl
Logistic	0.726	0.764	0.679	All - MapConfl
J48	0.774	0.793	0.776	All
Logistic	0.704	0.760	0.681	All
J48	0.781	0.797	0.784	Descriptive + DistProps + distance
J48	0.766	0.785	0.770	Descriptive + DistProps
J48	0.748	0.777	0.745	Descriptive + Distance
J48	0.758	0.783	0.744	Only descriptive
J48	0.774	0.788	0.740	Descriptive + Discourse
Logistic	0.720	0.762	0.675	Descriptive + DistProps + distance
Logistic	0.688	0.759	0.662	Descriptive + Discourse
Logistic	0.699	0.760	0.661	Descriptive + DistProps
Logistic	0.759	0.761	0.660	Descriptive + Distance
J48	0.577	0.760	0.656	Only physical + Discourse
Logistic	0.577	0.760	0.656	Only descriptive
Logistic	0.577	0.760	0.656	Only physical + Discourse
ZeroR	0.577	0.76	0.656	All
ONeR	0.577	0.76	0.656	All

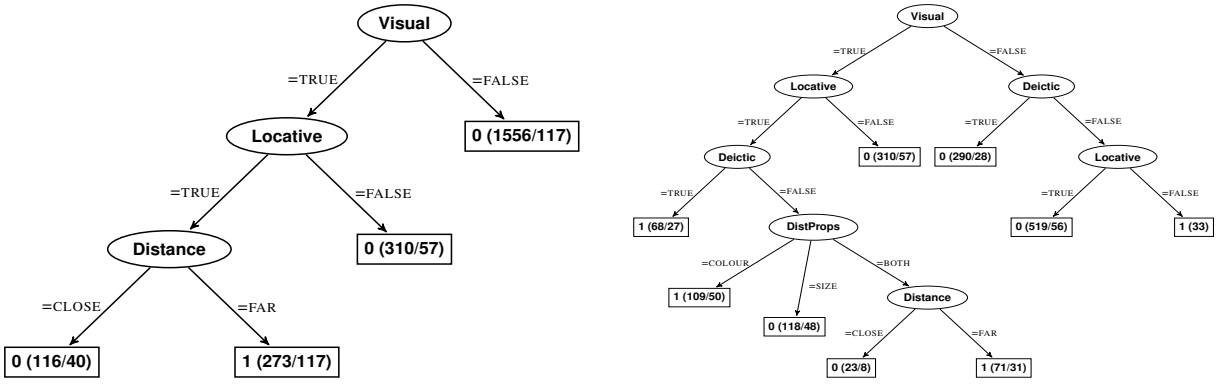
Table 3: Predicting pointing gestures with different feature combinations in the referential MREDI dataset.

the case of the decision tree classifier. This combination also exceeds the combination of descriptives, DistProps and Distance, though only marginally. However, this does remain the next best combination for J48, consistent with the results on the complete dataset. However, this combination performs quite badly in the case of the Logistic classifier.

The fact that using all features performs better this time is probably due to the fact that there are fewer non-referential utterances in this dataset. Once again, the role of Discourse seems marginal.

4.3 Analysis and discussion

Figure 2 shows the decision trees built by J48 for the two datasets when descriptive features are used together with *DistProps* and *Distance*.



(a) Complete MREDI dataset

(b) Referential MREDI dataset

Figure 2: J48 Decision trees from the complete and referential datasets

Our main findings can be summarised as follows. First, descriptive features play an important role in

the prediction of pointing; this replicates previous observations (Gatt and Paggio, 2013). Second, and more importantly, the prediction accuracy improves when physical features, representing aspects of the visual/perceptual context, are taken into account. This is especially true of Distance, suggesting that a sizeable shift of perceptual focus, from one landmark to another further away, motivates a pointing gesture, as shown in both trees in Figure 2. Once again, it is worth comparing this to the results of Beun and Cremers (1998), who find that shifts of perceptual focus play a role in increasing the amount of (descrpitive) information speakers include in a referring expression. However, they find no impact of focus shifts on pointing gestures; our results, by contrast, suggest that such shifts do play a role.

There are a number of striking features in the trees in the figure. First, the descriptive feature Locative plays a crucial role. All cases of pointing involve the presence of a Locative, with one exception: on the referential dataset (Figure 2b), in case no Visual, Deictic or Locative features are used, the tree predicts a pointing gesture. However, this case covers a very small number of instances (33), with 0% error rate. All of these turn out to be utterances where there is no descriptive reference at all and speakers rely exclusively on pointing. Example (3) below is typical of these.

(3) D: And a slightly bigger green to the right of that
M: M-hm
D: In the center of those like pack
M: Yeah
D: is number 9. [+pointing]

Clearly, these are cases in which the pointing gesture occurs as part of an extended sequence of utterances which jointly identify a landmark. Descriptive features have already been uttered; the pointing comes at the very end. In summary, the one case where Locatives don't feature in predicting a pointing gesture turns out to be a rather special case.

A second striking aspect of the trees is that while Deixis plays a predictive role in the tree based on the referential dataset, it doesn't in the case of the complete dataset. This is interesting in view of the relationship that has often been noted between referential pointing gestures and deictic expressions (Bangerter, 2004). Note, however, that there is no inconsistency between the two trees: the single path through the tree in Figure 2a that results in pointing is subsumed by the path in Figure 2b which specifies in addition that Deixis should be `false`, and DistProps should have the value `colour`. This still leaves open the question why Deixis plays no role in the full dataset, despite being included as part of the descriptive features that resulted in this tree. Indeed, we have already shown that, among the descriptive features, Deixis doesn't contribute much predictive power on the full dataset (see Section 4.1).

One possibility is that Deixis is generally under-represented in the corpus. However, there are proportionately fewer utterances in the full dataset containing Deixis (20%), compared to the referential dataset (30%). Furthermore, it may be partially dependent on the Locative features. There may be a priori reasons to assume this as a working hypothesis: Deixis anchors parts of the speech signal to physical properties of the common ground, potentially making it redundant with respect to location (which has already specified the relevant physical/spatial features of the common ground).

	Complete Dataset		**Referential Dataset**	
Locative	*Deictic*		*Deictic*	
	`false`	`true`	`false`	`true`
`false`	74	26	42	58
`true`	88	12	88	12
overall	80	20	70	30

Table 4: Deictic features (D and ID) relative to Locatives. All figures are percentages.

Table 4 displays the distribution of Deictic expressions with respect to Locatives, that is, the proportion of utterances containing a Deictic expression as a function of whether the utterances also contain a Locative expression. The tables shows proportions both for the full and the referential dataset.

Note that when Deictics are used, it is mostly in the absence of a Locative expression. A chi-square test of independence suggests that the frequency of use of Locative and Deictic expressions are not independent (complete dataset: $\chi^2_1 = 63.044, p < .001$; referential: $\chi^2 = 358.21, p < .001$). However, there is a higher proportion of Deictic expressions in the referential dataset (30% overall); this may account for the use of this feature in the decision tree for this dataset (it is more informative). Crucially, the trend in the use of deictic expressions is reversed in the two datasets: when Locative is `false` on the referential dataset, most utterances involve a deictic expression; the reverse is true on the complete dataset.

There is one path through the tree in Figure 2b which seems to contradict the hypothesis that deictic expressions are used in the absence of locatives. There are 68 cases where pointing is used when both Deictic and Locative are `true`. One possibility is that this is caused by our having defined the Deictic feature as `true` whenever there is an actual deictic expression (variable D in Table 1; e.g. *those squares*) or an identity expression (variable ID; e.g. *the same one we saw*). To investigate this further, Table 5 shows a breakdown of the frequencies of the presence or absence of a locative expression, as a function of whether a true deictic (D) or an identity expression (ID) is used in an utterance. Once again, proportions are displayed for both datasets.

True Deictic	**Complete dataset** *Locative*		**Referential dataset** *Locative*	
	`false`	`true`	`false`	`true`
`false`	54	46	26	74
`true`	78	22	76	24

(a) True deictic expressions (D)

Identity	**Complete dataset** *Locative*		**Referential dataset** *Locative*	
	`false`	`true`	`false`	`true`
`false`	56	44	31	69
`true`	68	32	67	33

(b) Identity expressions (ID)

Table 5: Identity (ID) and actual Deictic (D) expressions relative to Locatives. All figures are percentages.

There are two observations that stem from these proportions: First, in line with our earlier observations, there is a greater proportion of true deictic (D) expressions in utterances that contain no locative expression. For example, 78% of utterances in the complete dataset that have no locatives contain a deictic; the corresponding figure in the referential dataset is 76%. The same pattern holds for identity (ID) expressions. Second, out of the utterances that do not contain a locative, the proportion containing a true deictic (D) is greater than the proportion containing an identity expression (ID). This may explain the apparent exception – represented by the path in Figure 2b – to our generalisation that locatives and deictics are redundant with respect to each other, and locatives tend to be avoided if deictics are used. The explanation may lie in the conflation, in the boolean Deictic feature used in our experiments, of true deictics and identity expressions. The path in the decision tree where both Locative and Deictic are `true` may be accounting for utterances in which an identity expression is used, rather than a true deictic.

5 Conclusions and future work

This paper addressed the question of when pointing gestures should be generated, as a function of the features a speaker uses to identify a referent, as well as the features of the context in which an utterance is being produced. The best predictors of pointing are descriptive features, especially locatives, and features of the perceptual context, especially distance from the last referent. The latter is a marker of a shift of perceptual focus, akin to the focus shifts identified by Beun and Cremers (1998). Our study also sheds light on the relationship between pointing and the use of deictic expressions, suggesting that, while the two are often used together, deictics tend to be used more in the absence of locative features.

We also note some limitations of our methodology. Inspection of the results in Tables 2 and 3 shows that the best performing classifiers, though they exceed baselines, do not do so by a wide margin. We believe that one of the main reasons for this is the relative scarcity of pointing gestures in our dataset (as discussed in Section 3), which may have resulted in a sizeable subset of utterances where pointing was relatively straightforward to predict (e.g. based on one feature, as in the `OneR` baseline classifier). This is a limitation we intend to investigate in future work, through a more diverse dataset where pointing

features more strongly. In addition, it is worth noting that the ablative testing reported here does suggest that certain features play a greater role in determining when speakers choose to point.

Our work addresses an important question in Natural Language Generation systems that seek to generate multimodal referring acts, namely, how pointing and describing should be combined and when. In future work, we intend to extend this research in two ways: first, by extending our focus to incorporate the interactive features of a dialogue and their impact on referential success; and second, by focusing on other domains with a view to testing the generalisability of the results.

Acknowledgements

Thanks to Ielka van der Sluis, Adrian Bangerter and Paul Piwek, who collaborated on the development of the MREDI corpus. Thanks to the anonymous reviewers of COLING 2014 for helpful comments.

References

A. Anderson, M. Bader, E. Bard, E. Boyle, G. M. Doherty, S. Garrod, S. Isard, J. Kowtko, J. McAllister, J. Miller, C. Sotillo, H. S. Thompson, and R. Weinert. 1991. The HCRC Map Task corpus. *Language and Speech*, 34:351–366.

E. André and T. Rist. 1996. Coping with temporal constraints in multimedia presentation planning. In *Proceedings of the 13th National Conference on Artificial Intelligence (*AAAI*'96)*.

A. Bangerter. 2004. Using pointing and describing to achieve joint focus of attention in dialogue. *Psychological Science*, 15(6):415–419.

R.J. Beun and A. Cremers. 1998. Object reference in a shared domain of conversation. *Pragmatics and Cognition*, 6(1-2):121–152.

R. Dale and E. Reiter. 1995. Computational interpretation of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(8):233–263.

R. Dale. 1989. Cooking up referring expressions. In *Proceedings of the 27th annual meeting of the Association for Computational Linguistics (ACL'89)*, pages 68–75.

J.P. de Ruiter, A. Bangerter, and P. Dings. 2012. The interplay between gesture and speech in the production of referring expressions: Investigating the tradeoff hypothesis. *Topics in Cognitive Science*, 4:232–248.

N.J. Enfield. 2009. *The Anatomy of Meaning: Speech, Gesture and Composite Utterances*. Cambridge University Press, Cambridge.

A. Gatt and P. Paggio. 2013. What and where: An empirical investigation of pointing gestures and descriptions in multimodal referring actions. In *Proceedings of the 14th European Workshop on Natural Language Generation (*ENLG'13*)*.

S. Kita and A. Özyürek. 2003. What does cross-linguistic variation in semantic coordination of speech and gesture reveal?: Evidence for an interface representation of spatial thinking and speaking. *Journal of Memory and Language*, 48:16–32.

S. Kopp, K. Bergmann, and I. Wachsmuth. 2008. Multimodal communication from multimodal thinking: Towards an integrated model of speech and gesture production. *International Journal of Semantic Computing*, 2(1):115–136.

E. Krahmer and K. van Deemter. 2012. Computational generation of referring expressions: A survey. *Computational Linguistics*, 38(1):173–218.

E. Krahmer, S. van Erk, and A. Verleg. 2003. Graph-based generation of referring expressions. *Computational Linguistics*, 29(1):53–72.

A. Kranstedt and I. Wachsmuth. 2005. Incremental generation of multimodal deixis referring to objects. In *Proceedings of the 10th European Workshop on Natural Language Generation (ENLG'05)*.

D. McNeill and S.D. Duncan. 2000. Growth points in thinking for speaking. In D. McNeill, editor, *Language and Gesture*, pages 141–161. Cambridge University Press.

D. McNeill. 1985. So you think gestures are nonverbal? *Psychological Review*, 92(3):350–371.

P. Piwek. 2007. Modality choice for generation of referring acts: Pointing vs describing. In *Proceedings of the Workshop on Multimodal Output Generation (MOG'07).*, pages 129–139.

I. van der Sluis and E. Krahmer. 2007. Generating multimodal referring expressions. *Discourse Processes*, 44(3):145–174.

I. van der Sluis, P. Piwek, A. Gatt, and A. Bangerter. 2008. Towards a balanced corpus of multimodal referring expressions in dialogue. In *Proceedings of the Symposium on Multimodal Output Generation (MOG'08)*.

I.H. Witten and E. Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, second edition.

Generating Acrostics via Paraphrasing and Heuristic Search

Benno Stein **Matthias Hagen** **Christof Bräutigam**
Bauhaus-Universität Weimar, Germany
<first name>.<last name>@uni-weimar.de

Abstract

We consider the problem of automatically paraphrasing a text in order to find an equivalent text that contains a given acrostic. A text contains an acrostic, if the first letters of a range of consecutive lines form a word or phrase. Our approach turns this paraphrasing task into an optimization problem: we use various existing and also new paraphrasing techniques as operators applicable to intermediate versions of a text (e.g., replacing synonyms), and we search for an operator sequence with minimum text quality loss. The experiments show that many acrostics based on common English words can be generated in less than a minute. However, we see our main contribution in the presented technology paradigm: a novel and promising combination of methods from Natural Language Processing and Artificial Intelligence. The approach naturally generalizes to related paraphrasing tasks such as shortening or simplifying a given text.

1 Introduction

Given some text, paraphrasing means to rewrite it in order to improve readability or to achieve other desirable properties while preserving the original meaning (Androutsopoulos and Malakasiotis, 2010). The paper in hand focuses on a specific paraphrasing problem: rewriting a given text such that it encodes a given acrostic. A text contains an acrostic if the first letters of a range of consecutive lines form a word or phrase read from top to bottom. A prominent and very explicit example of former Governor Schwarzenegger is shown in Figure 1 (see the third and fourth paragraphs). Schwarzenegger himself characterized the appearance of that acrostic a "wild coincidence".[1] However, such a coincidence is highly unlikely: Using the simplistic assumption that first letters of words are independent of each other if more than ten words are in between (line length in the Schwarzenegger letter) and calculating with the relative frequencies of first letters in the British National Corpus (Aston and Burnard, 1998), the probability for the acrostic in Figure 1 can be estimated at $1.15 \cdot 10^{-12}$. Typically, a given text will not contain a given acrostic but has to be reformulated using different wording or formatting to achieve the desired effect. Thus we consider the purposeful generation of acrostics a challenging benchmark problem for paraphrasing technology, which is subject to soft and hard constraints of common language usage.

The paper shows how heuristic search techniques are applied to solve the problem. Different paraphrasing techniques are modeled as operators applicable to paraphrased versions of a text. By pruning the so-formed search space and by employing a huge corpus of text n-grams for the possible operators, we are able to generate acrostics in given texts. Our algorithmic solution is a novel combination of techniques from Natural Language Processing and Artificial Intelligence. We consider such combinations as a very promising research direction (Stein and Curatolo, 2006; Sturtevant et al., 2012), and we point out that the problem of acrostic generation serves as a serious demonstration object: the presented model along with the heuristic search approach generalizes easily to other paraphrasing tasks such as text shortening, improving readability, or e-journalism.

2 Related Work and Problem Definition

Rewriting a given text in order to "encode" an acrostic is a paraphrasing problem, which in turn is studied in the domain of computational linguistics and natural language processing. We review relevant literature

[1] www.huffingtonpost.com/2009/10/30/schwarzenegger-f-bomb-in_n_340579.html, last accessed: June 12, 2014.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2018–2029, Dublin, Ireland, August 23-29 2014.

To the Members of the Californian State Assembly:

I am returning Assembly Bill 1178 without my signature.

For some time now I have lamented the fact that major issues are overlooked while many unnecessary bills come to me for consideration. Water reform, prison reform, and health care are major issues my Administration has brought to the table, but the Legislature just kicks the can down the alley.

Yet another legislative year has come and gone without the major reforms Californians overwhelmingly deserve. In light of this, and after careful consideration, I believe it is unnecessary to sign this measure at this time.

Sincerely,
Arnold Schwarzenegger

Figure 1: Excerpt from a letter of former Governor Arnold Schwarzenegger to the Californian State Assembly in October 2009. The third and fourth paragraphs contain the acrostic "F∗∗∗ You".

of the topic and highlight techniques that will be employed in our work.

An important branch of the paraphrasing literature focuses on analyses with fixed corpora. Such corpora typically are *parallel* in the sense that they contain different formulations of the same facts (Barzilay and McKeown, 2001; Barzilay and Lee, 2003; Callison-Burch, 2008). These "facts" can be news articles on the same event (Clough et al., 2002; Dolan and Brockett, 2005), different translations of a source text to a target language (Pang et al., 2003), or cross-lingual parallel corpora (Bannard and Callison-Burch, 2005). As most of the early parallel corpora were constructed manually (especially the judgments of whether a pair of sentences forms a paraphrase), there are two shortcomings. First, the obtained paraphrases are usually specific to the domain covered in the corpus (e.g., showbiz news) and often do not generalize well. Second and probably more severe is the fact that manually building parallel corpora is very expensive, such that the available ones are rather small: the METER corpus contains only 1717 texts on legal issues and showbiz (Clough et al., 2002), the MSRP corpus contains only 5801 sentence pairs (Dolan and Brockett, 2005). Recently, new methods employ machine learning techniques to automatically build larger paraphrase collections from parallel corpora (Ganitkevitch et al., 2011; Ganitkevitch et al., 2013; Metzler et al., 2011; Metzler and Hovy, 2011). We include the paraphrase database (Ganitkevitch et al., 2013)—a database of extracted patterns from such large scale corpora—as one source of potential paraphrases in our algorithm.

Compared to the large body of literature that "extracts" paraphrases from (parallel) corpora, there is relatively little work on automatically paraphrasing a given text. Some of the early generation methods are based on rules that encode situations wherein a reformulation is possible (Barzilay and Lee, 2003). A problem with rules is that often the rather complicated patterns extracted from text corpora are hardly applicable to a given to-be-paraphrased text: manually created corpora are simply too small and machine generated paraphrasing rules often do not match in a given text. Other early methods use machine translation (Quirk et al., 2004). However, the need for large and expensive parallel manual translation corpora cannot be circumvented by using multiple resources (Zhao et al., 2008).

Another branch of paraphrasing methods is based on large thesaurus resources such as WordNet (Fellbaum, 1998). The idea is to insert synonyms into a text when the context fits (Bolshakov and Gelbukh, 2004; Kauchak and Barzilay, 2006). The most recent approaches are statistics-based (Chevelu et al., 2009; Chevelu et al., 2010; Zhao et al., 2009; Burrows et al., 2013).

Compared to the existing research, we have a more difficult use case here. Existing paraphrase generation focuses on sentence paraphrasing, while we have to consider a complete text that has to be rewritten/reformatted in order to contain a given acrostic. We will employ the above shown state-of-the-art paraphrasing procedures as "operators" in our approach. This new problem setting of applying different operators to a complete text forms a search problem with a huge search space. In order to deal with this search space, we apply powerful search heuristics from Artificial Intelligence. The combination of heuristic search with established text level paraphrasing techniques represents a new approach to tackle problems in computational linguistics. The acrostic generation problem is defined as follows:

ACROSTIC GENERATION

Given: (1) A text T and an acrostic x.
(2) A lower bound $l_{\min}$ and an upper bound $l_{\max}$ on the desired line length.

Task: Find a paraphrased version T^* of T in monospaced font that encodes x in some of its lines when possible. Each line of T^* has to meet the length constraints.

3 Modeling Paraphrasing as Search Problem

This section shows how to model paraphrasing in general and ACROSTIC GENERATION in particular as a search problem (Pearl, 1984). The search space is a universe $\mathcal{T}$ of candidate texts for which we can devise, at least theoretically, a systematic search strategy: if $\mathcal{T}$ is finite, each element $n \in \mathcal{T}$ is analyzed exactly once. The elements in $\mathcal{T}$ represent states (nodes), and there is a limited and a-priori known set of possibilities (edges, paraphrasing operators) to get from a node n to an adjacent or successor node n_i. A paraphrasing operator ϕ provides a number of parameters that control its application. Each state $n \in \mathcal{T}$ is considered an acrostic (sub)problem; the dedicated state $s \in \mathcal{T}$ represents the original problem while $\Gamma \subset \mathcal{T}$ is the set of solution nodes that have no problem associated with. The following subsections will outline important properties of the search space and introduce a suited cost measure to control the exploration of $\mathcal{T}$.

3.1 Search Space Structure

Solving an instance of ACROSTIC GENERATION under a so-called state-space representation means to find a *path* from s, which represents the original text T, to some goal state $\gamma \in \Gamma$. The problem of finding an acrostic consists of tightly connected subproblems (finding subsequences of the acrostic) that cannot be solved independently of each other. Most puzzles such as Rubik's cube are of this nature: changing a decision somewhere on the solution path will affect all subsequent decisions. By contrast, a so-called problem-reduction representation will exploit the fact that subproblems can be solved independently of each other. Many tasks of logical reasoning and theorem proving give rise to such a structure: given a set of axioms, the lemmas required for a proof can be derived independently, which in turn means that the sought solution (a plan or proof) is ideally represented by a *tree*.

Searching $\mathcal{T}$ under a state-space representation means to unfold a tree whose inner nodes link to successor nodes that encode alternative decisions; hence these inner nodes are also called OR-nodes. Each path from s that can be extended towards a goal state γ forms a *solution candidate*. Similarly, searching $\mathcal{T}$ under a problem-reduction representation also means to unfold a tree—however, the tree's inner nodes must be distinguished as AND-nodes and OR-nodes, whereas the successors of an AND-node encode subproblems all of which have to be solved. A solution candidate then is a tree comprised of (1) the root s, (2) OR-nodes with a single successor, and (3) AND-nodes with as many successors as subproblems all of which are characterized by the fact of being extensible towards goal states in Γ. Figure 2 contrasts both search space structures.

Under either representation, OR-graphs as well as AND-OR-graphs, the application of a sequence of

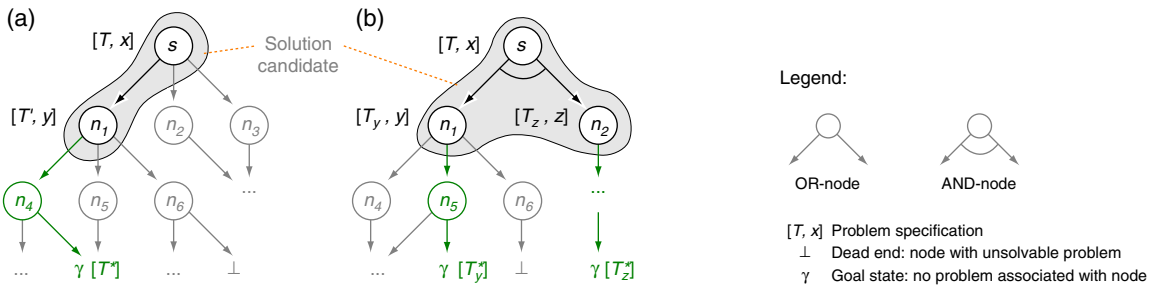


Figure 2: (a) State-space representation (OR-graph) versus (b) Problem-reduction representation (AND-OR-graph). OR-nodes encode alternative decisions, while AND-nodes decompose a problem into subproblems all of which are to be solved.

operators will easily lead to situations where states are revisited—precisely: are generated again. Search algorithms maintain so-called OPEN- and CLOSED-lists to manage the exploration status of generated nodes. However, because of the intricate state encoding, which must inform about the effect of all applied operators from s to an arbitrary node, the exponentially growing number of nodes during search, and the necessity of efficiently querying these lists, sophisticated data structures such as externalized hash tables and key value stores are employed. Typically, these data structures are tailored to the problem domain (here: to paraphrasing), and they model heuristic access strategies to operationalize a probabilistically controlled trade-off between false positive and true negative answers to state queries.

We have outlined the different search space structures since ACROSTIC GENERATION may show an OR-graph puzzle nature at first sight: paraphrasing at some position will affect all following text. Interestingly, there is a limited possibility to introduce "barriers" in the text, which allows for an AND-OR-graph modeling and hence for an isolated subproblem treatment. Examples include paraphrasing operators that do not affect line breaks, or acrostics consisting of several words and thus spanning several paragraphs.

Since in general the underlying linguistic considerations for the construction and maintenance of such barriers are highly intricate and complex, we capture this structural constraint probabilistically as shown in Equation (1). The equation models the problem difficulty or *effort for acrostic generation*, E, and introduces $P_{y \circ z}(|x|)$, which quantifies the probability for the event that an acrostic x can be treated independently as two partial acrostics y and z, where $x = yz$.

$$E(x) = \begin{cases} e(x) & \text{If } |x| = 1. \\ P_{y \circ z}(|x|) \cdot \Big(E(y) + E(z)\Big) + (1 - P_{y \circ z}(|x|)) \cdot E(y) \cdot E(z) & \text{If } |x| > 1. \end{cases} \tag{1}$$

Remarks. The effort for generating a single-letter acrostic of length 1 is $e(x)$, with $e(x) \geq 1$. Based on $e(x)$, we recursively model the *true effort* $E(x)$ for generating an acrostic $x = yz$ as follows: as additive effort if the generation of the acrostics y and z can be done independently, and as multiplicative effort otherwise. Observe how $P_{y \circ z}$ controls the search space structure: if $P_{y \circ z} = 0$ for all partitionings of x into y and z, one obtains a pure state-space representation for ACROSTIC GENERATION. Similarly, the other extreme with $P_{y \circ z} = 1$ results in a series of $|x|$ letter generation problems that can be solved independently of each other.

As an estimate $\hat{e}(x)$ for $e(x)$ we suggest the multiplicative inverse of the occurrence probabilities of the first letters in the English language, as computed from the British National Corpus (BNC). The BNC is a 100 million word collection of written and spoken language from a wide range of sources, designed to represent a wide cross-section of current British English (Aston and Burnard, 1998). The BNC probabilities vary between 0.115719 for the letter "s" and 0.00005 for the letter "z". As an estimate for $P_{y \circ z}$ we suggest the $N(5, 0.5)$ distribution to model the paragraph lengths in T or, equivalently, the number of characters of the (English) words in x. These choices give rise to $\hat{E}(x)$, the *estimated effort* for generating an acrostic x. $\hat{E}(x)$ is used to assess the (residual) problem complexity and, under a maximum likelihood approach, models the expected search effort. There is a close relation between the effort estimate $\hat{E}$ and the quality estimate $\hat{Q}$ introduced below, which will be exploited later on, in Equation (3).

3.2 Cost Measure Structure

Cost measures—equivalently: merit measures—form the heart of systematic search strategies and determine whether an acceptable solution of a complex problem can be heuristically constructed within reasonable time. Here, we refer to general best-first search strategies as well as variants that relax strict admissibility. As a working example consider the following text T about Alan Turing taken from the English Wikipedia, where the task is to generate the acrostic $x = \texttt{Turing}$ with $l_{\min} = 55$ and $l_{\max} = 60$.

```
Alan Mathison Turing was a British mathematician, logician,
cryptanalyst and computer scientist. He was highly influential in
the development of computer science, giving a formalization of the
concepts of algorithm and computation with the Turing machine,
which can be considered a model of a general purpose computer.
```

A possible solution T^* (a paragraph's last line may be shorter than $l_{\min}$):

```
The British mathematician Alan Mathison Turing was also an
unrivaled logician, cryptanalyst and computer scientist. He
revolutionized the development of computer science, giv-
ing a formalization of the concepts of algorithm and defi-
nite computation with the Turing machine, which can be re-
garded a model of a general purpose computer.
```

T^* is of a high quality though it introduces an exaggerating tone, this way violating Wikipedia's neutrality standard. Also note that the applied paraphrasing operators vary in their quality, which is rooted in both the kind and the context of the operators. Table 1 (left) shows a selection of the operators, some of which are applied in a combined fashion. Section 4 introduces the operators in greater detail.

To further formalize the quantification of a cost measure C or a merit measure Q, we stipulate on the following properties:

1. The quality of the original text T cannot be improved. Each paraphrasing operator ϕ introduces unavoidable deficiencies in T.
2. The overall quality of a solution T^* depends on the quality of all applied paraphrasing operators.
3. Following readability theory and relevant research, the severity of text deficiencies—here introduced by a paraphrasing operator ϕ—has a disproportionate impact on the text quality (Meyer, 2003).
4. To render different problems and solutions comparable, the achieved quality of a solution T^* has to be normalized.

Equation (2) below shows the basic structure of Q, the proposed, unnormalized merit measure. Its optimization yields Q^*. $Q^*(n)$ assigns to a node $n \in \mathcal{T}$ the maximum paraphrasing quality of a text T^* that contains the partial acrostic associated with n. Likewise, $Q^*(s)$ characterizes the quality of the optimum solution for solving ACROSTIC GENERATION.

$$\frac{1}{Q^*(n)} = \begin{cases} 0 & \text{If } n \in \Gamma. \\ \min\limits_i \left\{ \dfrac{1}{q(n,n_i)} + \dfrac{1}{Q^*(n_i)} \right\} & \text{Otherwise.} \end{cases} \tag{2}$$

Remarks. The state (node) n_i denotes a direct successor of the state (node) n in the search space $\mathcal{T}$. Associated with n_i is a text resulting from the application of a paraphrasing operator ϕ to the text associated with n, whereas $q(n, n_i)$ quantifies the *local quality* achieved with ϕ. The measure in Equation (2) is both of an additive form and formulated as a minimization problem. As shown in the following, it can be reformulated for a best-first algorithm scheme, ensuring admissibility under a delayed termination condition. Also note that the merit measure operationalizes the above Property 3 via the harmonic mean computation. Accordingly, we obtain a normalized *overall quality* $\bar{Q}^*$ given an acrostic x as $\bar{Q}^* = |x| \cdot Q^*$.

To turn Equation (2) into actionable knowledge, the quality $q(n, n_i)$ of a paraphrasing operator ϕ when moving from n to a successor n_i needs to be quantified. We employ for q the domain $[0; 1]$, where 0 and 1 encode the worst and best achievable quality respectively. By construction the normalized quality $\bar{Q}^*$ will then lie in the interval $[0; 1]$ as well, thus greatly simplifying the interpretation of the measure.

Table 1 (right) shows values for the local quality of the operators in the Alan Turing example, which are derived from linguistic quality considerations and the experimental analysis detailed in Section 5. The comment column argues the linguistic meaningfulness. If we agree on $q = 1.0$ for the first two lines of the generated acrostic $x =$ `Turing` and recursively apply the merit measure defined in Equation (2), we obtain $Q = 0.127$ as unnormalized and $\bar{Q} = |x| \cdot Q = 0.76$ as normalized overall quality.

To make Equation (2) applicable as cost estimation heuristic $f(n)$ in a best-first algorithm scheme, Equation (3) below unravels its recursive structure in the usual way as $f(n) = g(n) + h(n)$. The semantics is as follows: under an optimistic estimate $h(n)$ (= underestimating costs or overestimating merits) the

Table 1: Left: Paraphrasing operators in the Alan Turing example. Right: Values for the local quality of the respective operators, which entail the normalized overall quality $\bar{Q} = 0.76$ for the example.

Line	Operator ϕ	Text → paraphrased text	$q(n, n_i)$	Comment
3	synonym	highly influential → revolutionized	0.9	stylistically well, exaggerating tone
4	hyphenation	giving → giv-ing	0.6	unexpected hyphen for a short word
5	tautology	computation → defi-nite computation	0.6	tautology arguable, hyphen unusual
6	synonym	considered → re-garded	0.7	synonym suited, hyphen acceptable

total cost (the overall quality) for solving ACROSTIC GENERATION via a path *along node* n is always larger (smaller) than $f(n)$. In particular, $g(n)$ accumulates the true cost (the achieved quality) for the partial acrostic via a concrete path $s = n_0, n_1, \ldots, n_k = n$, while $h(n)$ gives an underestimation of the cost (overestimation of the quality) for the remaining part of the acrostic. Observe that the additive form of Equation (2) guarantees the parent discarding property (Pearl, 1984), which states that no decision on a path from n to a goal state γ can change the value for $g(n)$.

A tricky part is the construction of $h(n)$, which, on the one hand, may ensure admissibility, while, on the other hand, should be as close as possible to the real cost. Here, the measure $E(x)$ for the problem difficulty from Equation (1) comes into play, which models the problem decomposability and which informs us about the largest remaining subproblem (= the depth of the deepest remaining OR-graph) when solving x. Without loss of generality, admissibility is ensured if (a) the probability $P_{y \circ z}$ used in $\hat{E}(x)$ is biased towards decomposability, and if (b) we assume that the remaining acrostic x can be solved by always applying the cheapest (maximum quality-preserving) operator $q_{\max}$.

$$\underbrace{\frac{1}{\hat{Q}(n)}}_{f(n)} = \underbrace{\sum_{i=1}^{k} \frac{1}{q(n_{i-1}, n_i)}}_{g(n)} + \underbrace{\log_K \left(\hat{E}(\tau(n)) \right) \cdot \frac{1}{q_{\max}}}_{h(n)}, \quad \text{where } n_0 = s,\ n_k = n \tag{3}$$

Remarks. $\tau(n)$ denotes the remaining acrostic x that is associated with node $n \in \mathcal{T}$. The logarithm base K serves for normalization purposes with regard to the BNC letter frequencies $\hat{e}(x), |x| = 1$, which are used within $\hat{E}(x)$ in Equation (1). We define K as the multiplicative inverse of the occurrence probability of the least frequent letter in the remaining acrostic $x = \tau(n)$, which gives rise to the inequality $\log_K(\hat{E}(x)) \leq |x|$. This choice entails two properties: (1) it underestimates the remaining acrostic length and hence ensures the admissibility characteristic of $h(n)$, and, (2) it yields an increasing accuracy of $h(n)$ when approaching a goal state in Γ. Finally, we can substitute 1.0 as an upper bound for $q_{\max}$, again preserving the admissibility of $h(n)$.

Admissibility, i.e., the guarantee of optimality during best-first search, may not be the ultimate goal: if $h(n)$ underestimates costs (overestimates merits) too rigorously, best-first search degenerates to a kind of breadth-first search—precisely: to uniform-cost search. Especially if computing power is a scarce resource, we may be better off with a depth-preferring strategy. Observe that the logarithm base K in Equation (3) provides us a means to smoothly vary between the two extremes, namely by choosing K from $[K_{\min}; K_{\max}]$, where $K_{\min}$ ($K_{\max}$) specifies the multiplicative inverse of the occurrence probability of the most (least) frequent letter in the remaining acrostic $x = \tau(n)$.

4 Paraphrasing Operators

Most of the following operators used in our heuristic search process employ state-of-the-art linguistic tools or are based on standard knowledge from Wikipedia. Table 2 shows information about the role of individual operators in our experiments from Section 5; the table illustrates also the effort for preparing (offline) and applying (online) the operators.

4.1 Context-Independent Operators

Line break Since we are dealing with text that should spread over several lines, breaking between lines is one of the most basic operators. Similarly, it is the most efficient operator, and Column 4 of Table 2 illustrates the performance of the others in relation to this operator. Line breaks are possible at the end of sentences (i.e., a paragraph break), while a line break in between words is only possible if it falls in the $[l_{\min}; l_{\max}]$-window given by the line length constraints.

Hyphenation Related to line breaks are hyphenations. We re-implemented and employ the standard TEX hyphenation algorithm (Knuth, 1986). Analogous to line breaks, hyphenation is applicable if the line after hyphenating (and line breaking) has a length in the $[l_{\min}; l_{\max}]$-window.

Function word synonym Specific groups of so-called synsemantic words can often be replaced by each other without changing a text's meaning. We have identified 40 such groups from a list of Sequence Publishing[2] and the Paraphrase Database (Ganitkevitch et al., 2013). Examples are {can, may}, {so, thus, therefore, consequently, as a result}, and {however, nevertheless, yet}.

Contraction and expansion Some local text changes can be achieved by contracting or expanding formulations like "he'll" or "won't". We have identified 240 such pairs from Wikipedia.[3] Other possibilities are to spell out / contract standard abbreviations and acronyms. We have mined a list of several thousand such acronyms from the Web.[4] Finally, also small numbers can be spelled out or be written as numerals (e.g., "five" instead of "5"). It is interesting to note that this operator was hardly ever used on successful paths in our experiments.

Spelling In principle, we want to generate text that is correctly spelled. In certain situations, however, it can nevertheless be beneficial to introduce some slight mistakes in order to change word lengths or to generate letters not present in the correctly spelled text. We employ a list of 3 000 common misspellings mined from Wikipedia[5] (e.g., "accidently" instead of "accidentally"). We also include several standard typos related to computer keyboards (e.g., an "m" is often typed as an "n" and vice versa) as well as phonetic misspellings (e.g., "f" and "ph" often sound similar). Since the quality score of wrong spellings tends to be low, this operator has to be treated with care. Especially at the beginning of words, typos are less common than within words such that we allow typos only within words.

Wrong hyphenation Similar to wrong spellings is the purposeful choice of a wrong hyphenation. As with wrong spellings the quality score is typically low. We thus employ this operator very carefully, avoiding for instance syllables on a new line with just two letters. Analogous to correct hyphenation, the line length has to be in the $[l_{\min}; l_{\max}]$-window to apply wrong hyphenation. Despite its questionable quality, this operator is used pretty often in the experiments since it has a very high probability of "generating" a desired letter.

4.2 Context-Dependent Operators

Synonym For identifying synonyms, WordNet's synsets (Fellbaum, 1998) is used. Since only a small subset of the synset members of a to-be-replaced word w is reasonable in the context around w in T, we check in the Google n-gram corpus (Brants and Franz, 2006) whether the synonym in fact fits in the same context. In this regard the public and highly efficient Netspeak API (Stein et al., 2010; Riehmann et al., 2012) is employed. For example, given "hello world", the most frequent phrase with a synonym for world is "hello earth". The Google n-grams are up to five words long, such that at most four context words can be checked before or after w. Previous studies showed that more context yields higher quality synonyms (Metzler and Hovy, 2011; Pasca and Dienes, 2005), so that we use at least two words before or after w. Higher quality scores are achieved if the context is matched before as well as after w.

[2] `sequencepublishing.com/academic.html\#function-words`, last accessed: June 12, 2014

[3] `en.wikipedia.org/wiki/List_of_English_contractions\#English`, last accessed: June 12, 2014, `en.wikipedia.org/wiki/English_auxiliaries_and_contractions`, last accessed: June 12, 2014

[4] `www.acronymfinder.com`, last accessed: June 12, 2014

[5] `en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings/For_machines`, last accessed: June 12, 2014

Table 2: Statistics for the applicability, usage, and effort of single operators. "Application probability" reports whether an operator is applicable at all at some node, "Usage" reports the application probability on a solution path, "Effort" reports the (online) application effort as multiple of the fastest operator (the Line break operator), and "Offline time" reports the preprocessing time in ms per word before the actual search is started. All numbers are profiled within the experiment setup described in Section 5.

Paraphrasing operator	**Application probability** (in %)	**Usage probability** (in %)	**Effort** (multiple of Line break)	**Offline time** (in ms per word)
Line break	16.14	21.13	1.00	0.00
Hyphenation	5.48	9.38	1.22	1.38
Function word synonym	1.43	2.57	1.33	0.01
Contraction and expansion	0.29	0.00	1.93	0.02
Spelling	6.98	1.57	1.18	0.11
Wrong hyphenation	9.53	37.63	1.07	1.38
Synonyms	16.69	2.47	1.66	23.24
Word insertion or deletion	43.46	25.24	2.46	42.75
Average	**12.50**	**12.50**	**1.49**	**14.35**

Word insertion or deletion Similar to the synonym replacement, the insertion or deletion of short phrases is handled. For all positions of the given text, the Google n-grams are checked with Netspeak (see above) for a word w that sufficiently often appears within the context of the text. Similarly, for each word w in the text, it is checked whether there are sufficiently many n-grams without the word but the same surrounding context. In both cases, w is a candidate to be inserted or deleted. Given "hello world", the most frequently used intermediate word is "cruel", yielding the phrase "hello cruel world." Again, as with synonyms, context size and quality are positively correlated. We thus use at least two words as context and favor variants that match more context.

4.3 Further Operator Ideas

In pilot experiments, also the three operator ideas discussed below were analyzed. The ideas show promising results for specific cases, but they easily lead to unexpected text flows due the introduction of odd sentences or names. The operators require future work to better fit them in the given text's context, and they are not employed within the experiments in Section 5.

Tautology It is often possible to introduce entirely new phrases or sentences in a text, which may confirm a previous sentence or which introduce a (nearly) arbitrary but true statement. We tested a small list, including among others "As a matter of fact this is true." or "I didn't know that until now." However, due to improper context such tautologies may mess up a text significantly.

Sentence beginning The beginning of a sentence can often be modified without changing its meaning. Possibilities include the addition of function words like "in general" or "actually", but also the addition of a prefix like "⟨someone⟩ said that …" or "⟨time⟩ ⟨someone⟩ said that …" where ⟨someone⟩ is to be replaced by a person's name or {I, he, she} depending on the full context of the text (e.g., author's name or gender of name mentioned before). The ⟨time⟩ expression may expand to "yesterday" or "last week", etc. Especially with the usage of names, a whole bunch of letters can be generated. However, context is more subtle for this operator compared to the usage of Google n-grams.

Full PPDB The paraphrase database (Ganitkevitch et al., 2013) comes in different sizes and quality levels. Many synonymity relations for nouns are already covered by WordNet, and function word replacements are already an operator on their own. Still, the rich variety of the full data set can form a semantically strong operator. However, in our pilot experiments, the full PPDB patterns often decreased text quality unacceptably, such that we refrained to use PPDB as a single operator in our experiments.

5 Experimental Evaluation

Goal of the evaluation is to show that our approach is able to efficiently generate acrostics in different situations. In this regard, we analyze the general success of acrostic generation, the influence of different operators, and effects on the text quality.

5.1 Experiment Setup

To model different "use cases" in which acrostics have to be inserted, we use texts of different genres: newspaper articles, emails, and Wikipedia articles. We sample 50 newspaper articles from the Reuters Corpus Volume 1 (Lewis et al., 2004), 50 emails from the Enron corpus (Klimt and Yang, 2004), and 50 articles from the English Wikipedia. Each text contains at least 150 words excluding tables and lists.

As target acrostics for all of the above text types, the 25 most common adjectives, nouns, prepositions, verbs, and 50 other common English words are chosen (in total 150 words).[6] This scenario reflects the inclusion of arbitrary words. Other target acrostics are formed by the 100 most common male and female first names from the US (in total 200 words).[7] This models the standard poetry usage of acrostics where often a writer's name is encoded. For all input texts, we also model self-referentiality by using a text's first phrases as the target acrostics (in total 150 phrases for which at least the first word has to be generated). In these cases, the first letter of the acrostic is also the first letter of the text—a fact that enables the controlled evaluation of the importance of the producibility of the first letter.

The evaluation system is a standard quad-core PC running Ubuntu 12.04 with 16 GB of RAM. A relevant subset of all operator application possibilities is preprocessed and stored in-memory (e.g., the synonym n-gram frequencies for every word), whereas the preprocessing time (about one minute in total per run) is not counted for the search process. We then conduct an A* search using the preprocessed operator tables and an admissible instance of Equation (3). To safe runtime, we slightly transform the problem setting and require the acrostic to start at the beginning of the given text. Pilot experiments show that a good choice for line lengths is $l_{\min} = 50$ and $l_{\max} = 70$. Note that this is only slightly more flexible than a standard line length between 55 and 65 characters (i.e., about 10-12 words) but eases acrostic generation. The experiments also reveal that a successful run (the acrostic can be generated) usually takes less than 30 seconds for the search part. An unsuccessful run (the acrostic cannot be generated) takes five to ten minutes until its termination caused by the memory constraints for the open list.

5.2 Experiment Discussion

Given our hardware and time restrictions, about 20% of the runs are successful altogether. The producibility of the first letter is critical for the overall success: we observe an almost 90% success rate for the self-referential acrostics compared to the about 20% for all others. Statistics for the successful runs are given in Table 3. As can be seen, our system is able to generate about 90 000 nodes with 550 goal checks per second. This yields reasonable answer times on the test acrostics: the average number of goal checks needed when the acrostic can be generated is below 10 000 (about 20 seconds of runtime). Only very few successful runs took more than 40 seconds; the self-referential acrostics that often are two or three words long form the main exception. Not that surprisingly, shorter acrostics are on average generated faster than longer ones. Interestingly, besides self-referential acrostics, male first names seem to be the most difficult acrostics when taking the required runtime into account. Note in this regard that many of the (longer) female names start with a more common first letter, which can be generated faster.

Since our approach is the first attempt at the problem of acrostic generation, we cannot compare to other systems from the literature. Instead, we compare to a baseline system that can only use line breaking and hyphenation as its operators. This also helps to further examine the effect of the producibility of the first letter. Whenever the acrostic's first letter is not the first letter of the text, the baseline fails right from the start: recall that for our experiments we require the acrostic to start at the text's beginning. For less than 1% of our test cases, the baseline can generate the acrostic. Most of these few cases are self-referential first words. Even if the first letter is already present, usually the second or third one are not producible by

[6] `en.wikipedia.org/wiki/Common_English_words`, last accessed: June 12, 2014.

[7] `www.ssa.gov/OACT/babynames/decades/century.html`, last accessed: June 12, 2014.

Table 3: Experimental results for complete acrostic generations. For each of the acrostic types (Column 1), several thousand runs were conducted for which we report averaged values of the acrostic lengths in letters (Column 2). The columns 3-10 relate to successful generations and report averaged values for the runtime in seconds, size of the explored search tree, generated nodes and goal checks per second, used main memory, and the quality change according to the introduced measures.

Acrostic type	**Length** (in letters)	**Runtime** (total in s)	**Nodes** (total)	**Nodes** (per s)	**Goal checks** (per s)	**Memory** (in MByte)	**Quality-related measures**		
							Δ WFC	**Δ ARI**	**Δ SMOG**
Common English words									
Adjective	4.36	3.25	286 960	88 269	578	270	-0.99	-1.61	-0.91
Noun	4.47	3.40	285 016	83 837	576	277	-0.39	-0.96	-0.50
Preposition	3.44	3.16	280 593	88 853	556	243	-1.59	-2.28	-1.29
Verb	3.59	2.76	251 161	90 898	595	236	-0.95	-1.60	-0.92
Other	3.29	2.41	218 974	90 755	601	206	-1.10	-2.05	-1.11
Common US first names									
Male	6.00	9.32	851 665	91 368	554	649	-0.74	-1.87	-0.93
Female	6.07	7.82	740 418	94 693	546	575	-0.60	-1.77	-0.93
Self-referential									
First words	10.33	36.09	3 164 873	87 690	518	1 985	-0.31	-0.09	0.20
Average	**5.19**	**8.53**	**759 957**	**89 545**	**565**	**372**	**-0.83**	**-1.53**	**-0.80**

the simple operators. Using all operators, our approach is able to produce a self-referential acrostic of more than seven characters in 80% of the cases. On average, acrostics of ten characters are possible for the self-referential cases. This further highlights the importance of the first letter: whenever it is producible, the success ratio is much higher.

To compare the importance of the different operators, we count for the successful generations in the experiments of Table 3, how often operators are used and how long the search paths are. Table 2 contains information on the applicability of the different operators. About 21% of the operator applications are line breaks, another 9% are hyphenations. Interestingly, about 38% of the operator applications are wrong hyphenations despite the low quality of this operator. Even though our heuristic tries to avoid wrong hyphenations, there are a lot of situations where all other operators fail. Although not reflected by standard quality metrics (see next subsection), a wrong hyphenation usually is eye-catching for human readers, which gives rise to a desirable further quality improvement that should be aimed for in future work. The other operator usages are mostly word insertions and deletions (about 25%), synonym replacements (3%), and function words (3%). The context-independent operators of contractions and spelling correspond to only about 1% of all operator applications.

5.3 Quality-Related Analysis

Table 3 also contains information about the text quality before and after generating the acrostic. To algorithmically measure text quality-related effects, we employ a word frequency class analysis and a readability analysis.

The frequency class WFC(w) of a word w relates to its customariness in written language and has successfully been employed for text genre analysis (Stein and Meyer zu Eißen, 2008). Let $\varphi(w)$ denote the frequency of a word in a given corpus; then the Wortschatz[8] defines WFC(w) as $\lfloor \log_2(\varphi(w^*)/\varphi(w)) \rfloor$, where w^* denotes the most frequently used word in the respective corpus. Here we use as reference the Google n-gram corpus (Brants and Franz, 2006) whose most frequent word is "the", which corresponds to the word frequency class 0; the most uncommonly used words within this corpus have a word frequency class of 26. The readability of the text before and after acrostic generation is quantified according to the standard ARI (Smith and Senter, 1967) and SMOG (McLaughlin, 1969) measures, implemented in the Phantom Readability Library.[9] Both measures have been designed to estimate the U.S. grade

[8] `wortschatz.uni-leipzig.de`, last accessed: June 12, 2014.

[9] `http://niels.drni.de/s9y/pages/phantom.html`, last accessed: June 12, 2014

level equivalent to the education required for understanding a given text. Hence, larger readability scores indicate more difficult texts. The Automated Readability Index ARI (Smith and Senter, 1967) is designed for being easily automatable and uses only the number of characters (excluding whitespace and punctuation), words, and sentences in the text (delimited by a period, an exclamation mark, a question mark, a colon, a semicolon, or an ellipsis). The Simple Measure of Gobbledygook SMOG (McLaughlin, 1969) includes the number of words with more than three syllables, so-called polysyllables.

$$\mathrm{ARI} = 4.71 \cdot \frac{\text{Characters}}{\text{Words}} + 0.5 \cdot \frac{\text{Words}}{\text{Sentences}} - 21.43 \qquad \mathrm{SMOG} = 1.0430 \cdot \sqrt{30 \cdot \frac{\text{Polysyllables}}{\text{Sentences}}} + 3.1291$$

Of course, the above three measures cannot capture text quality as human judges would perceive it. Still, they have their merits and can indicate interesting trends: On average, the texts after acrostic generation use more common words (cf. the negative Δ WFC) and are easier to read (cf. the negative Δ ARI and Δ SMOG) for almost all acrostic types. Thus, one may argue that the quality is not harmed too much; still some issues like wrong hyphenation are ignored by the metrics (cf. the above discussion of individual operators). A deeper analysis of operator quality and improved quality of paraphrased texts (e.g., further operators or avoiding wrong hyphenations) constitute very promising directions for future work.

6 Conclusion and Outlook

We have presented the first algorithmic approach to acrostic generation. The experiments show that the heuristic search approach is able to generate about 20% of the target acrostics in reasonable time, whereas the producibility of the first letter plays a key role. Our solution successfully combines paraphrasing techniques from Natural Language Processing with a heuristic search strategy from Artificial Intelligence. This way, our problem modeling opens a novel and very promising research direction, and the application of our framework to other paraphrasing problems is the most interesting line of future work. As for the acrostic use case, the resulting text's quality gives the most obvious possibility for improvements. We plan to further analyze better quality measures for the individual operators and to develop more sophisticated operators like changing a text's tense or even anaphora exploitation (Schmolz et al., 2012).

References

Ion Androutsopoulos and Prodromos Malakasiotis. 2010. A Survey of Paraphrasing and Textual Entailment Methods. *Journal of Artificial Intelligence Research*, 38(1):135–187.

Guy Aston and Lou Burnard. 1998. The BNC Handbook. `http://www.natcorp.ox.ac.uk`.

Colin J. Bannard and Chris Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. In *Proceedings of ACL 2005*, pages 597–604.

Regina Barzilay and Lillian Lee. 2003. Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment. In *Proceedings of HLT 2003*, pages 16–23.

Regina Barzilay and Kathleen McKeown. 2001. Extracting Paraphrases From a Parallel Corpus. In *Proceedings of ACL 2001*, pages 50–57.

Igor A. Bolshakov and Alexander F. Gelbukh. 2004. Synonymous Paraphrasing Using WordNet and Internet. In *Proceedings of NLDB 2004*, pages 312–323.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram Version 1. Linguistic Data Consortium LDC2006T13.

Steven Burrows, Martin Potthast, and Benno Stein. 2013. Paraphrase Acquisition via Crowdsourcing and Machine Learning. *ACM Transactions on Intelligent Systems and Technology*, 4(3):43:1–43:21.

Chris Callison-Burch. 2008. Syntactic Constraints on Paraphrases Extracted from Parallel Corpora. In *Proceedings of EMNLP 2008*, pages 196–205.

Jonathan Chevelu, Thomas Lavergne, Yves Lepage, and Thierry Moudenc. 2009. Introduction of a New Paraphrase Generation Tool Based on Monte-Carlo Sampling. In *Proceedings of ACL 2009*, pages 249–252.

Jonathan Chevelu, Ghislain Putois, and Yves Lepage. 2010. The True Score of Statistical Paraphrase Generation. In *Proceedings of COLING 2010 (Posters)*, pages 144–152.

Paul Clough, Robert Gaizauskas, Scott S. L. Piao, and Yorick Wilks. 2002. METER: MEasuring TExt Reuse. In *Proceedings of ACL 2002*, pages 152–159.

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing 2005*, pages 1–8.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Juri Ganitkevitch, Chris Callison-Burch, Courtney Napoles, and Benjamin Van Durme. 2011. Learning Sentential Paraphrases From Bilingual Parallel Corpora for Text-to-Text Generation. In *Proceedings of EMNLP 2011*, pages 1168–1179.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of HLT 2013*, pages 758–764.

David Kauchak and Regina Barzilay. 2006. Paraphrasing for Automatic Evaluation. In *Proceedings of HLT 2006*.

Bryan Klimt and Yiming Yang. 2004. The Enron Corpus: A New Dataset for Email Classification Research. In *Proceedings of ECML 2004*, pages 217–226.

Donald E. Knuth. 1986. *The TEXbook*. Addison-Wesley.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A New Benchmark Collection for Text Categorization Research. *The Journal of Machine Learning Research*, 5:361–397.

G. Harry McLaughlin. 1969. SMOG Grading: A New Readability Formula. *Journal of Reading*, 12(8):639–646.

Donald Metzler and Eduard Hovy. 2011. Mavuno: A Scalable and Effective Hadoop-Based Paraphrase Acquisition System. In *Proceedings of the Third Workshop on Large Scale Data Mining 2011*, pages 3:1–3:8.

Donald Metzler, Eduard H. Hovy, and Chunliang Zhang. 2011. An Empirical Evaluation of Data-Driven Paraphrase Generation Techniques. In *Proceedings of ACL 2011 (Short Papers)*, pages 546–551.

Bonnie J. F. Meyer. 2003. Text Coherence and Readability. *Topics in Language Disorders*, 23(3):204–224.

Bo Pang, Kevin Knight, and Daniel Marcu. 2003. Syntax-Based Alignment of Multiple Translations: Extracting Paraphrases and Generating New Sentences. In *Proceedings of HLT 2003*, pages 102–109.

Marius Pasca and Péter Dienes. 2005. Aligning Needles in a Haystack: Paraphrase Acquisition Across the Web. In *Proceedings of IJCNLP 2005*, pages 119–130.

Judea Pearl. 1984. *Heuristics*. Addison-Wesley.

Chris Quirk, Chris Brockett, and William B. Dolan. 2004. Monolingual Machine Translation for Paraphrase Generation. In *Proceedings of EMNLP 2004*, pages 142–149.

Patrick Riehmann, Henning Gruendl, Martin Potthast, Martin Trenkmann, Benno Stein, and Bernd Froehlich. 2012. WORDGRAPH: Keyword-in-Context Visualization for NETSPEAK's Wildcard Search. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1411–1423.

Helene Schmolz, David Coquil, and Mario Döller. 2012. In-Depth Analysis of Anaphora Resolution Requirements. In *Proceedings of TIR 2012*, pages 174–179.

Edgar A. Smith and R. J. Senter. 1967. Automated Readability Index. Technical Report AMRL-TR-6620, Wright-Patterson Air Force Base.

Benno Stein and Daniel Curatolo. 2006. Phonetic Spelling and Heuristic Search. In *Proceedings of ECAI 2006*, pages 829–830.

Benno Stein and Sven Meyer zu Eißen. 2008. Retrieval Models for Genre Classification. *Scandinavian Journal of Information Systems (SJIS)*, 20(1):91–117.

Benno Stein, Martin Potthast, and Martin Trenkmann. 2010. Retrieving Customary Web Language to Assist Writers. In *Proceedings of ECIR 2010*, pages 631–635.

Nathan Sturtevant, Ariel Felner, Maxim Likhachev, and Wheeler Ruml. 2012. Heuristic Search Comes of Age. In *Proceedings of AAAI 2012*.

Shiqi Zhao, Cheng Niu, Ming Zhou, Ting Liu, and Sheng Li. 2008. Combining Multiple Resources to Improve SMT-Based Paraphrasing Model. In *Proceedings of ACL 2008*, pages 1021–1029.

Shiqi Zhao, Xiang Lan, Ting Liu, and Sheng Li. 2009. Application-Driven Statistical Paraphrase Generation. In *Proceedings of ACL 2009*, pages 834–842.

Does a Computational Linguist have to be a Linguist?

Martin Kay
Stanford University and Universität des Saarlandes
Stanford, CA, USA and Saarbrücken, Germany
mjkay@stanford.edu and kay@coli.uni-saarland.de

Invited Speaker Abstract

Early computational linguists supplied much of theoretical basis that the ALPAC report said was needed for research on the practical problem of machine translation. The result of their efforts turned out to be more fundamental in that it provided a general theoretical basis for the study of language use as a process, giving rise eventually to constraint-based grammatical formalisms for syntax, finite-state approaches to morphology and phonology, and a host of models how speakers might assemble sentences, and hearers take them apart. Recently, an entirely new enterprise, based on machine learning and big data, has sprung on the scene and challenged the ALPAC committee's finding that linguistic processing must have a firm basis in linguistic theory. In this talk, I will show that the long-term development of linguistic processing requires linguistic theory, sophisticated statistical manipulation of big data, and a third component which is not linguistic at all.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, page 2030, Dublin, Ireland, August 23-29 2014.

Query Lattice for Translation Retrieval

Meiping Dong[†], Yong Cheng[‡], Yang Liu[†], Jia Xu[‡], Maosong Sun[†], Tatsuya Izuha[◇], Jie Hao[#]

[†]State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Sci. and Tech., Tsinghua University, Beijing, China
hellodmp@163.com, {liuyang2011,sms}@tsinghua.edu.cn

[‡]Institute for Interdisciplinary Information Sciences
Tsinghua University, Beijing, China
chengyong3001@gmail.com, xu@tsinghua.edu.cn

[◇]Toshiba Corporation Corporate Research & Development Center
tatsuya.izuha@toshiba.co.jp

[#]Toshiba (China) R&D Center
haojie@toshiba.com.cn

Abstract

Translation retrieval aims to find the most likely translation among a set of target-language strings for a given source-language string. Previous studies consider the single-best translation as a query for information retrieval, which may result in translation error propagation. To alleviate this problem, we propose to use the *query lattice*, which is a compact representation of exponentially many queries containing translation alternatives. We verified the effectiveness of query lattice through experiments, where our method explores a much larger search space (from 1 query to 1.24×10^{62} queries), runs much faster (from 0.75 to 0.13 second per sentence), and retrieves more accurately (from 83.76% to 93.16% in precision) than the standard method based on the query single-best. In addition, we show that query lattice significantly outperforms the method of (Munteanu and Marcu, 2005) on the task of parallel sentence mining from comparable corpora.

1 Introduction

Translation retrieval aims to search for the most probable translation candidate from a set of target-language strings for a given source-language string. Early translation retrieval methods were widely used in example-based and memory-based translation systems (Sato and Nagao, 1990; Nirenburg et al., 1993; Baldwin and Tanaka, 2000; Baldwin, 2001). Often, the document set is a list of translation records that are pairs of source-language and target-language strings. Given an input source string, the retrieval system returns a translation record of maximum similarity to the input on the source side. Although these methods prove to be effective in example-based and memory-based translation systems, they heavily rely on parallel corpora that are limited both in size and domain.

More recently, Liu et al. (2012) have proposed a new translation retrieval architecture that depends only on monolingual corpora. Given an input source string, their system retrieves translation candidates from a set of target-language sentences. This can be done by combining machine translation (MT) and information retrieval (IR): machine translation is used to transform the input source string to a coarse translation, which serves as a query to retrieve the most probable translation in the monolingual corpus. Therefore, it is possible for translation retrieval to have access to a huge volume of monolingual corpora that are readily available on the Web.

However, the MT + IR pipeline suffers from the *translation error propagation problem*. Liu et al. (2012) use 1-best translations, which are inevitably erroneous due to the ambiguity and structural divergence of natural languages, as queries to the IR module. As a result, translation mistakes will be

Corresponding author: Jia Xu. Tel: +86-10-62781693 Ext 1683. Homepage: iiis.tsinghua.edu.cn/∼xu

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2031–2041, Dublin, Ireland, August 23-29 2014.

propagated to the retrieval process. This situation aggravates when high-accuracy MT systems are not available for resource-scarce languages.

In this work, we propose to use *query lattice* in translation retrieval to alleviate the translation error propagation problem. A query lattice is a compact representation of exponentially many queries. We design a retrieval algorithm that takes the query lattice as input to search for the most probable translation candidate from a set of target-language sentences. As compared with Liu et al. (2012), our approach explores a much larger search space (from 1 query to 1.24×10^{62} queries), runs much faster (from 0.75 second per sentence to 0.13), and retrieves more accurately (from 83.76% to 93.16%). We also evaluate our approach on extracting parallel sentences from comparable corpora. Experiments show that our translation retrieval system significantly outperforms a state-of-the-art parallel corpus mining system.

2 Related Work

Our work is inspired by three research topics: retrieving translation candidates from parallel corpus, using lattice to compactly represent exponentially many alternatives, and using lattice as query in information retrieval.

1. *Translation Retrieval using Parallel Corpus*. The idea of retrieving translation candidates from existing texts originated in example-based and memory-based translation (Sato and Nagao, 1990; Nirenburg et al., 1993; Baldwin and Tanaka, 2000; Baldwin, 2001). As these early efforts use a parallel corpus (e.g., translation records that are pairs of source-language and target-language strings), they focus on calculating the similarity between two source-language strings. In contrast, we evaluate the translational equivalence of a given source string and a target string in a large monolingual corpus.

2. *Lattice in Machine Translation*. Lattices have been widely used in machine translation: considering Chinese word segmentation alternatives (Xu et al., 2005), speech recognition candidates (Matsoukas et al., 2007), SCFG (Dyer et al., 2008) and so on in the decoding process, minimum bayes risk decoding (Tromble et al., 2008), minimum error rate training (Macherey et al., 2008), system combination (Feng et al., 2009), just to name a few. In this work, we are interested in how to use a lattice that encodes exponentially many translation candidates as a single query to retrieve similar target sentences via an information retrieval system.

3. *Query Lattice in Information Retrieval*. The use of lattices in information retrieval dates back to Moore (1958). Most current lattice-based IR systems often treat lattices as conceptional hierarchies or thesauri in formal concept analysis (Priss, 2000; Cheung and Vogel, 2005). In spoken document retrieval, however, lattices are used as a compact representation of multiple speech recognition transcripts to estimate the expected counts of words in each document (Saraclar and Sproat, 2004; Zhou et al., 2006; Chia et al., 2010). Our work is significantly different from previous work that uses the bag-of-words model because translation retrieval must take structure and dependencies in text into account to ensure translational equivalence.

3 Query Lattice for Translation Retrieval

3.1 Translation Retrieval

Let $\mathbf{f}$ be a source-language string, $\mathbf{E}$ be a set of target-language strings, the problem is how to find the most probable translation $\hat{\mathbf{e}}$ from $\mathbf{E}$. Note that $\mathbf{E}$ is a monolingual corpus rather than a parallel corpus. Therefore, string matching on the source side (Sato and Nagao, 1990; Nirenburg et al., 1993; Baldwin and Tanaka, 2000; Baldwin, 2001) does not apply here.

We use $P(\mathbf{e}|\mathbf{f})$ to denote the probability that a target-language sentence $\mathbf{e}$ is the translation of a source-language sentence $\mathbf{f}$. As suggested by Liu et al. (2012), it can be decomposed into two sub-models by

introducing a coarse translation $\mathbf{q}$ as a hidden variable:

$$P(\mathbf{e}|\mathbf{f}) = \sum_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} P(\mathbf{q},\mathbf{e}|\mathbf{f}) \quad (1)$$

$$= \sum_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} P(\mathbf{q}|\mathbf{f}) \times P(\mathbf{e}|\mathbf{q},\mathbf{f}) \quad (2)$$

where $P(\mathbf{q}|\mathbf{f})$ is a **translation** sub-model, $P(\mathbf{e}|\mathbf{q},\mathbf{f})$ is a **retrieval** sub-model, and $\mathbf{Q}(\mathbf{f})$ is the set of all possible translations of the sentence $\mathbf{f}$. Note that $\mathbf{q}$ actually serves as a **query** to the retrieval sub-model.

To take advantage of various translation and retrieval information sources, we use a log-linear model (Och and Ney, 2002) to define the conditional probability of a query $\mathbf{q}$ and a target sentence $\mathbf{e}$ conditioned on a source sentence $\mathbf{f}$ parameterized by a real-valued vector $\boldsymbol{\theta}$:

$$P(\mathbf{q},\mathbf{e}|\mathbf{f};\boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f}))}{\sum_{\mathbf{q}'\in\mathbf{Q}(\mathbf{f})}\sum_{\mathbf{e}'\in\mathbf{E}}\exp(\boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q}',\mathbf{e}',\mathbf{f}))} \quad (3)$$

where $\mathbf{h}(\cdot)$ is a vector of feature functions and $\boldsymbol{\theta}$ is the corresponding feature weight vector.

Accordingly, the decision rule for the latent variable model is given by

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \sum_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} \exp(\boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f})) \right\} \quad (4)$$

As there are exponentially many queries, it is efficient to approximate the summation over all possible queries by using maximization instead:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \max_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} \left\{ \boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f}) \right\} \right\} \quad (5)$$

Unfortunately, the search space is still prohibitively large since we need to enumerate all possible queries. Liu et al. (2012) split Eq. (5) into two steps. In the first step, a translation module runs to produce the 1-best translation $\hat{\mathbf{q}}$ of the input string $\mathbf{f}$ as a query:

$$\hat{\mathbf{q}} \approx \arg\max_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} \left\{ \boldsymbol{\theta}_t\cdot\mathbf{h}_t(\mathbf{q},\mathbf{e},\mathbf{f}) \right\} \quad (6)$$

where $\mathbf{h}_t(\cdot)$ is a vector of translation features and $\boldsymbol{\theta}_t$ is the corresponding feature weight vector. In the second step, a monolingual retrieval module takes the 1-best translation $\hat{\mathbf{q}}$ as a query to search for the target string $\hat{\mathbf{e}}$ with the highest score:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \boldsymbol{\theta}_r\cdot\mathbf{h}_r(\hat{\mathbf{q}},\mathbf{e},\mathbf{f}) \right\} \quad (7)$$

where $\mathbf{h}_r(\cdot)$ is a vector of retrieval features and $\boldsymbol{\theta}_r$ is the corresponding feature weight vector.

Due to the ambiguity of translation, however, state-of-the-art MT systems are still far from producing high-quality translations, especially for distantly-related languages. As a result, the 1-best translations are usually erroneous and potentially introduce retrieval mistakes.

A natural solution is to use n-best lists as queries:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \max_{\mathbf{q}\in\mathbf{N}(\mathbf{f})} \left\{ \boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f}) \right\} \right\} \quad (8)$$

where $\mathbf{N}(\mathbf{f}) \subset \mathbf{T}(\mathbf{f})$ is the n-best translations of the input source sentence $\mathbf{f}$.

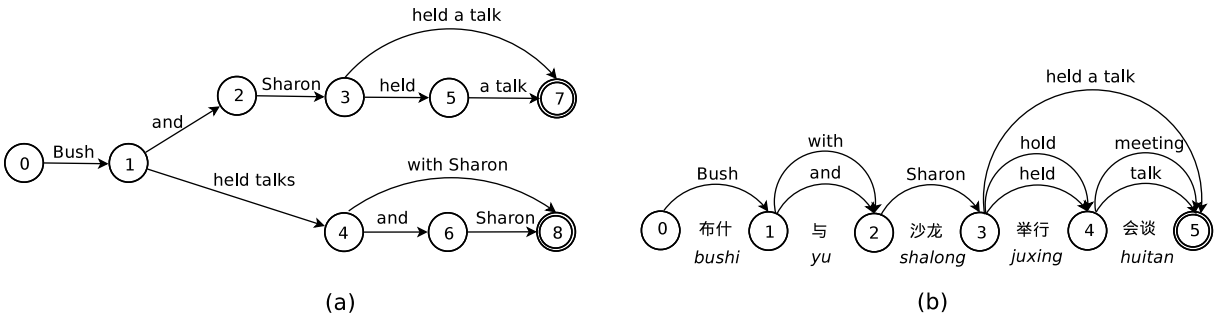


Figure 1: Two kinds of query lattices: (a) search graph that is generated *after* phrase-based decoding and (b) translation option graph that is generated *before* decoding. Translation option graph is more compact and encodes more translation candidates.

Although using n-best lists apparently improves the retrieval accuracy over using 1-best lists, there are two disadvantages. First, the decision rule in Eq. (8) requires to enumerate all the n translations and retrieve for n times. In other words, the time complexity increases linearly. Second, an n-best list only accounts for a tiny fraction of the exponential search space of translation. To make things worse, there are usually very few variations in n-best translations because of spurious ambiguity - a situation where multiple derivations give similar or even identical translations.

Therefore, we need to find a more elegant way to enable the retrieval module to explore exponentially many queries without sacrificing efficiency.

3.2 Query Lattice

We propose to use **query lattice** to compactly represent exponentially many queries. For example, given a source sentence "*bushi yu shalong juxing huitan*", we can use the **search graph** produced by a phrase-based translation system (Koehn et al., 2007) as a lattice to encode exponentially many derivations.

Figure 1(a) shows a search graph for the example source sentence. Each edge is labeled with an English phrase as well as the corresponding translation feature value vector. Node 0 denotes the starting node. Node 7 and node 8 are two ending nodes. Each path from the starting node to an ending node denotes a query. Paths that reach the same node in the lattice correspond to recombined hypotheses that have equivalent feature histories (e.g., coverage, last generated target words, the end of last covered source phrase, etc) in phrase-based decoding.

However, there are two problems with using search graph as query lattice. First, it is computationally expensive to run a phrase-based system to generate search graphs. The time complexity for phrase-based decoding with beam search is $O(n^2b)$ (Koehn et al., 2007), where n is the length of source string and b is the beam width. Moreover, the memory requirement is usually very high due to language models. As a result, translation is often two orders of magnitude slower than retrieval. Second, a search graph has too many "duplicate" edges due to different reordering, which increase the time complexity of retrieval (see Section 3.3). For example, in Figure 1(a), the English phrase "Sharon" occurs two times due to different reordering.

Alternatively, we propose to use **translation option graph** as query lattice. In a phrase-based translation system, translation options that are phrase pairs matching a substring in the input source string are collected *before* decoding. These translation options form a query lattice with monotonic reordering. Figure 1(b) shows an example translation option graph, in which nodes are sorted according to the positions of source words. Each edge is labeled with an English phrase as well as the corresponding translation feature value vector.

We believe that translation option graph has three advantages over search graph:

1. *Improved efficiency in translation*. Translation option graph requires no decoding.

2. *Improved efficiency in retrieval*. Translation option graph has no duplicate edges.

Algorithm 1 Retrieval with lattice as query.

procedure LATTICERETRIEVE($\mathbf{L}(\mathbf{f}), \mathbf{E}, k$)
  $Q \leftarrow$ GETWORDS($\mathbf{L}(\mathbf{f})$) ▷ Get distinct words in the lattice to form a coarse query
  $\mathbf{E}_k \leftarrow$ RETRIEVE($\mathbf{E}, Q, k$) ▷ Retrieve top-k target sentences using the coarse query
  for all $\mathbf{e} \in \mathbf{E}_k$ **do**
    FINDPATH($\mathbf{L}(\mathbf{f}), \mathbf{e}$) ▷ Find a path with the highest score
  end for
  SORT($\mathbf{E}_k$) ▷ Sort retrieved sentences according the scores
  return $\mathbf{E}_k$
end procedure

Algorithm 2 Find a path with the highest score.

procedure FINDPATH($\mathbf{L}(\mathbf{f}), \mathbf{e}$)
  for $v \in \mathbf{L}(\mathbf{f})$ in topological order **do**
    $path(v) \leftarrow \emptyset$ ▷ Initialize the Viterbi path at node v
    $score(v) \leftarrow 0$ ▷ Initialize the Viterbi score at node v
    for $u \in IN(v)$ **do** ▷ Enumerate all antecedents
      $p \leftarrow path(u) \cup \{e_{u \rightarrow v}\}$ ▷ Generate a new path
      $s \leftarrow score(u) +$ COMPUTESCORE($e_{u \rightarrow v}$) ▷ Compute the path score
      if $s > score(v)$ **then**
        $path(v) \leftarrow p$ ▷ Update the Viterbi path
        $score(v) \leftarrow s$ ▷ Update the Viterbi score
      end if
    end for
  end for
end procedure

3. *Enlarged search space*. Translation option graph represents the entire search space of monotonic decoding while search graph prunes many translation candidates.

In Figure 1, the search graph has 9 nodes, 10 edges, 4 paths, and 3 distinct translations. In contrast, the translation option graph has 6 nodes, 9 edges, 10 paths, and 10 distinct translations. Therefore, translation option graph is more compact and encodes more translation candidates.

Although translation option graph ignores language model and lexcialized reordering models, which prove to be critical information sources in machine translation, we find that it achieves comparable or even better retrieval accuracy than search graph (Section 4). This confirms the finding of Liu et al. (2012) that language model and lexicalized reordering models only have modest effects on translation retrieval.

3.3 Retrieval with Query Lattice

Given a target corpus $\mathbf{E}$ and a query lattice $\mathbf{L}(\mathbf{f}) \subset \mathbf{Q}(\mathbf{f})$, our goal is to find the target sentence $\hat{\mathbf{e}}$ with the highest score $\boldsymbol{\theta} \cdot \mathbf{h}(\mathbf{q}, \mathbf{e}, \mathbf{f})$:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e} \in \mathbf{E}} \left\{ \max_{\mathbf{q} \in \mathbf{L}(\mathbf{f})} \left\{ \boldsymbol{\theta} \cdot \mathbf{h}(\mathbf{q}, \mathbf{e}, \mathbf{f}) \right\} \right\} \quad (9)$$

Due to the exponentially large search space, we use a coarse-to-fine algorithm to search for the target sentence with the highest score, as shown in Algorithm 1. We use an example to illustrate the basic idea. Given an input source sentence "*bushi yu shalong juxing le huitan*", our system first generates a query lattice like Figure 1(a). It is non-trivial to directly feed the query lattice to a retrieval system. Instead, we would like to first collect all distinct words in the lattice: {"*Bush*", "*and*" , "*Sharon*", "*held*", "*a*", "*talk*", "*talks*", "*with*"}. This set serves as a coarse single query and the retrieval system returns a list of target sentences that contain these words:

			Chinese	English
Training			1.21M	1.21M
Dev	in-domain	query	5K	
		document		2.23M
	out-of-domain	query	5K	
		document		2.23M
Test	in-domain	query	5K	
		document		2.23M
	out-of-domain	query	5K	
		document		2.23M

Table 1: The datasets for the retrieval evaluation. The training set is used to train the phrase-based translation model and language model for Moses (Koehn et al., 2007). The development set is used to optimize feature weights using the minimum-error-rate algorithm (Och, 2003). A development set consists of a query set and a document set. The test set is used to evaluate the retrieval accuracy. To examine the effect of domains on retrieval performance, we used two development and test sets: in-domain and out-domain.

President Bush gave a talk at a meeting
Bush held a meeting with Sharon
Sharon and Bush attended a meeting held at London

Note that as a retrieval system usually ignores the structural dependencies in text, the retrieved sentences (scored by retrieval features) are relevant but not necessarily translations of the input. Therefore, we can match each retrieved sentence against the query lattice to find a path with the highest score using additional translation features. For example, the Viterbi path for "*Bush held a meeting with Sharon*" in Figure 1(a) is "*Bush held talks with Sharon*". The translation features of matched arcs in the path are collected to compute the overall score according to Eq. (9). Finally, the algorithm returns a sorted list:

Bush held a meeting with Sharon
President Bush gave a talk at a meeting
Sharon and Bush attended a meeting held at London

More formally, the input of Algorithm 1 are a query lattice $\mathbf{L}(\mathbf{f})$, a target corpus $\mathbf{E}$, and a parameter k (line 1). The function GETWORDS simply collects all the distinct words appearing in the lattice (line 2), which are used for constructing a coarse boolean query Q. Then, the function RETRIEVE runs to retrieve the top-k target sentences $\mathbf{E}_k$ in the target corpus $\mathbf{E}$ only using standard IR features according to the query Q (line 3). These first two steps eliminate most unlikely candidates and return a coarse set of target sentence candidates efficiently. [1] Then, a procedure FINDPATH$(\mathbf{L}(\mathbf{f}), \mathbf{e})$ runs to search for the translation with the highest score for each candidate (lines 4-6). Finally, the algorithm returns the sorted list of target sentences (lines 7-9).

Algorithm 2 shows the procedure FINDPATH$(\mathbf{L}(\mathbf{f}), \mathbf{e})$, which searches for the path with higher score using a Viterbi-style algorithm. The function COMPUTESCORE scores an edge according to the Eq. (9) which linearly combines the translation and retrieval features.

Generally, the lattice-based retrieval algorithm has a time complexity of $O(k|E|)$, where $|E|$ is the number of edges in the lattice.

4 Experiments

In this section, we try to answer two questions:

1. Does using query lattices improve translation retrieval accuracy over using n-best lists?

2. How does translation retrieval benefit other end-to-end NLP tasks such as machine translation?

[1] In our experiments, we set the parameter k to 500 as a larger value of k does not give significant improvements but introduce more noises.

Accordingly, we evaluated our system in two tasks: translation retrieval (Section 4.1) and parallel corpus mining (Section 4.2).

4.1 Evaluation on Translation Retrieval

4.1.1 Experimental Setup

In this section, we evaluate the accuracy of translation retrieval: given a query set (i.e., source sentences), our system returns a sorted list of target sentences. The evaluation metrics include precision@n and recall.

The datasets for the retrieval evaluation are summarized in Table 1. The **training set**, which is used to train the phrase-based translation model and language model for the-state-of-the-art phrase-based system Moses (Koehn et al., 2007), contains 1.21M Chinese-English sentences with 32.0M Chinese words and 35.2M English words. We used the SRILM toolkit (Stolcke, 2002) to train a 4-gram language model on the English side of the training corpus. The **development set**, which is used to optimize feature weights using the minimum-error-rate algorithm (Och, 2003), consists of **query set** and a **document set**. We sampled 5K parallel sentences randomly, in which 5K Chinese sentences are used as queries and half of their parallelled English sentences(2.5K) mixed with other English sentences(2.3M) as the retrieval document set. As a result, we can compute precision and recall in a noisy setting. The **test set** is used to compute retrieval evaluation metrics. To examine the effect of domains on retrieval performance, we used two data sets: **in-domain** and **out-domain**. The in-domain development and test sets are close to the training set while the out-domain data sets are not.

We compare three variants of translation retrieval: 1-best list, n-best list, and lattice. For query lattice, we further distinguish between search graph and translation option graph. They are generated by Moses with the default setting.

We use both translation and retrieval features in the experiments. The translation features include phrase translation probabilities, phrase penalty, distance-based and lexicalized reordering models, language models, and word penalty. Besides the conventional IR features such as term frequency and inverse document frequency, we use five additional featured derived from BLEU (Papineni et al., 2002): the n-gram matching precisions between query and retrieved target sentence ($n = 1, 2, 3, 4$) and brevity penalty. These features impose structural constraints on retrieval and ensure translation closeness of retrieved target sentences. The minimum-error-rate algorithm supports a variety of loss functions. The loss function we used in our experiment is $1-$P@n. Note that using translation option graph as query lattice does not include language models and distance-based lexicalized reordering models as features.

4.1.2 Evaluation Results

Table 2 shows the results on the in-domain test set. The "# candidates" column gives the number of translation candidates explored by the retrieval module for each source sentence on average. The lattices, either generated by search graph or by translation options, contain exponentially many candidates. We find that using lattices dramatically improves the precisions over using 1-best and n-best lists. All the improvements over 1-best and n-best lists are significant statistically. The 1-best, n-best, and the search graph lattice share with the same translation time: 5,640 seconds for translating 5,000 queries. Note that the translation time is zero for the translation option graph because it does not need phrase-based decoding. For retrieval, the time cost for the n-best list method generally increases linearly. As the search graph lattice contains many edges, the retrieval time increases by an order of magnitude as compared with 100-best list. An interesting finding is that using translation options as a lattice contains more candidates and consumes much less time for retrieval than using search graph as a lattice. One possible reason is that a search graph generated by Moses usually contains many redundant edges. For example, Figure 1 is actually a search graph and many phrases occurs multiple times in the lattice (e.g., "and" and "Sharon"). In contrast, a lattice built by translation options hardly has any redundant edges but still represents exponentially many possible translations. We can also see that the lattice constructed by search graph considering language model can benefit the precision much, especially when n is little. But this advantage decreases with n increasing and the time consumed by translation options as lattice is much less than the search graph as lattice. Besides, the margin between them is not too large so we can

method	# candidates	P@n					time	
		n=1	n=5	n=10	n=20	n=100	translation	retrieval
1-best	1	87.40	91.40	92.24	92.88	93.64	5,640	82
10-best	10	89.84	93.20	93.96	94.36	95.56	5,640	757
100-best	100	90.76	94.32	95.00	95.76	96.76	5,640	7,421
lattice (graph)	1.20×10^{54}	93.60	96.08	96.28	96.52	96.80	5,640	89,795
lattice (options)	4.14×10^{62}	93.28	95.84	95.96	96.16	96.84	0	307

Table 2: Results on the in-domain test set. We use the minimum-error-rate training algorithm (Och, 2003) to optimize the feature with the respect to $1-$P@n.

method	# candidates	P@n					time	
		n=1	n=5	n=10	n=20	n=100	translation	retrieval
1-best	1	67.32	76.60	79.40	81.80	83.76	3,660	92
10-best	10	72.68	80.96	83.36	85.84	88.76	3,660	863
100-best	100	78.60	85.76	87.76	89.64	92.16	3,660	8,418
lattice (graph)	1.51×10^{61}	84.32	89.40	90.68	91.56	92.44	3,660	67,205
lattice (options)	1.24×10^{65}	81.92	88.00	89.80	91.24	93.16	0	645

Table 3: Results on the out-of-domain test set.

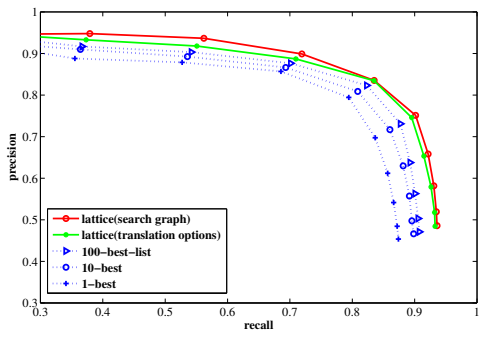


Figure 2: In-domain Precision-Recall curves.

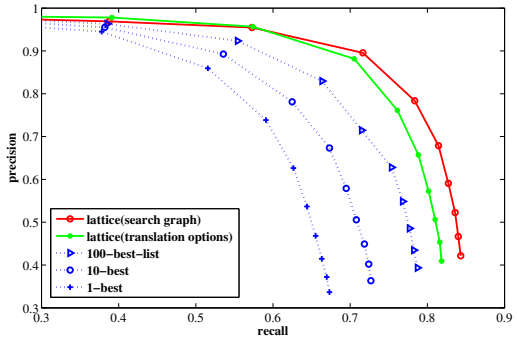


Figure 3: Out-domain Precision-Recall curves.

abandon some little precision for obtain the large time reducing. Therefore, using translation options as lattices seems to be both effective and efficient.

Table 3 shows the results on the out-of-domain test set. While the precisions for all methods drop, the margins between lattice-based retrieval and n-best list retrieval increase, suggesting that lattice-based methods are more robust when dealing with noisy datasets.

Figures 2 and 3 show the Precision-Recall curves on the in-domain and out-of-domain test sets. As the query set is derived from parallel sentences, recall can be computed in our experiments. The curves show that using lattices clearly outperforms using 1-best and n-best lists. The margins are larger on the out-of-domain test set.

4.2 Evaluation on Parallel Corpus Mining

In this section, we evaluate translation retrieval on the parallel corpus mining task: extracting a parallel corpus from a comparable corpus.

4.2.1 Experimental Setup

The comparable corpus for extracting parallel sentences contains news articles published by Xinhua News Agency from 1995 to 2010. Table 4 shows the detailed statistics. There are 1.2M Chinese and 1.7M English articles.

We re-implemented the method as described in (Munteanu and Marcu, 2005) as the baseline system.

language	articles	sentences	words	vocabulary
Chinese	1.2M	18.5M	441.2M	2.1M
English	1.7M	17.8M	440.2M	3.4M

Table 4: The Xinhua News Comparable Corpus from 1995 to 2010

Munteanu and Marcu (2005)			*this work*		
English words	Chinese words	BLEU	English words	Chinese Words	BLEU
5.00M	4.12M	22.84	5.00M	3.98M	25.44
10.00M	8.20M	25.10	10.00M	8.17M	26.62
15.00M	12.26M	25.41	15.00M	12.49M	26.49
20.00M	16.30M	25.56	20.00M	16.90M	26.87

Table 5: Comparison of BLEU scores using parallel corpora extracted by the baseline and our system. Given a comparable corpus (see Table 4), both systems extract parallel corpora that are used for training phrase-base models (Koehn et al., 2007). The baseline system is a re-implementation of the method described in (Munteanu and Marcu, 2005). Our system uses translation option graph as query lattice. Our system significantly outperforms the baseline for various sizes.

It assigned a score to each sentence pair using a classifier. Our system used translation option graph as query lattices due to its simplicity and effectiveness. For each source sentence in the comparable corpus, our system retrieved the top target sentence together with a score.

To evaluate the quality of extracted parallel corpus, we trained phrase-based models on it and ran Moses on NIST datasets. The development set is the NIST 2005 test set and the test set is the NIST 2006 test set. The final evaluation metric is case-insensitive BLEU-4.

4.2.2 Evaluation Results

Table 5 shows the comparison of BLEU scores using parallel corpora extracted by the baseline and our system. We find that our system significantly outperforms the baseline for various parallel corpus sizes. This finding suggests that using lattice to compactly represent exponentially many alternatives does help to alleviate the translation error propagation problem and identify parallel sentences of high translational equivalence.

5 Conclusion

In this work, we propose to use query lattice to address the translation error propagation problem in translation retrieval. Two kinds of query lattices are used in our experiments: search graph and translation option graph. We show that translation option graph is more compact and represents a much larger search space. Our experiments on Chinese-English datasets show that using query lattices significantly outperforms using n-best lists in the retrieval task. Moreover, we show that translation retrieval is capable of extracting high-quality parallel corpora from a comparable corpus. In the future, we plan to apply our approach to retrieving translation candidates directly from the Web, which can be seen as a huge monolingual corpus.

Acknowledgments

This research is supported by the 973 Program (No. 2014CB340501), the National Natural Science Foundation of China (No. 61331013 and No. 61033001), the 863 Program (No. 2012AA011102), Toshiba Corporation Corporate Research & Development Center, and the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme.

References

T. Baldwin and H. Tanaka. 2000. The effects of word order and segmentation on translation retrieval performance. In *Proceedings of COLING*.

Timothy Baldwin. 2001. Low-cost, high-performance translation retrieval: Dumber is better. In *Proceedings of ACL*, pages 18–25, Toulouse, France, July. Association for Computational Linguistics.

Karen Cheung and Douglas Vogel. 2005. Complexity reduction in lattice-based information retrieval. *Information Retrieval*, pages 285–299.

Tee Kiah Chia, Khe Chai Sim, Haizhou Li, and Hwee Tou Ng. 2010. Statistical lattice-based spoken document retrieval. *ACM Transactions on Information Systems*, 28(1).

Christopher Dyer, Smaranda Muresan, and Philip Resnik. 2008. Generalizing word lattice translation. In *Proceedings of ACL-HLT*, pages 1012–1020, Columbus, Ohio, June. Association for Computational Linguistics.

Yang Feng, Yang Liu, Haitao Mi, Qun Liu, and Yajuan Lü. 2009. Lattice-based system combination for statistical machine translation. In *Proceedings of EMNLP*, pages 1105–1113, Singapore, August. Association for Computational Linguistics.

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

Chunyang Liu, Qi Liu, Yang Liu, and Maosong Sun. 2012. THUTR: A translation retrieval system. In *Proceedings of COLING - Demo and Poster Sessions*, pages 321–328, Mumbai, India, December. The COLING 2012 Organizing Committee.

Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *Proceedings of EMNLP*, pages 725–734, Honolulu, Hawaii, October. Association for Computational Linguistics.

Spyros Matsoukas, Ivan Bulyko, Bing Xiang, Kham Nguyen, Richard Schwartz, and John Makhoul. 2007. Integrating speech recognition and machine translation. In *Proceedings of ICASSP*, volume 4, pages IV–1281. IEEE.

C.N. Moore. 1958. A mathematical theory of the use of language symbols in retrieval. In *ICSI 1958*.

Dragos Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–1504.

S. Nirenburg, C. Domashnev, and D.J. Grannes. 1993. Two approaches to matching in example-based machine translation. In *TMI 1993*.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL*, pages 295–302, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167, Sapporo, Japan, July. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Uta Priss. 2000. Lattice-based information retrieval. *Knwoledge Organization*, 27(3):132–142.

Murat Saraclar and Richard Sproat. 2004. Lattice-based search for spoken utterance retrieval. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL*, pages 129–136, Boston, Massachusetts, USA, May. Association for Computational Linguistics.

S. Sato and M. Nagao. 1990. Toward memory-based translation. In *Proceedings of COLING*.

Andreas Stolcke. 2002. Srilm: an extensible language modeling toolkit. In *Proceedings of ICSLP*.

Roy Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of EMNLP*, pages 620–629, Honolulu, Hawaii, October. Association for Computational Linguistics.

Jia Xu, Evgeny Matusov, Richard Zens, and Hermann Ney. 2005. Integrated chinese word segmentation in statistical machine translation. In *Proceedings of IWSLT 2005*, pages 141–147, Pittsburgh, PA, October.

Zheng-Yu Zhou, Peng Yu, Ciprian Chelba, and Frank Seide. 2006. Towards spoken-document retrieval for the internet: Lattice indexing for large-scale web-search architectures. In *Proceedings of HLT-NAACL*, pages 415–422, New York City, USA, June. Association for Computational Linguistics.

RED: A Reference Dependency Based MT Evaluation Metric

Hui Yu[†§] **Xiaofeng Wu**[‡] **Jun Xie**[†] **Wenbin Jiang**[†] **Qun Liu**[‡†] **Shouxun Lin**[†]
[†]Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences
[§]University of Chinese Academy of Sciences
{yuhui,xiejun,jiangwenbin,sxlin}@ict.ac.cn
[‡]CNGL, School of Computing, Dublin City University
{xiaofengwu,qliu}@computing.dcu.ie

Abstract

Most of the widely-used automatic evaluation metrics consider only the local fragments of the references and translations, and they ignore the evaluation on the syntax level. Current syntax-based evaluation metrics try to introduce syntax information but suffer from the poor parsing results of the noisy machine translations. To alleviate this problem, we propose a novel dependency-based evaluation metric which only employs the dependency information of the references. We use two kinds of reference dependency structures: headword chain to capture the long distance dependency information, and fixed and floating structures to capture the local continuous ngram. Experiment results show that our metric achieves higher correlations with human judgments than BLEU, TER and HWCM on WMT 2012 and WMT 2013. By introducing extra linguistic resources and tuning parameters, the new metric gets the state-of-the-art performance which is better than METEOR and SEMPOS on system level, and is comparable with METEOR on sentence level on WMT 2012 and WMT 2013.

1 Introduction

Automatic machine translation (MT) evaluation plays an important role in the evolution of MT. It not only evaluates the performance of MT systems, but also makes the development of MT systems rapider (Och, 2003). According to the type of the employed information, the automatic MT evaluation metrics can be classified into three categories: lexicon-based metrics, syntax-based metrics and semantic-based metrics.

The lexicon-based metrics, such as BLEU (Papineni et al., 2002), TER (Snover et al., 2006), METEOR (Lavie and Agarwal, 2007) and AMBER (Chen and Kuhn, 2011; Chen et al., 2012), are good at capturing the lexicon or phrase level information, e.g. fixed phrases or idioms. But they cannot adequately reflect the syntax similarity. Current efforts in syntax-based metrics, such as the headword chain based metric (HWCM) (Liu and Gildea, 2005), the LFG dependency tree based metric (Owczarzak et al., 2007) and syntactic/semantic-role overlap (Giménez and Màrquez, 2007) , suffer from the parsing of the potentially noisy machine translations, so the improvement of their performance is restricted due to the serious parsing errors. Semantic-based metrics, such as MEANT (Lo et al., 2012; Lo and Wu, 2013), have the similar problem that the accuracy of semantic role labeling (SRL) can also drop due to the errors in translations. To avoid the parsing of potentially noisy translations, the CCG based metric (Mehay and Brew, 2007) only uses the parsing result of reference and employs 2-gram dependents, but it did not achieve the state-of-the-art performance.

In this paper, we propose a novel dependency tree based MT evaluation metric. The new metric only employs the reference dependency tree, leaving the translation unparsed to avoid the error propagation. We use two kinds of reference dependency structures in our metric. One is the headword chain (Liu and Gildea, 2005) which can capture long distance dependency information. The other is fixed and floating structure (Shen et al., 2010) which can capture local continuous ngram. When calculating the matching score between the headword chain and the translation, we use a distance-based similarity. Experiment

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2042–2051, Dublin, Ireland, August 23-29 2014.

results show that our metric achieves higher correlations with human judgments than BLEU, TER and HWCM on WMT 2012 and WMT 2013. After introducing extra resources and tuning parameters on WMT 2010, the new metric is better than METEOR and SEMPOS on system level and comparable with METEOR on sentence level on WMT 2012 and WMT2013.

The remainder of this paper is organized as follows. Section 2 describes our new reference dependency based MT evaluation metric. In Section 3, we introduce some extra resources to this new metric. Section 4 presents the parameter tuning for the new metric. Section 5 gives the experiment results. Conclusions and future work are discussed in Section 6.

2 RED: A Reference Dependency Based MT Evaluation Metric

The new metric is a REference Dependency based automatic evaluation metric, so we name it RED. We present the new metric detailedly in this section. The description of dependency ngrams is given in Section 2.1. The method to score the dependency ngram is presented in Section 2.2. At last, the method of calculating the final score is introduced in Section 2.3.

2.1 Two Kinds of Dependency Ngrams

To capture both the long distance dependency information and the local continuous ngrams, we use both the headword chain and the fixed-floating structures in our new metric, which correspond to the two kinds of dependency ngram (dep-ngram), headword chain ngram and fixed-floating ngram.

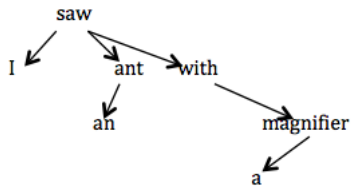


Figure 1: An example of dependency tree.

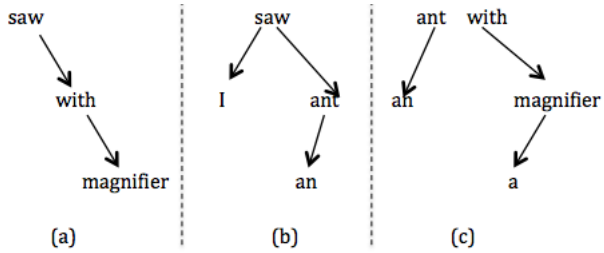


Figure 2: Different kinds of structures extracted from the dependency tree in Figure 1. (a): Headword chain. (b): Fixed structure. (c): Floating structure.

2.1.1 Headword chain

Headword chain is a sequence of words which corresponds to a path in the dependency tree (Liu and Gildea, 2005). For example, Figure 2(a) is a 3-word headword chain extracted from the dependency tree in Figure 1. Headword chain can represent the long distance dependency information, but cannot capture most of the continuous ngrams. In our metric, headword chain corresponds to the headword chain ngram in which the positions of the words are considered. So the form of headword chain ngram is expressed as $(w1_{pos1}, w2_{pos2}, ..., wn_{posn})$, where n is the length of the headword chain ngram. For example, the headword chain in Figure 2(a) is expressed as $(saw_2, with_5, magnifier_7)$.

2.1.2 Fixed and floating structures

Fixed and floating structures are defined in Shen et al. (2010). Fixed structures consist of a sub-root with children, each of which must be a complete constituent. They are called fixed dependency structures because the head is known or fixed. For example, Figure 2(b) shows a fixed structure. Floating structures consist of a number of consecutive sibling nodes of a common head, but the head itself is unspecified. Each of the siblings must be a complete constituent. Figure 2(c) shows a floating structure. Fixed-floating structures correspond to fixed-floating ngrams in our metric. Fixed-floating ngrams don't need the position information, and can be simply expressed as $(w1, w2, ..., wn)$, where n is the length of the

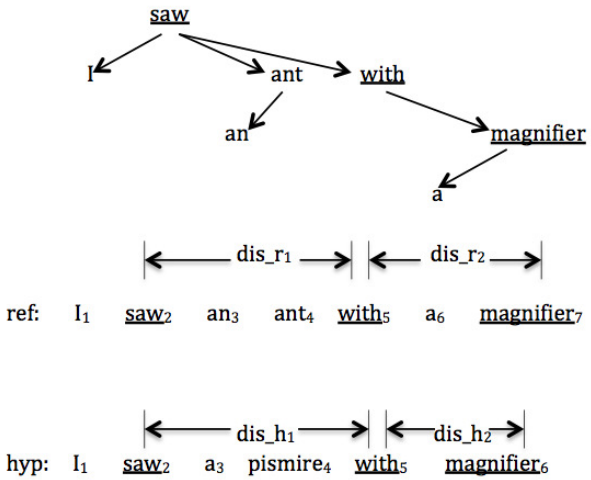


Figure 3: An example of calculating matching score for a headword chain ngram $(saw_2, with_5, magnifier_7)$. dis_r_1 and dis_r_2 are the distances between the corresponding two words in the reference. dis_h_1 and dis_h_2 are the distances between the corresponding two words in the hypothesis.

fixed-floating ngram. For example, the fixed structure in Figure 2(b) and the floating structre in Figure 2(c) can be expressed as (I, saw, an, ant) and $(an, ant, with, a, magnifier)$ respectively.

2.2 Scoring Dep-ngrams

Headword chain ngrams may not be continuous, while fixed-floating ngrams must be continuous. So the scoring methods of the two kinds of dep-ngrams are different, and we introduce the two scoring methods in Section 2.2.1 and Section 2.2.2 respectively.

2.2.1 Scoring headword chain ngram

For a headword chain ngram $(w1_{pos1}, w2_{pos2}, ..., wn_{posn})$, if we can find all these n words in the string of the translation with the same order as they appear in the reference sentence, we consider it a match and the matching score is a distance-based similarity which is calculated by the relative distance, otherwise it is not a match and the score is 0. The matching score is a decimal value between 0 and 1, which is more suitable than just use integer 0 and 1. For example, if the distance between two words in reference is 1, but the distance in two different hypotheses are 2 and 5 respectively. It's more reasonable to score them 0.5 and 0.2 rather than 1 and 0.

The relative distance dis_r_i between every two adjacent words in this kind of dep-ngram is calculated by Formula (1), where pos_{wi} is the position of word wi in the sentence. In Formula (1), we have $1 \leq i \leq n-1$ and n is the length of the dep-ngram. Then a vector $(dis\_r_1, dis\_r_2, ..., dis\_r_{n-1})$ is obtained. In the same way, we obtain vector $(dis\_h_1, dis\_h_2, ..., dis\_h_{n-1})$ for the translation side.

$$dis\_r_i = |pos_{w(i+1)} - pos_{wi}| \tag{1}$$

The matching score $p_{(d,hyp)}$ for a headword chain ngram (*d*) and the translation (*hyp*) is calculated according to Formula (2), where $n > 1$. When the length of the dep-ngram equals 1, the matching score equals 1 if the translation has the same word, otherwise, the matching score equals 0.

$$p_{(d,hyp)} = \begin{cases} exp(-\frac{\sum_{i=1}^{n-1} |dis\_r_i - dis\_h_i|}{n-1}) & if\ match \\ 0 & if\ unmatch \end{cases} \tag{2}$$

An example illustrating the calculation of the matching score $p_{(d,hyp)}$ is shown in Figure 3. There is a 3-word headword chain ngram $(saw_2, with_5, magnifier_7)$ in the dependency tree of the reference.

For this dep-3gram, the words are represented with underline in the reference dependency tree and the reference sentence in Figure 3. We can also find all the same three underlined words in the translation with the same order as they appear in the reference. Therefore, there is a match for this dep-3gram. To compute the matching score between this dep-3gram and the translation, we have:

- Calculate the distance

$$dis\_r_1 = |pos_{with} - pos_{saw}| = |5-2| = 3 \qquad dis\_r_2 = |pos_{magnifier} - pos_{with}| = |7-5| = 2$$
$$dis\_h_1 = |pos_{with} - pos_{saw}| = |5-2| = 3 \qquad dis\_h_2 = |pos_{magnifier} - pos_{with}| = |6-5| = 1$$

- Get the matching score as Formula (3) according to Formula (2). d denotes $(saw_2, with_5, magnifier_7)$ and hyp denotes the translation in the example.

$$p_{(d,hyp)} = exp(-\frac{|dis\_r_1 - dis\_h_1| + |dis\_r_2 - dis\_h_2|}{3-1}) = exp(-\frac{|3-3| + |2-1|}{3-1}) = exp(-0.5) \tag{3}$$

We also tried other methods to calculate the matching score, such as the cosine distance and the absolute distance, but the relative distance performed best. For a headword chain ngram with more than one matches in the translation, we choose the one with the highest matching score.

2.2.2 Scoring fixed-floating ngram

The words in the fixed-floating ngram are continuous, so we restrict the matched string in the translation also to being continuous. That means, for a fixed-floating ngram $(w1, w2, ..., wn)$, if we can find all these n words continuous in the translation with the same order as they appear in the reference, we think the dep-ngram can match with the translation. The matching score can be obtained by Formula (4), where d stands for a fixed-floating ngram and hyp stands for the translation.

$$p_{(d,hyp)} = \begin{cases} 1 & if\ match \\ 0 & if\ unmatch \end{cases} \tag{4}$$

2.3 Scoring RED

In the new metric, we use Fscore to obtain the final score. Fscore is calculated by Formula (5), where α is a value between 0 and 1.

$$Fscore = \frac{precision \cdot recall}{\alpha \cdot precision + (1-\alpha) \cdot recall} \tag{5}$$

The dep-ngrams of the reference and the string of the translation are used to calculate the precision and recall. In order to calculate precision, the number of the dep-ngrams in the translation should be given, but there is no dependency tree for the translation in our method. We know that the number of dep-ngrams has an approximate linear relationship with the length of the sentence, so we use the length of the translation to replace the number of the dep-ngrams in the translation dependency tree. Recall can be calculated directly since we know the number of the dep-ngrams in the reference. The precision and recall are computed as follows.

$$precision = \frac{\sum_{d \in D_n} p_{(d,hyp)}}{len_h}, \quad recall = \frac{\sum_{d \in D_n} p_{(d,hyp)}}{count_{n(ref)}}$$

D_n is the set of dep-ngrams with the length of n. len_h is the length of the translation. $count_{n(ref)}$ is the number of the dep-ngrams with the length of n in the reference.

The final score of RED is achieved using Formula (6), in which a weighted sum of the dep-ngrams' Fscore is calculated. w_{ngram} ($0 \leq w_{ngram} \leq 1$) is the weight of dep-ngram with the length of *n*. Fscore$_n$ is the Fscore for the dep-ngrams with the length of *n*.

$$RED = \sum_{n=1}^{N}(w_{ngram} \times Fscore_n) \tag{6}$$

3 Introducing Extra Resources

Many automatic evaluation metrics can only find the exact match between the reference and the translation, and the information provided by the limited number of references is not sufficient. Some evaluation metrics, such as TERp (Snover et al., 2009) and METOER, introduce extra resources to expand the reference information. We also introduce some extra resources to RED, such as stem, synonym and paraphrase. The words within a sentence can be classified into content words and function words. The effects of the two kinds of words are different and they shouldn't have the same matching score, so we introduce a parameter to distinguish them. The methods of applying these resources are introduced as follows.

- Stem and Synonym

  Stem(Porter, 2001) and synonym (WordNet[1]) are introduced to RED in the following three steps. First, we obtain the alignment with Meteor Aligner (Denkowski and Lavie, 2011) in which not only exact match but also stem and synonym are considered. We use stem and synonym together with exact match as three match modules. Second, the alignment is used to match for a dep-ngram. We think the dep-ngram can match with the translation if the following conditions are satisfied. 1) Each of the words in the dep-ngram has a matched word in the translation according to the alignment; 2) The words in dep-ngram and the matched words in translation appear in the same order; 3) The matched words in translation must be continuous if the dep-ngram is a fixed-floating ngram. At last, the match module score of a dep-ngram is calculated according to Formula (7). Different match modules have different effects, so we give them different weights.

  $$s_{mod} = \frac{\sum_{i=1}^{n} w_{m_i}}{n}, \ \ 0 \leq w_{m_i} \leq 1 \tag{7}$$

  m_i is the match module (exact, stem or synonym) of the ith word in a dep-ngram. w_{m_i} is the match module weight of the ith word in a dep-ngram. n is the number of words in a dep-ngram.

- Paraphrase

  When introducing paraphrase, we don't consider the dependency tree of the reference, because paraphrases may not be contained in the headword chain and fixed-floating structures. First, the alignment is obtained with METEOR Aligner, only considering paraphrase. Second, the matched paraphrases are extracted from the alignment and defined as paraphrase-ngram. The score of a paraphrase is $1 \times w_{par}$, where w_{par} is the weight of paraphrase-ngram.

- Function word

  We introduce a parameter w_{fun} ($0 \leq w_{fun} \leq 1$) to distinguish function words and content words. w_{fun} is the weight of function words. The function word score of a dep-ngram or paraphrase-ngram is computed according to Formula (8).

  $$s_{fun} = \frac{C_{fun} \times w_{fun} + C_{con} \times (1 - w_{fun})}{C_{fun} + C_{con}} \tag{8}$$

  C_{fun} is the number of function words in the dep-ngram or paraphrase-ngram. C_{con} is the number of content words in the dep-ngram or paraphrase-ngram.

[1] http://wordnet.princeton.edu/

We use RED-plus (REDp) to represent RED with extra resources, and the final score are calculated as Formula (9), in which $Fscore_p$ is obtained using $precison_p$ and $recall_p$ as Formula (10).

$$REDp = \sum_{n=1}^{N} (w_{ngram} \times Fscore_{p_n}) \tag{9}$$

$$Fscore_p = \frac{precision_p \cdot recall_p}{\alpha \cdot precision_p + (1-\alpha) \cdot recall_p} \tag{10}$$

$precision_p$ and $recall_P$ in Formula (10) are calculated as follows.

$$precision_p = \frac{score_{par_n} + score_{dep_n}}{len_h}\ , \quad recall_p = \frac{score_{par_n} + score_{dep_n}}{count_n(ref) + count_n(par)}$$

len_h is the length of the translation. $count_{n(ref)}$ is the number of the dep-ngrams with the length of n in the reference. $count_n(par)$ is the number of paraphrases with length of n in reference. $score_{par_n}$ is the match score of paraphrase-ngrams with the length of *n*. $score_{dep_n}$ is the match score of dep-ngrams with the length of *n*. $score_{par_n}$ and $score_{dep_n}$ are calculated as follows.

$$score_{par_n} = \sum_{par \in P_n} (1 \times w_{par} \times s_{fum})\ , \qquad score_{dep_n} = \sum_{d \in D_n} (p_{(d,hyp)} \times s_{mod} \times s_{fun})$$

P_n is the set of paraphrase-ngrams with the length of *n*. D_n is the set of dep-ngrams with the length of *n*.

4 Parameter Tuning

There are several parameters in REDp, and different parameter values can make the performance of REDp different. For example,w_{ngram} represents the weight of dep-ngram with the length of *n*. The effect of ngrams with different lengths are different, and they shouldn't have the same weight. So we can tune the parameters to find their best values.

We try a preliminary optimization method to tune parameters in REDp. A heuristic search is employed and the parameters are classified into two subsets. The parameter optimization is a grid search over the two subsets of parameters. When searching Subset 1, the parameters in Subset 2 are fixed, and then Subset 1 and Subset 2 are exchanged to finish this iteration. Several iterations are executed to finish the parameter tuning process. This heuristic search may not find the global optimum but it can save a lot of time compared with exhaustive search. The optimization goal is to maximize the sum of Spearman's ρ rank correlation coefficient on system level and Kendall's τ correlation coefficient on sentence level. ρ is calculated using the following equation.

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2-1)}$$

where d_i is the difference between the human rank and metric's rank for system i. n is the number of systems. τ is calculated as follows.

$$\tau = \frac{\text{number of concordant pairs} - \text{number of discordant pairs}}{\text{number of concordant pairs} + \text{number of discordant pairs}}$$

The data of into-English tasks in WMT 2010 are used to tune parameters. The tuned parameters are listed in Table 1.

5 Experiments

5.1 Data

The test sets in experiments are WMT 2012 and WMT 2013. The language pairs are German-to-English (de-en), Czech-to-English (cz-en), French-to-English (fr-en), Spanish-to-English (es-en) and Russian-to-English (ru-en). The number of translation systems for each language pair are showed in Table 2. For each language pair, there are 3003 sentences in WMT 2012 and 3000 sentences in WMT 2013.

Parameter	α	w_{fun}	w_{exact}	w_{stem}	w_{syn}	w_{par}	w_{1gram}	w_{2gram}	w_{3gram}
tuned values	0.9	0.2	0.9	0.6	0.6	0.6	0.6	0.5	0.1

Table 1: Parameter values after tuning on WMT 2010. α is from Formula (10). w_{fun} is the weight of function word. w_{exact}, w_{stem} and w_{syn} are the weights of the three match modules 'exact stem synonym' respectively. w_{par} is the weight of paraphrase-ngram. w_{1gram}, w_{2gram} and w_{3gram} are the weights of dep-ngram with the length of 1, 2 and 3 respectively.

Language pairs	cz-en	de-en	es-en	fr-en	ru-en
WMT2012	6	16	12	15	-
WMT2013	12	23	17	19	23

Table 2: The number of translation systems for each language pair on WMT 2012 and WMT 2013.

We parsed the reference into constituent tree by Berkeley parser[2] and then converted the constituent tree into dependency tree by Penn2Malt[3]. Presumably, the performance of the new metric will be better if the dependency trees are labeled by human. Reference dependency trees are labeled only once and can be used forever so it will not increase costs.

5.2 Baselines

In the experiments, we compare the performance of our metric with the widely-used lexicon-based metrics such as BLEU[4], TER[5] and METEOR[6], dependency-based metric HWCM and semantic-based metric SEMPOS (Macháček and Bojar, 2011) which has the best performance on system level according to the published results of WMT 2012.

The results of BLEU are obtained using 4-gram with smoothing option. The version of TER is 0.7.25. The results of METEOR are obtained by Version 1.4 with task option '*rank*'. We re-implement HWCM which employs an epsilon value of 10^{-3} to replace zero for smoothing purpose. The correlations of SEMPOS are obtained from the published results of WMT 2012 and WMT 2013.

5.3 Experiment Results

The experiments on both system level and sentence level are carried out. On system level, the correlations are calculated using Spearman's rank correlation coefficient ρ (Pirie, 1988). Kendall's rank correlation coefficient τ (Kendall, 1938) is employed to evaluate the sentence level correlation. Our method performs best when the maximum length of dep-ngram is set to 3, so we only present the results with the maximum length of 3. RED represents the new metric with exact match and the parameter values are set as follows. $\alpha = 0.5$. $w_{1gram} = w_{2gram} = w_{3gram} = 1/3$. REDp represents the new metric with extra resources and tuned parameter values which are listed in Table (1).

5.3.1 System level correlations

The system level correlations are shown in Table 3. RED is better than BLEU, TER and HWCM on average on both WMT 2012 and WMT 2013, which reflects that using syntactic information and only parsing the reference side are helpful. REDp gets the best result on all of the language pairs except cz-en on WMT 2012. The significant improvement from RED to REDp illustrates the effect of extra resources and the parameter tuning. Stem, synonym and paraphrase can enrich the reference and provide extra knowledge for automatic evaluation metric. There are several parameters in REDp, and different parameter values can make the performance of REDp different. So the performance can be optimized through parameter tuning. SEMPOS got the best correlation according to the published results of WMT

[2] http://code.google.com/p/berkeleyparser/downloads/list
[3] http://stp.lingfil.uu.se/~nivre/research/Penn2Malt.html
[4] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v13a.pl
[5] http://www.cs.umd.edu/~snover/tercom
[6] http://www.cs.cmu.edu/~alavie/METEOR/download/meteor-1.4.tgz

2012, and METEOR got the best correlation according to the published results of WMT 2013 on into-English task on system level. REDp gets better result than SEMPOS and METEOR on both WMT 2012 and WMT 2013, so REDp achieves the state-of-the-art performance on system level.

data	WMT 2012					WMT 2013					
Metrics	cz-en	de-en	es-en	fr-en	ave	cz-en	de-en	es-en	fr-en	ru-en	ave
BLEU	.886	.671	.874	.811	.811	.936	.895	.888	.989	.670	.876
TER	.886	.624	.916	.821	.812	.800	.833	.825	.951	.581	.798
HWCM	.943	.762	.937	.818	.865	.902	.904	.886	.951	.756	.880
METEOR	.657	.885	.951	**.843**	.834	.964	.961	.979	.984	.789	.935
SEMPOS	.943	.924	.937	.804	.902	.955	.919	.930	.938	.823	.913
RED	**1.0**	.759	.951	.818	.882	.964	.951	.930	.989	.725	.912
REDp	.943	**.947**	**.965**	**.843**	**.925**	**.982**	**.973**	**.986**	**.995**	**.800**	**.947**

Table 3: System level correlations on WMT 2012 and WMT 2013. The value in bold is the best result in each column. *ave* stands for the average result of the language pairs on WMT 2012 or WMT 2013.

5.3.2 Sentence level correlations

The sentence level correlations on WMT 2012 and WMT 2013 are shown in Table 4. RED is better than BLEU and HWCM on all the language pairs, which reflects the effectiveness of syntactic information and only parsing the reference. By introducing extra resources and parameter tuning, REDp achieves significant improvement over RED. Stem, synonym and paraphrase can enrich the reference and provide extra knowledge for automatic evaluation metric. There are several parameters in REDp, and different parameter values can make the performance of REDp different. A better performance can be exploited through parameter tuning. From the results of REDp and METEOR, we can see that REDp gets the comparable results with METEOR on sentence level on both WMT 2012 and WMT 2013.

data	WMT 2012					WMT 2013					
Metrics	cz-en	de-en	es-en	fr-en	ave	cz-en	de-en	es-en	fr-en	ru-en	ave
BLEU	.157	.191	.189	.210	.187	.199	.220	.259	.224	.162	.213
HWCM	.158	.207	.203	.204	.193	.187	.208	.247	.227	.175	.209
METEOR	**.212**	**.275**	**.249**	**.251**	**.247**	**.265**	**.293**	**.324**	**.264**	**.239**	**.277**
RED	.165	.218	.203	.221	.202	.210	.239	.292	.246	.196	.237
REDp	**.212**	.271	.234	.250	.242	.259	.290	.323	.260	.223	.271

Table 4: Sentence level correlations on WMT 2012 and WMT 2013. The value in bold is the best result in each column. *ave* stands for the average result of the language pairs on WMT 2012 or WMT 2013.

6 Conclusion and Future Work

In this paper, we propose a reference dependency based automatic MT evaluation metric RED. The new metric only uses the dependency trees of the reference, which avoids the parsing of the potentially noisy translations. Both long distance dependency information and the local continuous ngrams are captured by the new metric. The experiment results indicate that RED achieves better correlations than BLEU, TER and HWCM on both system level and sentence level. REDp, the improved version of RED through adding extra resources and preliminary parameter tuning, gets state-of-the-art results which are better than METEOR and SEMPOS on system level. On sentence level, REDp gets the comparable performance with METEOR.

In the future, we will use the dependency forest instead of the dependency tree to reduce the effect of parsing errors. We will also apply RED and REDp to the tuning process of SMT to improve the translation quality.

Acknowledgements

The authors were supported by National Natural Science Foundation of China (Contract 61202216) and National Natural Science Foundation of China (Contract 61379086). Qun Liu's work was partially supported by the Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the CNGL at Dublin City University. Sincere thanks to the three anonymous reviewers for their thorough reviewing and valuable suggestions.

References

Boxing Chen and Roland Kuhn. 2011. Amber: A modified bleu, enhanced ranking metric. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 71–77, Edinburgh, Scotland, July. Association for Computational Linguistics.

Boxing Chen, Roland Kuhn, and George Foster. 2012. Improving amber, an mt evaluation metric. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 59–63, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 85–91. Association for Computational Linguistics.

Jesús Giménez and Lluís Màrquez. 2007. Linguistic features for automatic evaluation of heterogenous mt systems. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 256–264. Association for Computational Linguistics.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: an automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 228–231, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ding Liu and Daniel Gildea. 2005. Syntactic features for evaluation of machine translation. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 25–32.

Chi-kiu Lo and Dekai Wu. 2013. MEANT at WMT 2013: A tunable, accurate yet inexpensive semantic frame based MT evaluation metric. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 422–428, Sofia, Bulgaria, August. Association for Computational Linguistics.

Chi-kiu Lo, Anand Karthik Tumuluru, and Dekai Wu. 2012. Fully automatic semantic mt evaluation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 243–252, Montréal, Canada, June. Association for Computational Linguistics.

Matouš Macháček and Ondřej Bojar. 2011. Approximating a deep-syntactic metric for mt evaluation and tuning. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 92–98. Association for Computational Linguistics.

Dennis Mehay and Chris Brew. 2007. BLEUTRE: Flattening Syntactic Dependencies for MT Evaluation. In *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*.

F.J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics.

Karolina Owczarzak, Josef van Genabith, and Andy Way. 2007. Dependency-based automatic evaluation for machine translation. In *Proceedings of the NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation*, SSST '07, pages 80–87, Stroudsburg, PA, USA. Association for Computational Linguistics.

K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

W Pirie. 1988. Spearman rank correlation coefficient. *Encyclopedia of statistical sciences*.

Martin F Porter. 2001. Snowball: A language for stemming algorithms.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2010. String-to-dependency statistical machine translation. *Computational Linguistics*, 36(4):649–671.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Matthew Snover, Nitin Madnani, Bonnie J Dorr, and Richard Schwartz. 2009. Fluency, adequacy, or hter?: exploring different human judgments with a tunable mt metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 259–268. Association for Computational Linguistics.

Quality Estimation of English-French Machine Translation: A Detailed Study of the Role of Syntax

Rasoul Kaljahi[†‡], Jennifer Foster[†], Raphael Rubino[†‡], Johann Roturier[‡]
[†]NCLT, School of Computing, Dublin City University, Ireland
{rkaljahi, jfoster, rrubino}@computing.dcu.ie
[‡]Symantec Research Labs, Dublin, Ireland
johann_roturier@symantec.com

Abstract

We investigate the usefulness of syntactic knowledge in estimating the quality of English-French translations. We find that dependency and constituency tree kernels perform well but the error rate can be further reduced when these are combined with hand-crafted syntactic features. Both types of syntactic features provide information which is complementary to tried-and-tested non-syntactic features. We then compare source and target syntax and find that the use of parse trees of machine translated sentences does not affect the performance of quality estimation nor does the intrinsic accuracy of the parser itself. However, the relatively flat structure of the French Treebank does appear to have an adverse effect, and this is significantly improved by simple transformations of the French trees. Finally, we provide further evidence of the usefulness of these transformations by applying them in a separate task – parser accuracy prediction.

1 Introduction

Quality Estimation (QE) for Machine Translation (MT) involves judging the correctness of the output of an MT system given an input and no reference translation (Blatz et al., 2003; Ueffing et al., 2003; Specia et al., 2009). An accurate QE-for-MT system would mean that reliable decisions could be made regarding whether to publish a machine translation as is or to re-direct it to a translator, either for post-editing or to be translated from scratch. The scores produced by a QE system can also be used to choose between translations, in a system combination framework or in n-best list reranking. The work presented here takes place in the context of a wider study, the aim of which is to develop an English-French QE system so that technical support material that is produced on a daily basis by a company's English-speaking customers can be translated automatically into French and made available with confidence to the company's French-speaking customer base.

It is reasonable to assume that syntactic features are useful in QE for MT as a way of capturing the syntactic complexity of the source sentence, the grammaticality of the target translation and the syntactic symmetry between the source sentence and its translation. This assumption has been borne out by previous research which has demonstrated the usefulness of syntactic features for English-Spanish QE (Hardmeier et al., 2012; Rubino et al., 2012). We focus more closely on understanding the role of syntax by comparing the use of hand-crafted features and tree kernels (Collins and Duffy, 2002; Moschitti, 2006), and by teasing apart the contribution of target and source syntax.

We find that both tree kernels and manually engineered features produce statistically significantly better results than a strong set of non-syntactic features provided as a baseline by the organisers of the 2012 WMT shared task on QE for MT (Callison-Burch et al., 2012), and that both types of syntactic features can be combined fruitfully with this baseline. Furthermore, we show that it is worthwhile to combine tree kernels with hand-crafted features. Our tree kernel features are the complete set of tree fragments of both the constituency and dependency trees of the source and target sentences. Our hand-crafted feature set consists of an initial set of 489 constituency and dependency features which are then reduced to a set of 144 with no significant loss in performance.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2052–2063, Dublin, Ireland, August 23-29 2014.

We then show that source (English) constituency trees significantly outperform target (French) translation constituency trees in this task. We hypothesise that this is happening because a) the French parser has a lower accuracy compared to the English, or b) the target trees sentences are harder to parse, representing, as they do, potentially ill-formed machine translations which may result in noisier parse trees which are harder to learn from. If the first hypothesis were true, we would expect to see a drop in the accuracy of our QE system when we use lower-accuracy parses. We do not observe this. If the second hypothesis were true, we would expect to observe that the target trees were also less useful than the source trees in the opposite translation direction (French-English). Instead, we find that the target (English) constituency trees significantly outperform the source (French) constituency trees, suggesting that the difference between source and target that we observe in the original English-French experiment is related neither to intrinsic parser accuracy nor to translation direction but rather to the languages/treebanks.

We explore the extent to which the difference between French and English constituency trees is due to the relatively flatter structure of the French treebank. We use simple transformation heuristics to introduce more nodes into the French trees and significantly improve the performance. We also apply these heuristics in a second task, parser accuracy prediction. This task is similar to QE for MT except we are predicting the quality of a parse tree in the absence of a reference parse tree. We also find here that the modified trees also outperform the original trees, suggesting that one must proceed with caution when using French Treebank tree fragments in a machine-learning task.

The paper's novel contributions are as follows:

1. Evidence that syntactic information is useful in English-French QE for MT and further evidence that it is useful in QE for MT in general
2. A comparison of two methods of representing syntactic information in QE
3. A more comprehensive set of syntactic features than has been previously been used in QE for MT
4. A comparison of the role of source and target syntax in English-French QE for MT
5. A set of heuristics that can be applied to French Treebank trees resulting in performance improvements in the tasks of both QE for MT and parser accuracy prediction

The rest of this paper is organised as follows: we discuss related work in using syntax in QE in Section 2, we describe the data in Section 3, and we then go on to describe the QE framework and the systems built in Section 4. We follow this with an investigation of the role of source and target syntax in Section 5 before presenting our heuristics to modify the French constituency trees in Section 6.

2 Related Work

Features extracted from parser output have been used before in training QE for MT systems. Quirk (2004) uses a single syntax-based feature which indicates whether a full parse for the source sentence could be found. Hardmeier et al. (2012) employ tree kernels to predict the 1-to-5 post-editing cost of a machine-translated sentence. They use tree kernels derived from syntactic constituency and dependency trees of the source side (English) and only dependency trees of the translation side (Spanish). The tree kernels are used both alone and combined with non-syntactic features. The combined setting ranked second in the 2012 shared task on QE for MT (Callison-Burch et al., 2012). Rubino et al. (2012) explore a variety of syntactic features extracted from the output of both a hand-crafted broad-coverage grammar/parser and a statistical constituency parser on the WMT 2012 data set. They find that the syntactic features make an important contribution to the overall system. In a framework for combining QE and automatic metrics to evaluate MT output, Specia and Giménez (2010) use part-of-speech (POS) tag language model probabilities of the MT output 3-grams as features for QE and features built upon syntactic chunks, dependencies and constituent structure to build automatic MT evaluation metrics. Avramidis (2012) builds a series of models for estimating post-editing effort using syntactic features such as parse probabilities and syntactic label frequency. In a similar vein, Gamon et al. (2005) use POS tag trigrams, CFG rules and features derived from a semantic analysis of the MT output to classify it as fluent or disfluent.

In this work, we compare the use of tree kernels and hand-crafted features extracted from the constituency and dependency trees of the source and target sides of a translation pair, as well as comparing the role of source and target syntax. In addition, we conduct a more in-depth analysis of these approaches and compare the utility of syntactic information extracted from the source side and target sides of the translation.

3 Data

While there is evidence to suggest that predicting human evaluation scores is superior to predicting automatic metrics in QE for ME (Quirk, 2004), it has also been shown that human judgements are not necessarily consistent (Snover et al., 2006). A more practical consideration is that human evaluation exists for just a few language pairs and domains. To the best of our knowledge, the only available English-to-French data set which contains human judgements of translation quality are as follows:

- CESTA (Hamon et al., 2007), which is selected from the Official Journal of the European Commission and also from the health domain. In addition to the domain (and style) difference to newswire (the domain on which our parsers are trained), a major stumbling block which prevents us from using this data set is its small size: only 1135 segments have been evaluated manually.
- WMT 2007 (Callison-Burch et al., 2007), which contains only 302 distinct source segments (each with approx. 5 translations) only half of which is in the news domain.
- FAUST[1], which is out-of-domain and difficult to apply to our setting as the evaluations and post-edits are user feedbacks, often in the form of phrases/fragments.

Thus, we instead attempt to predict automatic metric scores as there is a sufficient amount of parallel text for our language pair and domain. We use `BLEU`[2](Papineni et al., 2002), `TER`[3](Snover et al., 2006) and `METEOR`[4] (Denkowski and Lavie, 2011), which are the most-widely used MT evaluation metrics. All metrics are applied at the segment level.[5]

We randomly select 4500 parallel segments from the News development data sets released for the WMT13 translation task (Bojar et al., 2013). In order to be independent of any one translation system, we translate the data set with the following three systems and randomly choose 1500 distinct segments from each:

- ACCEPT[6]: a phrase-based Moses system trained on training sets of WMT12 releases of Europarl and News Commentary plus data from Translators Without Borders (TWB)
- SYSTRAN: a proprietary rule-based system
- Bing[7]: an online translation system

The data set is randomly split into 3000 training, 500 development and 1000 test segments. We use the development set for tuning model parameters and building hand-crafted feature sets, and the test set for testing model performance and analyses purposes.

4 Syntax-based QE

One way to employ syntactic information in a machine-learning task is to manually compile a set of features that can be extracted automatically from a parse tree. An example of one such feature is the label of the root of the tree. Another method is to directly use these trees in a *tree kernel* (Collins and Duffy, 2002; Moschitti, 2006). This approach allows exponentially-sized feature spaces (e.g. all subtrees

[1] `http://www.faust-fp7.eu/faust/Main/DataReleases`

[2] Version 13a of MTEval script was used at the segment level.

[3] TER COMpute 0.7.25: `http://www.cs.umd.edu/~snover/tercom/`

[4] METEOR 1.4: `http://www.cs.cmu.edu/~alavie/METEOR/`

[5] We present `1-TER` to be more easily comparable to `BLEU` and `METEOR`. There is no upper bound for `TER` scores unlike the other two metrics. Scores higher than 1 occur when the number of errors is higher than the segment length. To avoid this, scores higher than 1 are cut-off to 1 before being converted to `1-TER`.

[6] `http://www.accept.unige.ch/Products/D_4_1_Baseline_MT_systems.pdf`

[7] `http://www.bing.com/translator`

of a tree) to be efficiently modelled using dynamic programming and has shown to be effective in many natural language processing tasks including parsing and named entity recognition (Collins and Duffy, 2002), semantic role labelling (Moschitti, 2006), sentiment analysis (Wiegand and Klakow, 2010) and QE for MT (Hardmeier et al., 2012). Although there can be overlap between the information captured by the two approaches, each can capture information that the other one cannot. In addition, while tree kernels involve minimal feature engineering, hand-crafted features offer more flexibility. Moschitti (2006) shows that combining the two is beneficial. We use both hand-crafted features and tree kernels, applied separately and combined together.

For parsing the English and French data into their constituency structures, a PCFG-LA parser[8] is used. We train the English parser on the training section of the Wall Street Journal (WSJ) section of the *Penn Treebank* (PTB) (Marcus et al., 1993). The French parser is trained on the training section of the *French Treebank* (FTB) (Abeillé et al., 2003). We obtain dependency parses by converting the English constituency parses using the `Stanford` converter (de Marneffe and Manning, 2008) and the French parses using `Const2Dep` (Candito et al., 2010). We evaluate the performance of the QE models using Root Mean Square Error (`RMSE`) and Pearson correlation coefficient (`r`). To compute the statistical significance of the performance differences between QE models, we use paired bootstrap resampling following Koehn (2004). We randomly resample (with replacement) a set of N instances from the predictions of each of the two given systems, where N is the size of the test set. We repeat this sampling N times and count the number of times each of the two settings is better in terms of each measure (RMSE and `Pearson r`). If a setting is better more than 95% of the time, we consider it statistically significant at $p < 0.05$.

In the following sections, we first describe our baseline systems and then the quality estimation systems build using tree kernels, hand-crafted features and a combination of both.

4.1 Baseline QE Systems

In order to verify the usefulness of syntax-based QE, we build two baselines. The first baseline (`BM`) uses the mean of the segment-level evaluation scores in the training set for all instances. In the second baseline (`BW`), the 17 baseline features of the WMT12 QE Shared Task are used. `BW` is considered a strong baseline as the system that used only these features was ranked higher than many of the participating systems. We use support vector regression implemented in the `SVMLight` toolkit[9] to build `BW`. The Radial Basis Function (RBF) kernel is used. The results for both baselines are presented in the first two rows of Table 1. Since `BW` is a stronger baseline than `BM`, we will compare all syntax-based systems to `BW` only.

4.2 Syntax-based QE with Tree Kernels

Tree kernels are kernel functions that compute the similarity between two instances of data represented as trees based on the number of common fragments between them. Therefore, the need for explicitly encoding an instance in terms of manually-designed and extracted features is eliminated, while benefitting from a very high-dimensional feature space. Moschitti (2006) introduces an efficient implementation of tree kernels within a support vector machine framework. Instead of extracting all possible tree fragments, the algorithm compares only tree fragments rooted in two similar nodes. This algorithm is made available through `SVMLight-TK` software[10], which is used in this work.

In order to extract tree kernels from dependency trees, the labels on the arcs must be removed. Following Tu et al. (2012), the nodes in the resulting tree representation are word forms and dependency relations, omitting POS tag information. An example is shown in Figure 1. A word is a child of its dependency relation to its head. The dependency relation in turn is the child of the head word. This continues until the root of the tree.

Based on preliminary experiments on our development set, we use *subset* tree kernels, where the tree fragments are subtrees rooted at any node in the tree so that no production rule expanding a node in the

[8] `https://github.com/CNGLdlab/LORG-Release`. The Lorg parser is very similar to the Berkeley parser (Petrov et al., 2006), the main difference being its unknown word handling mechanism (Attia et al., 2010).

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

[10] `http://disi.unitn.it/moschitti/Tree-Kernel.htm`

	BLEU		1-TER		METEOR	
	RMSE	r	RMSE	r	RMSE	r
BM	0.1626	0	0.1965	0	0.1657	0
BW	**0.1601**	**0.1766**	**0.1949**	**0.1565**	**0.1625**	**0.2047**
TK	0.1581	0.2437	0.1888	0.2774	0.1595	0.2715
BW+TK	**0.1570**	**0.2696**	**0.1879**	**0.2939**	**0.1576**	**0.3111**
HC	0.1603	0.1998	0.1913	0.2365	0.1610	0.2516
BW+HC	**0.1587**	**0.2418**	**0.1899**	**0.2611**	**0.1585**	**0.2964**
SyQE	0.1577	0.2535	0.1887	0.2797	0.1594	0.2743
BW+SyQE	**0.1568**	**0.2802**	**0.1879**	**0.2937**	**0.1576**	**0.3127**

Table 1: QE performances measured by RMSE and Pearson r; BM: Mean baseline, BW: WMT 17 baseline features, TK: tree kernels, HC: hand-crafted features, SyQE: full syntax-based systems (TK+HC). Statistically significantly better scores compared to their counterpart (upper row in the row block) are in bold.

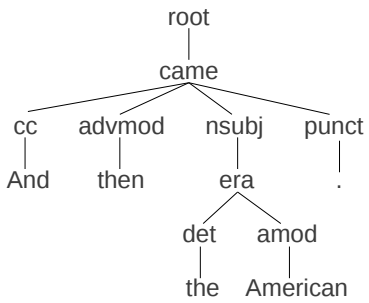


Figure 1: Tree Kernel Representation of Dependency Structure for *And then the American era came.*

subtree is split. Unlike *subtree* kernels, *subset* tree kernels allow tree fragments with non-terminals as leaves. We tune the C parameter for Pearson r on the development set, with all other parameters left as default.

We build a system with all four parse trees for every training instance, which includes the constituency and dependency trees of the source and target side of the translation. The third row of Table 1 shows the performance of this system which is named TK. The results achieved using this system represent a statistically significant improvement over the BW baseline results. In order to examine their complementarity, we combine these tree kernels and the baseline features (BW+TK) in the fourth row of Table 1. This combined system performs better than the two individual systems.

While BLEU prediction is the most accurate (lowest RMSE), METEOR prediction appears to be the easiest to learn (highest Pearson r). TER prediction seems to be more difficult than BLEU and METEOR prediction, especially in terms of prediction error. This is probably related to the distribution of each of these metric scores in our data set. The standard deviations (σ) of BLEU, TER and METEOR scores are 0.1620, 0.1943 and 0.1652 respectively. The substantially higher σ of TER scores makes them harder to predict accurately leading to higher prediction error.

4.3 Syntax-based QE with Hand-crafted Features

We design a set of constituency and dependency *feature types*, some of which have previously been used by the works described in Section 2 and some introduced here. Each feature type contains at least two features, one extracted from the source and the other from the translation. Numerical feature types can be further instantiated by extracting the ratio and differences between the source and target side feature values. Some feature types are parametric meaning that they can be varied by changing the value of a parameter. For example, the non-terminal label is a parameter for the non-terminal-label-count

	Constituency
∗1	Label of the root node of the constituency tree
2	Height of the constituency tree which is the number of edges from root node to the farthest terminal (leaf) node
∗3	Number of nodes in the constituency tree
4	Log probability of the constituency parse assigned by the parser
∗5	Parseval F_1 score of the tree with respect to a tree produced by the Stanford parser (Klein and Manning, 2003)
∗6	Right hand side of the CFG production rule expanding the root node
7	All non-lexical and lexical CFG production rules expanding the tree nodes
∗8	Average arity of the non-lexical CFG production rules expanding the constituency tree nodes
9	Counts of each non-terminal label in the tree
∗10	POS unigrams, 3-grams and 5-grams
11	POS n-gram scores against language models trained on the POS tags of the respective treebanks using the SRILM toolkit (`http://www.speech.sri.com/projects/srilm/`) with Witten-Bell smoothing
∗12	Counts of each 12 universal POS tags (Petrov et al., 2012)
∗13	Location of the first verb in the sentence in terms of the token distance from the beginning
∗14	Average number of POS n-grams in each n-gram frequency quartile of the POS corpora of the respective treebanks
	Dependency
∗1	POS tag of the top node (dependent of the dummy root node) of the dependency tree
∗2	Number of dependents of the top node
∗3	Sequence of all dependency relations which modify the top node
∗4	Sequence of the POS tags of the dependents of the top node
∗5	Average number of dependents per node
∗6	Height of the tree computed in the same way for the constituency tree
∗7	3- and 5-gram sequences of dependency relations of the tokens to their head
∗8	Number of most frequent dependency relations in our `News` training set
∗9	Dependency relation n-gram scores against language models trained on the respective treebanks for each language
∗10	Average number of dependency relation n-grams in each n-gram frequency quartile of the respective treebanks
∗11	Pairs of tokens and their dependency relations to their head

Table 2: Constituency and dependency feature types

feature type. Therefore, it instantiates as several features, one for each non-terminal-label.

As in `BW`, we use support vector machines (SVM) to build the QE systems using these hand-crafted features. We keep only those features which fire for more than a threshold which is set empirically on the development set. Table 2 lists our syntax-based feature types and their descriptions. Those that have, to the best of our knowledge, not been used in QE for MT before are marked with an asterisk.

The total number of feature-value pairs in the full feature set is 489. Since this feature set is large and contains many sparse features, we attempt to reduce it through ablation experiments in which we directly compare the effect of leaving out features that we suspect may be redundant. For example, we investigate whether either the ratio or difference of the source and target numerical features or both of them are redundant by building three systems, one without ratio features, one without difference features and one with neither. This process is also carried out for log probability and perplexity features, original and universal POS-tag-based features, n-gram and language model score features, lexical and non-lexical CFG rules, and n-gram orders (i.e. 3-gram vs. 5-gram features). This process proved useful: we found, for example, that either 3- or 5-grams worked better than both together and features based on universal POS tags better than those based on original POS tags.

The final reduced feature set contains 144 features-value pairs. We build one QE system with all 489 features `HC-all` and one with the reduced set of 144 features `HC` . Table 3 compares the performance on the development and test set. The system with the reduced feature set performs consistently better than the `HC-all` system on the development set, mostly with statistically significant differences. However, on the test set, the performance degrades albeit not statistically significantly. Considering a more than 70% reduction in feature set size, this relatively small degradation is tolerable. We use the reduced feature set as our hand-crafted feature set for the rest of the work.

Compared to `TK` in Table 1 (third and fourth versus fifth and sixth rows), the performances are lower for all MT metrics, though not statistically significantly. It is worth noting that we observed an opposite

	BLEU		1-TER		METEOR	
	RMSE	r	RMSE	r	RMSE	r
Development Set						
HC-all	0.1567	0.3026	0.1851	0.2746	0.1575	0.2996
HC	**0.1540**	0.3398	**0.1819**	**0.3263**	**0.1547**	0.3452
Test Set						
HC-all	0.1603	0.2108	0.1902	0.2510	0.1607	0.2493
HC	0.1603	0.1998	0.1913	0.2365	0.1610	0.2516

Table 3: QE performance with all hand-crafted syntactic features HC-all and the reduced feature set HC. Statistically significantly better scores compared to their counterpart (upper row) are in bold.

	RMSE	r
TK-CD-ST	0.1581	0.2437
TK-CD-S	0.1584	0.2294
TK-CD-T	0.1597	0.2101
TK-C-S	**0.1583**	**0.2312**
TK-C-T	0.1608	0.1479
TK-D-S	0.1598	0.1869
TK-D-T	0.1598	0.2102

Table 4: BLEU prediction performances with tree kernels of only source S or translation T side trees. The scores in bold are statistically better than their counterparts in the same row block. The original result with source and target combined is provided for reference in the first row.

behaviour on the development set, where hand-crafted features largely outperform tree kernels. This suggests that the tree kernels are more generalisable. We also combine these features with the WMT 17 baseline features (BW+HC). This combination also improves over both syntax-based and baseline systems, confirming again the usefulness of syntactic information in addition to surface features.

We combine tree kernels and hand-crafted features to build a full syntax-based QE system (SyQE), which improves over both TK and HC (Table 1) . The improvements for TER and METEOR prediction are slight but statistically significant for BLEU prediction. This system is also combined with BW in BW+SyQE (the last row of Table 1), resulting in statistically significant gains for all metrics.

5 Source and Target Syntax in Syntax-based QE

We now turn our attention to the parts played by source and target syntax in QE for MT. To save space, we present only the BLEU scores for the tree kernel systems. Table 4 shows the results achieved by systems built using either the source or target side of the translations.

At a glance, it can be seen that the source side constituency tree kernels outperform the target side ones, while the opposite is the case for dependency tree kernels. The differences for constituency trees are however substantially bigger. When both constituency and dependency trees are combined, the source side trees perform better (TK-CD-S vs. TK-CD-T).

The following three hypotheses could explain this difference between TK-C-S and TK-C-T:

1. **The Role of Parser Accuracy:** The fact that French parsing models do not reach the high Parseval F1s achieved by English parsing models could explain the difference in usefulness between the French and English consistuency trees. On the standard parsing test sets, the English parsing model achieves an F1 of 89.6 and the French an F_1 of 83.4.
2. **Parsing Machine Translation Output:** The difference between the source and target could be happening because the target side is machine translation output and (presumably) represents a lower

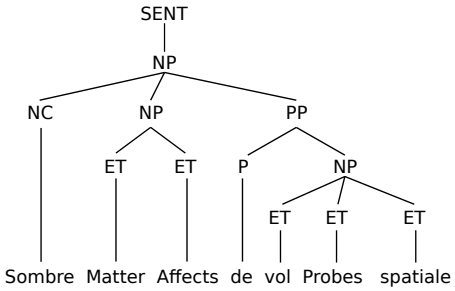


Figure 2: Parse tree of the machine translation of *Dark Matter Affects Flight of Space Probes* to French

quality set of sentences than the source (see Figure 2 for an example of a parse tree for a poor translation).

3. **Differences in Annotation Strategies:** The difference between the source and target could be due to the idiosyncrasies of the underlying treebanks which is not carried over via the conversion tools to the dependency structure.

Hypotheses 1 and 2 relate the usefulness of parse trees in QE to the intrinsic quality of the parse trees. French constituency trees are less accurate than English ones, either because the French parsing model is not as accurate as the English one (Hypothesis 1) or because the possibly ungrammatical nature of the French parsing input adversely affects the quality of the parse tree (Hypothesis 2). Although this low quality would be expected to affect the dependency trees in the same way since they are directly derived from the consistency trees, this is not the case and it appears that the problematic aspects of the French parses are abstracted away from the dependency trees.

To test the first hypothesis, we investigate the role of parser accuracy in QE. For both languages, we substitute the standard parsing models used in all our prior experiments with "lower-accuracy" models trained using only a fraction of the training data (following Quirk and Corston-Oliver (2006)). The English parsing model achieves an F_1 of 72.5 and the French an F_1 of 66.5, representing drops of approximately 17 points from the original models. The RMSE and Pearson r of the new QE model are 0.1583 and 0.2350 compared to 0.1581 and 0.2437 of the one trained with original trees (see also the third row of Table 1). These results show that the use of these lower-accuracy models has only a minimal and statistically insignificant effect on QE performance, suggesting that intrinsic parser accuracy is not the reason why the target constituency trees are less useful than the source constituency trees.[11]

To investigate the second hypothesis, we switch the translation direction to French-to-English. Therefore, we now parse the well-formed French input sentences and the machine-translated English segments. If the second hypothesis were true, the target side parse trees in this direction would still underperform the source side ones. The results are shown in Table 5. All the systems using target trees outperform those using source trees. The difference between source and target in the models that use constituency trees is especially substantial and statistically significant. Thus, it is apparent that the suspected lower quality of constituency parse trees of MT output is not the reason for the lower QE performance.

We now seek the answer in our third hypothesis, i.e. in the difference between the annotation schemes of the PTB and the FTB. One major difference, noted by, for example, Schluter and van Genabith (2007), is that the FTB has a relatively flatter structure. It lacks a verb phrase (`VP`) node and phrases modifying the verb are the sibling of the verb nucleus. We investigate this further in the next section.

6 Modifying French Parse Trees

In order to test whether the annotation strategy is a reason for the lower performance of French constituency tree kernels, we apply a set of three heuristics which introduce more structure to the French parse trees (1&2) or simply make them more PTB-like (3):

- *Heuristic 1* automatically adds a `VP` node above the verb node (`VN`) and at most 3 of its immediate adjacent nodes if they are noun or prepositional phrases (`NP` or `PP`).

[11] See (Kaljahi et al., 2013) for a more detailed exploration of the role of parser accuracy in QE for MT.

	RMSE	r
TK-FE/CD-ST	0.1561	0.2334
TK-FE/CD-S	0.1574	0.1830
TK-FE/CD-T	0.1559	**0.2423**
TK-FE/C-S	0.1581	0.1578
TK-FE/C-T	**0.1556**	**0.2336**
TK-FE/D-S	0.1577	0.1655
TK-FE/D-T	0.1579	0.1886

Table 5: BLEU prediction performances with tree kernels for Fr-En direction (FE) (C: constituency, D: dependency, S: source, T: translation)

	RMSE	r
TK-C-T	0.1608	0.1479
TK-C-T$_m$	**0.1591**	**0.2143**
TK-CD-ST	0.1581	0.2437
TK-CD-ST$_m$	**0.1574**	**0.2609**

Table 6: QE with tree kernels using original and modified French trees ($_m$)

- *Heuristic 2* stratifies some of the production rules in the tree by grouping together every two equal adjacent POS tags under a new node with a tag made of the POS tag suffixed with _St.
- *Heuristic 3* moves coordinated nodes (the immediate left sibling of the COORD node) under COORD.

Figure 3 shows examples of the application of each of these methods. We apply these heuristics to the parsed MT output in the English-French translation direction and rebuild the tree kernel system with translation side constituency trees (TK-C-T) and the full tree kernel system (TK-CD-ST) with the modified trees. The results are presented in Table 6. Despite the possibility of introducing linguistic errors, these heuristics yield a statistically significant improvement in QE performance. Unsurprisingly, the changes are bigger for the system with only translation side constituency trees as in the full system there are three other tree types involved. These results suggest that the structure of the French constituency trees is a factor in the lower performance of its tree kernels in QE.[12]

The gain achieved by applying these heuristics is related to the fact that there are more similar fragments extracted from the modified structure which are useful for the tree kernel system. For example, in the original top left tree in Figure 3, there is no chance that a fragment consisting only of VN and NP – a very common structure and thus useful in calculating tree similarity – will be extracted by the *subset* tree kernel. The reason is that this kernel type does not allow the production rule to be split (in this case the rule expanding the S node). However, after applying Heuristic 1, the fragment equivalent to VP -> VN NP production rule can be easily extracted. Among the three heuristics, the first one contributes the largest part of the improvement; the other two have a very slight effect according to the results of their individual application, though they contribute to the overall performance when all three are combined.

The success of using modified French trees in improving tree kernel performance may of course depend on the data set and even the task in hand, and may not be generalisable. We next explore this question by applying the modification to a different task *and* a different data set.

6.1 Parser Accuracy Prediction

The task we choose is parser accuracy prediction, the aim of which is to predict the accuracy of a parse tree without a reference (QE for parsing). The task was previously explored for English by Ravi et al.

[12] We also see a slightly smaller improvement for the hand-crafted features using the modified French trees. The combination of tree kernels and hand-crafted features with the modified trees leads to a statistically significant improvement over the combination with the original trees.

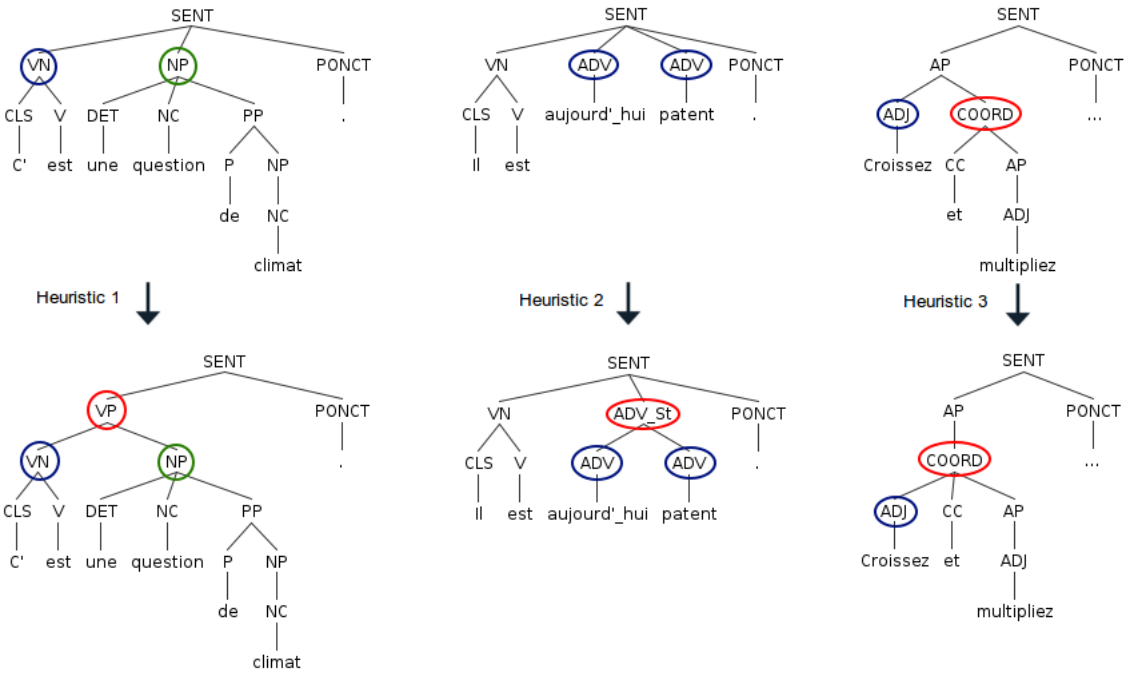


Figure 3: Application of tree modification heuristics on example French translation parse trees

	RMSE	r
PAP	0.1239	0.4035
PAP_m	0.1233	**0.4197**

Table 7: Parser Accuracy Prediction (PAP) performance with tree kernels using original and modified French trees ($_m$)

(2008). We build a tree kernel model to predict the accuracy of French parses. To train the system, we parse the training section of FTB with our French parser and score them using F_1. We use the FTB development set to tune the SVM C parameter and test the model on the FTB test set. Two parser accuracy prediction models are then built using this setting, one with the original parse trees and the second with the modified parse trees produced using the three heuristics listed above. The results are presented in Table 7.

Both RMSE and Pearson r improve with the modified trees, where the r improvement is statistically significant. Although the improvement we observe is not as large as the one we observed for the QE for MT task, the results add weight to our claim that the structure of the FTB trees should be optimised for use in tree kernel learning.

7 Conclusion

We analysed the utility of syntactic information in QE of English-French MT and found it useful both individually and combined with standard QE features. We found that tree kernels are a convenient and effective way of encoding syntactic knowledge but that our hand-crafted feature set also brings additional, useful information. As a result of comparing the role of source and target syntax, we also found that the constituent structure in the FTB could be amended to be more useful in QE for MT and parser accuracy prediction. Now that we have explored the role of syntax in this project, our next step is try to further improve our QE system by adding semantic information. However, there are many other ways in which the research in this paper could be further extended. Our focus is on the language pair English-French and the QE task but it would certainly be interesting to perform a similar analysis on the role of syntax in QE for other language pairs, or to investigate the impact of French tree modification on other tasks.

Acknowledgments

This research has been supported by the Irish Research Council Enterprise Partnership Scheme (EPSPG/2011/102 and EPSPD/2011/135) and the computing infrastructure of the Centre for Next Generation Localisation at Dublin City University. We are grateful to Djamé Seddah for useful discussions about the French Treebank. We also thank the reviewers for their helpful comments.

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for French. In *Treebanks: Building and Using Syntactically Annotated Corpora*. Kluwer Academic Publishers.

Mohammed Attia, Jennifer Foster, Deirdre Hogan, Joseph Le Roux, Lamia Tounsi, and Josef van Genabith. 2010. Handling unknown words in statistical latent-variable parsing models for Arabic, English and French. In *Proceedings of the 1st Workshop on Statistical Parsing of Morphologically Rich Languages*.

Eleftherios Avramidis. 2012. Quality estimation for machine translation output using linguistic analysis and decoding features. In *Proceedings of WMT*.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2003. Confidence estimation for machine translation. In *JHU/CLSP Summer Workshop Final Report*.

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 workshop on statistical machine translation. In *Proceedings of the 8th WMT*.

Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2007. (meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh WMT*.

Marie Candito, Benot Crabb, and Pascal Denis. 2010. Statistical French dependency parsing: treebank conversion and first results. In *Proceedings of LREC*.

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

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Proceedings of the COLING Workshop on Cross-Framework and Cross-Domain Parser Evaluation*.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of WMT*.

Michael Gamon, Anthony Aue, and Martine Smets. 2005. Sentence-level MT evaluation without reference translations: beyond language modeling. In *EAMT*.

Olivier Hamon, Antony Hartley, Andréi Popescu-Belis, and Khalid Choukri. 2007. Assessing human and automated quality judgments in the french MT evaluation campaign CESTA. In *Proceedings of the MT Summit*.

Christian Hardmeier, Joakim Nivre, and Jörg Tiedemann. 2012. Tree kernels for machine translation quality estimation. In *Proceedings of the WMT*.

Rasoul Samed Zadeh Kaljahi, Jennifer Foster, Raphael Rubino, Johann Roturier, and Fred Hollowood. 2013. Parser accuracy in quality estimation of machine translation: A tree kernel approach. In *Proceedings of IJCNLP*.

Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the ACL*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Alessandro Moschitti. 2006. Making tree kernels practical for natural language learning. In *Proceedings of EACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the ACL*.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact and interpretable tree annotation. In *Proceedings of the 21st COLING-ACL*.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of LREC*.

Chris Quirk and Simon Corston-Oliver. 2006. The impact of parse quality on syntactically-informed statistical machine translation. In *Proceedings of EMNLP*.

Chris Quirk. 2004. Training a sentence-level machine translation confidence measure. In *Proceedings of LREC*.

Sujith Ravi, Kevin Knight, and Radu Soricut. 2008. Automatic prediction of parser accuracy. In *Proceedings of EMNLP*.

Raphael Rubino, Jennifer Foster, Joachim Wagner, Johann Roturier, Rasoul Kaljahi, and Fred Hollowood. 2012. DCU-Symantec submission for the WMT 2012 quality estimation task. In *Proceedings of WMT*.

Natalie Schluter and Josef van Genabith. 2007. Preparing, restructuring, and augmenting a french treebank: Lexicalised parsers or coherent treebanks? In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*.

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

Lucia Specia and Jesús Giménez. 2010. Combining confidence estimation and reference-based metrics for segment level mt evaluation. In *Proceedings of AMTA*.

Lucia Specia, Marco Turchi, Nicola Cancedda, Marc Dymetman, and Nello Cristianini. 2009. Estimating the sentence-level quality of machine translation systems. In *EAMT*, pages 28–35.

Zhaopeng Tu, Yifan He, Jennifer Foster, Josef van Genabith, Qun Liu, and Shouxun Lin. 2012. Identifying high-impact sub-structures for convolution kernels in document-level sentiment classification. In *Proceedings of the ACL*.

Nicola Ueffing, Klaus Macherey, and Hermann Ney. 2003. Confidence measures for statistical machine translation. In *Machine Translation Summit IX*.

Michael Wiegand and Dietrich Klakow. 2010. Convolution kernels for opinion holder extraction. In *Proceedings of NAACL-HLT*.

Effective Incorporation of Source Syntax into Hierarchical Phrase-based Translation

Tong Xiao†‡, Adrià de Gispert§, Jingbo Zhu†‡, Bill Byrne§
† Northeastern University, Shenyang 110819, China
‡ Hangzhou YaTuo Company, Hangzhou 310012, China
§ University of Cambridge, CB2 1PZ Cambridge, U.K.
{xiaotong,zhujingbo}@mail.neu.edu.cn
{ad465,wjb31}@eng.cam.ac.uk

Abstract

In this paper we explicitly consider source language syntactic information in both rule extraction and decoding for hierarchical phrase-based translation. We obtain tree-to-string rules by the GHKM method and use them to complement Hiero-style rules. All these rules are then employed to decode new sentences with source language parse trees. We experiment with our approach in a state-of-the-art Chinese-English system and demonstrate +1.2 and +0.8 BLEU improvements on the NIST newswire and web evaluation data of MT08 and MT12.

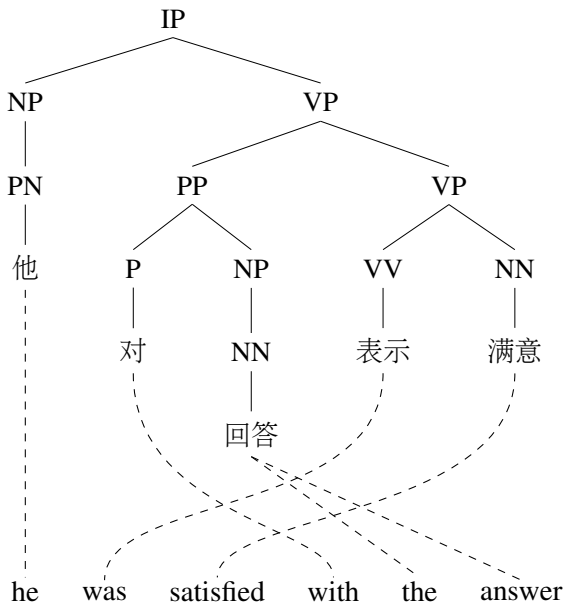
1 Introduction

Synchronous context free grammars (SCFGs) are widely used in statistical machine translation (SMT), with hierarchical phrase-based translation (Chiang, 2005) as the dominant approach. Hiero grammars are easily extracted from word-aligned parallel corpora and can capture complex nested translation relationships. Hiero grammars are formally syntactic, but rules are not constrained by source or target language syntax. This lack of constraint can lead to intractable decoding and bad performance due to the over-generation of derivations in translation. To avoid these problems, the extraction and application of SCFG rules is typically constrained by a source language span limit; (non-glue) rules are lexicalised; and rules are limited to two non-terminals which are not allowed to be adjacent in the source language. These constraints can yield good performing translation systems, although at a sacrifice in the ability to model long-distance movement and complex reordering of multiple constituents.

By contrast, the GHKM approach to translation (Galley et al., 2006) relies on a syntactic parse on either the source or target language side to guide SCFG extraction and translation. The parse tree provides linguistically-motivated constraints both in grammar extraction and in translation. This allows for looser span constraints; rules need not be lexicalised; and rules can have more than two non-terminals to model complex reordering multiple constituents. There are also modelling benefits as more meaningful features can be used to encourage derivations with "well-formed" syntactic tree structures. However, GHKM can have robustness problems in that translation relies on the quality of the parse tree and the diversity of rule types can lead to sparsity and limited coverage.

In this paper we describe a simple but effective approach to introducing source language syntax into hierarchical phrase-based translation to get the benefits of both approaches. Unlike previous work, we do not resort to soft/hard syntactic constraints (Marton and Resnik, 2008; Li et al., 2013) or Hiero-style rule extraction algorithms for incorporating syntactic annotation into SCFGs (Zollmann and Venugopal, 2006; Zhao and Al-Onaizan, 2008; Chiang, 2010). We instead use GHKM syntactic rules to augment the baseline Hiero grammar and decoder. Our approach uses GHKM rules if possible and Hiero rules if not. We report performance on a state-of-the-art Chinese-English system. In a large-scale NIST evaluation task, we find significant improvements of over 1.2 and 0.8 BLEU relative to a strong Hiero baseline on the newswire and web evaluation data of MT08 and MT12. We also investigate variations in the GHKM formalism and find, for example, that our approach works well with binarized trees.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2064–2074, Dublin, Ireland, August 23-29 2014.



Hiero-style SCFG Rules

h_1	X → ⟨他, he⟩
h_2	X → ⟨对, with⟩
h_3	X → ⟨回答, the answer⟩
h_4	X → ⟨表示 满意, was satisfied⟩
h_5	X → ⟨X_1 表示 满意, was satisfied X_1⟩
h_6	X → ⟨X_1 表示 X_2, was X_2 X_1⟩
h_7	X → ⟨X_1 对 X_2 表示 满意, X_1 was satisfied with X_2⟩

Tree-to-String Rules

r_1	NP(PN(他)) → he
r_2	P(对) → with
r_3	NP(NN(回答)) → the answer
r_4	VP(VV(表示) NN(满意)) → was satisfied
r_5	PP(x_1:P x_2:NP) → x_1 x_2
r_6	VP(x_1:PP x_2:VP) → x_2 x_1
r_7	IP(x_1:NP x_2:VP) → x_1 x_2
r_8	VP(PP(P(对) x_1:NP) x_2:VP) → x_2 with x_1

Figure 1: Hiero-syle and tree-to-string rules extracted from a pair of word-aligned Chinese-English sentences with a source language (Chinese) parse tree.

2 Background

2.1 Hierarchical Phrase-based Translation

In the hierarchical phrase-based approach, translation is modelled using SCFGs. In general, probabilistic SCFGs can be learned from word-aligned parallel data using heuristic methods (Chiang, 2007). We can first extract initial phrase pairs and then obtain hierarchical phrase rules (i.e., rules with non-terminals on the right hand side). Once the SCFG is obtained, new sentences can be decoded by finding the most likely derivation of SCFG rules. See Figure 1 for example rules extracted from a sentence pair with word alignments. A sequence of such rules covering the words of the source sentence is a SCFG derivation, e.g., rules h_7, h_1 and h_3 generate a derivation for the sentence pair.

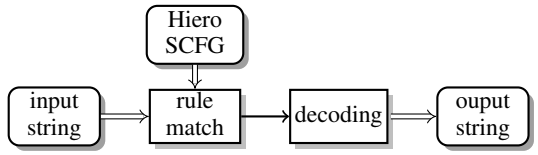
The Hiero SCFG allows vast numbers of derivations which can make unconstrained decoding intractable. In practice, several constraints are applied to control the model size and reduce ambiguity. Typically these are: (a) a rule span limit to be applied in decoding and sometimes also in rule extraction, set to 10; (b) a limit on the rank of the grammar (number of non-terminals that can appear on a rule), set to 2; and (c) a prohibition of consecutive non-terminals on the source language side of a rule (except the glue rules).

2.2 Tree-to-String Translation

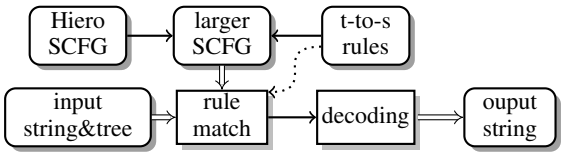
Instead of modelling the problem based on surface strings, tree-to-string systems model the translation equivalency relations from source language syntactic trees to target language strings using derivations of tree-to-string rules (Liu et al., 2006; Mi et al., 2008; Huang and Mi, 2010; Feng et al., 2012). A tree-to-string rule is a tuple $\langle s_r, t_r, \sim \rangle$, where s_r is a source language tree-fragment with terminals and non-terminals at leaves; t_r is a string of target-language terminals and non-terminals; and $\sim$ is a 1-to-1 alignment between the non-terminals of s_r and t_r, for example, VP(VV(提高) x_1:NN) → increases x_1 is a tree-to-string rule, where the non-terminals labeled with the same index x_1 indicate the alignment.

To obtain tree-to-string rules, a popular way is to perform the GHKM rule extraction (Galley et al., 2006) on the bilingual sentences with both word alignment and source (or target) language phrase-structure tree annotations. In GHKM extraction, we first compute the set of the minimally-sized translation rules that can explain the mappings between source language tree and target-language string while respecting the alignment and reordering between the two languages. More complex rules are then learned by composing two or more minimal rules. See Figure 1 for rules extracted using GHKM.

One of the advantages of the above model is that non-terminals in tree-to-string rules are linguistically



(a) decoding with Hiero rules only



(b) decoding with Hiero and tree-to-string rules

Figure 2: Overview of the Hiero baseline (a) and our approach (b). ⇒ means input or output of the decoder. t-to-s is a short for tree-to-string.

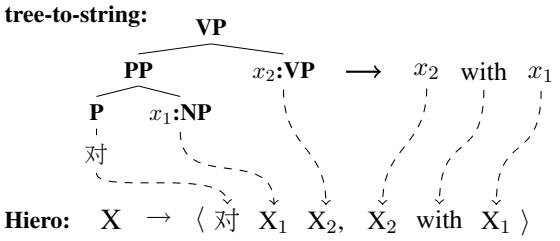


Figure 3: Converting the tree-to-string rule r_8 from Figure 1 to a Hiero-style rule.

motivated and can span word sequences with arbitrary length. Also, one can use rules with consecutive (or more than two) source language non-terminals when the source language parse tree is available. For example, r_8 in Figure 1 has a good Chinese syntactic structure indicating the reordered translations of NP and VP. However, such a rule would not normally be included in a Hiero grammar, as it would require consecutive source language non-terminals (see Figure 3).

3 The Proposed Approach

Both the tree-to-string model and the hierarchical phrase-based model have their own strengths and weaknesses. For example, tree-to-string systems are good at modelling long distance reordering, while hierarchical phrase-based systems are relatively more powerful in handling ill-formed sentences[1] and free translations (Zhao and Al-Onaizan, 2008; Vilar et al., 2010). Here we present a method to enhance hierarchical phrase-based systems with tree-to-string rules and benefit from both models. The idea is simple: we obtain both the tree-to-string grammar and the Hiero-style SCFG from the training data, and then use tree-to-string rules as additional rules in decoding with the SCFG.

Figure 2 shows an overview of our approach and the usual hierarchical phrase-based approach. Our approach requires source language parse trees to be input in both rule extraction and decoding. In rule extraction, we acquire tree-to-string rules using the GHKM method and Hiero-style rules using the Hiero-style rule extraction method to form a larger SCFG. Then, we make use of both the input string and parse tree to decode with the SCFG rules. We now describe our approach.

3.1 Transforming Tree-to-String Rules into SCFG Rules

As described in Section 2, tree-to-string rules have a different form from that of SCFG rules. We will use tree-to-string rules in our hierarchical phrase-based systems by converting each tree-to-string rule into an SCFG rule. The purpose of doing this is to make tree-to-string rules directly accessible to the Hiero-style decoder which performs decoding with SCFG rules.

The rule mapping is straightforward: given a tree-to-string rule $\langle s_r, t_r, \sim \rangle$, we take the frontier nodes of s_r as the source language part of the right hand side of the resulting SCFG rule, and keep t_r and $\sim$ unchanged. Then we replace the non-terminal label with that used in the hierarchical phrase-based system (e.g., X). See Figure 3 for rule mapping of rule r_8 of Figure 1.

In this way, every tree-to-string rule is associated with exactly one SCFG rule. Therefore we can obtain a larger SCFG by combining the rules from the original Hiero-style SCFG and the transformed tree-to-string rules. As explained next, to prevent computational problems we will apply these new rules

[1]For example, the parser fails for 4% of the sentences in our training corpus, and 3% and 6% of the newswire and web development/test sentences, indicating that the data is sometimes ill-formed.

only on the spans that are consistent with the input parse trees. The main goal is to use the tree and the adapted tree-to-string rules to provide the decoder with new linguistically-sensible translation hypotheses that may be prevented by the usual Hiero constraints, and to do so without incurring a computational explosion.

We categorize SCFG rules into two categories based on their availability in Hiero and GHKM extraction. If an SCFG rule is obtained from Hiero extraction, it is a *type 1* rule; If not (i.e., this rule is only available in GHKM extraction), it is a *type 2* rule. E.g., the SCFG rule in Figure 3 is a type 2 rule because it is not available in the original Hiero-style SCFG but can be generated from the tree-to-string rule.

Next we describe how each of these rule types are applied in decoding. We also describe which features are used and how they are computed for each rule type.

3.2 Decoding

Both types of SCFG rules can be employed by usual Hiero decoders with a slight modification. Here we follow the description of Hiero decoding by Iglesias et al. (2011). The source sentence is parsed under the Hiero grammar using the CYK algorithm. Each cell in the CYK grid has associated with it a list of rules that apply to its span; these rules are used to construct a recursive transition network (RTN) which represents all translations of the source sentence under the grammar. The RTN is expanded to a weighted finite state automaton for composition with n-gram language models (de Gispert et al., 2010). Translations are produced via shortest path computation.

This procedure accommodates type 1 rules directly. For tree-to-string rules associated with type 2, we attempt to match rules to the source syntactic tree. If a match is found: the source span of the matching tree fragment is noted and the CYK cell for that span is selected; the tree-to-string rule is converted to a Hiero-style rule; and that rule is added to the list of rules in the selected CYK cell. Once this process is finished, RTN construction, expansion, and language model composition proceeds as usual. Similar modifications could be made to incorporate these rules into cube pruning (Chiang, 2007), cube growing (Huang and Chiang, 2007), and PDT intersection and expansion (Iglesias et al., 2011). We now elaborate on the rule matching strategy.

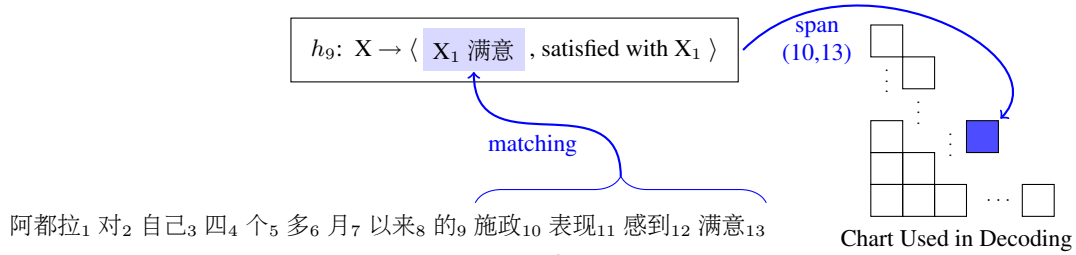
Type 1 Rules The source sentence is parsed as is usual in Hiero-style translation, with the exception that we impose no span limit on rule applications for source spans corresponding to constituents in the Chinese syntactic tree. Rule matching, the procedure that determines if a rule applies to a source span, is based on string matching (see Figure 4(a)). For example, the type 1 rule h_9 in Figure 4(c) can be applied to spans (1,13) and (2,13) since both of them agree with tree constituents (see Figure 4(b)). But h_9 is not applied to span (3,13) because that span is longer than 10 words and agrees with no syntactic tree constituent.

Type 2 Rules If the source side of a tree-to-string rule matches an input tree fragment: 1) that rule is converted to a Hiero-style SCFG rule (Section 3.1); and 2) the Hiero-style rule is added to the rules linked with the CYK grid cell associated with the span of the source syntactic tree fragment. Here, rules are applied via tree matching. For example, rule h_{11} in Figure 4(b) matches the tree fragment spanning positions (2,13).

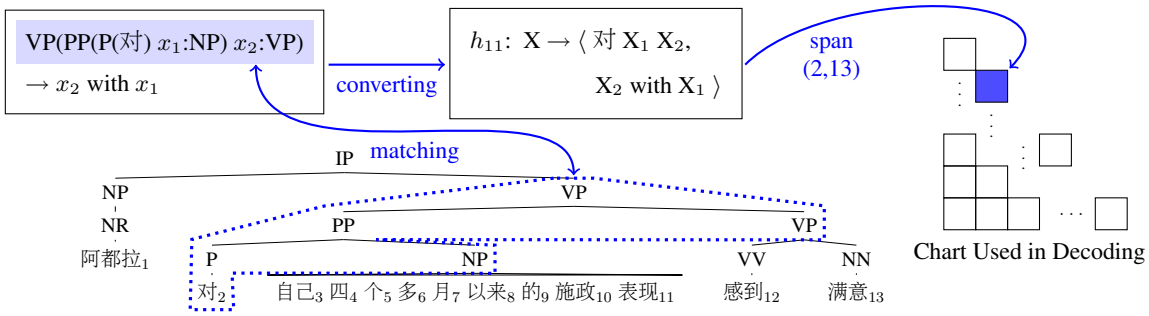
It is worth noting that some type 1 rules may be found via both Hiero-style and tree-to-string grammar extraction. In this case we monitor whether a rule can be applied as a tree-to-string rule using tree-matching so that features (Section 3.3) and weights can be set appropriately. As an example, rule h_{10} in Figure 4 is available in both extraction methods. For span (2,11), this rule can be matched via both string matching and tree matching. We then note that we can apply h_{10} as a tree-to-string rule for span (2, 11) and activate the corresponding features defined in Section 3.3. For other spans (e.g., spans (2,3)-(2,10)), no tree fragments can be matched and the baseline features are used for h_{10}.

3.3 Features

The baseline feature set used in this work consists of 12 features (Pino et al., 2013), including a 4-gram language model, a strong 5-gram language model, bidirectional translation probabilities, bidirectional lexical weights, a word count, a phrase count, a glue rule count, a frequency-1 rule count, a frequency-2



(a) matching a type 1 rule (h_9) with the input string



(b) matching a type 2 rule (h_{11}) with the input parse tree

ID	Type	Hiero-style Rule	Tree-to-string Rule	Applicable Spans
h_8	type 1	X → ⟨ 感到 满意, is satisfied ⟩	N/A	(12,13)
h_9	type 1	X → ⟨ X$_1$ 满意, satisfied with X$_1$ ⟩	N/A	(i,13), $i = 1, 2$ or $4 \leq i \leq 12$
h_{10}	type 1	X → ⟨ 对 X$_1$, with X$_1$ ⟩	PP(P(对) x_1NP) → with NPx_1	(2,j), $3 \leq j \leq 11$ or $j = 13$
h_{11}	type 2	X → ⟨ 对 X$_1$ X$_2$, X$_2$ with X$_1$ ⟩	VP(PP(P(对) x_1:NP) x_2:VP) → x_2 with x_1	(2,13)

(c) example rules used in decoding

Figure 4: Decoding with both Hiero-style and tree-to-string grammars (span limit = 10). A span (i,j) means spanning from position i to position j.

rule count, and a larger-than-frequency-2 rule count [2]. In addition, we introduce several features for applying tree-to-string rules.

- **Rule type indicators**. We consider four indicator features, indicating tree-to-string rules, lexicalized tree-to-string rules, rules with consecutive non-terminals, and non-lexicalized rules. Note that the tree-to-string rule indicator feature is in principle a generalization of the soft syntactic features (Marton and Resnik, 2008), in that a bonus (or penalty) is applied when a rule application is consistent with a source tree constituent. The difference lies in that the tree-to-string rule indicator feature does not distinguish between different syntactic labels, whereas soft syntactic features do.

- **Features in syntactic MT**. In general tree-to-string rules have their own features which are different from those used in Hiero-style systems. For example, the features in syntactic MT systems can be defined as the generation probabilities conditioned on the root symbol of the tree-fragment. Here we choose five popular features used in syntactic MT systems, including the bi-directional phrase-based conditional translation probabilities (Marcu et al., 2006) and three syntax-based conditional probabilities (Mi and Huang, 2008). All these probabilities can be computed by relative-frequency estimates. For example, the phrase-based features are the probabilities of translating between the frontier nodes of s_r and t_r. The syntax-based features are the probabilities of generating r conditioned on its root,

[2]We experimented with soft syntactic features (Marton and Resnik, 2008) but found no improvement over our baseline system.

source and target language sides, respectively. More formally, we use the following estimates for these probabilities:

$$\begin{aligned}
\mathrm{P}_{phr}(t_r \mid s_r) &= \frac{\sum_{r'':\varphi(s_{r''})=\varphi(s_r)\wedge t_{r''}=t_r} c(r'')}{\sum_{r':\varphi(s_{r'})=\varphi(s_r)} c(r')} \\
\mathrm{P}_{phr}(s_r \mid t_r) &= \frac{\sum_{r'':\varphi(s_{r''})=\varphi(s_r)\wedge t_{r''}=t_r} c(r'')}{\sum_{r':t_{r'}=t_r} c(r')} \\
\mathrm{P}(r \mid root(r)) &= \frac{c(r)}{\sum_{r':root(r')=root(r)} c(r')} \\
\mathrm{P}(r \mid s_r) &= \frac{c(r)}{\sum_{r':s_{r'}=s_r} c(r')} \\
\mathrm{P}(r \mid t_r) &= \frac{c(r)}{\sum_{r':t_{r'}=t_r} c(r')}
\end{aligned}$$

where $c(r)$ is the count of r, and $root(\cdot)$ and $\varphi(\cdot)$ are functions that return the source root symbol for a tree-to-string rule and the sequence of leaf nodes for a tree-fragment respectively.

4 Evaluation

4.1 Experimental Setup

We report results in the NIST MT12 Chinese-English task, where our baseline system was among the top academic systems. The parallel training corpus consists of 9.2 million sentence pairs which are provided within the NIST Chinese-English MT12 track. Word alignments are obtained using MTTK (Deng and Byrne, 2008) in both Chinese-to-English and English-to-Chinese directions, and then unioning the links. The data from newswire and web genres was used for tuning and test. The development sets contain 1,755 sentences and 2160 sentences for the two genres respectively. The test sets (newswire: 1,779 sentences, web: 1768 sentences) contain all newswire and web evaluation data of MT08 (mt08), MT12 (mt12), and MT08 progress test (mt08.p). All Chinese sentences in the training, development and test sets were parsed using the Berkeley parser (Petrov and Klein, 2007). A Kneser-Ney 4-gram language model was trained on the AFP and Xinhua portions of the English Gigaword in addition to the English side of the parallel corpus. A stronger 5-gram language model was trained on all English data of NIST MT12 and the Google counts corpus using the "stupid" backoff method (Brants et al., 2007).

For decoding we use HiFST, which is implemented with weighted finite state transducers (de Gispert et al., 2010). A two-pass decoding strategy is adopted; first, only the 4-gram language model and the translation model are activated; and then, the 5-gram language model is applied for second-pass rescoring of the translation lattices generated by the first-pass decoding stage. We extracted SCFG rules from the parallel corpus using the standard heuristics (Chiang, 2007) and filtering strategies (Iglesias et al., 2009). The span limit was set to 10 in extracting basic phrases and decoding. All features weights were optimized using lattice-based minimum error rate training (Macherey et al., 2008).

For tree-to-string extraction, we used a reimplementation of the GHKM method (Xiao et al., 2012) and extracted rules from a 600K-sentence portion of the parallel data. To prune the tree-to-string rule set, we restricted the extraction to rules with at most 5 frontier non-terminals and 5 terminals. Also, we discarded lexicalized rules with a Chinese-to-English translation probability of < 0.02 and non-lexicalized rules with a Chinese-to-English translation probability of < 0.10.

4.2 Results

We report MT performance in Table 1 by case-insensitive BLEU (Papineni et al., 2002). The experiments are organized as follows:

- Baseline and Span Limits (exp01 and exp02)
  First we study the effect of removing the span limit for tree constituents, that is, SCFG rules can be

Entry	System	Newswire					Web				
		tune (1755)	mt08 (691)	mt12 (400)	mt08.p (688)	all test (1779)	tune (2160)	mt08 (666)	mt12 (420)	mt08.p (682)	all test (1768)
exp01	baseline	35.84	35.85	35.47	35.50	35.63	29.98	25.15	23.07	27.19	25.33
exp02	+= no span limit	36.05	36.08	35.70	35.54	35.79	30.11	25.28	23.08	27.17	25.37
exp03	+= t-to-s rules	36.63	36.51	36.08	36.09	36.25*	30.80	26.00	23.08	27.80	25.83
exp04	+= t-to-s features	36.82	36.49	36.53	36.16	36.38*	30.91	26.03	23.27	27.85	25.98*
exp05	t-to-s baseline	34.63	34.44	34.87	33.66	34.25*	28.30	23.40	21.38	25.30	23.56*
exp06	exp04 on spans > 10	36.17	36.11	35.71	35.86	35.92	30.18	25.30	23.12	27.36	25.45
exp07	exp04 with null trans.	36.10	36.03	35.35	34.86	35.42	29.96	25.32	22.58	23.33	24.12*
exp08	exp04 + left binariz.	37.11	37.46	37.03	36.30	36.91*	31.18	26.15	23.54	27.98	26.13*
exp09	exp04 + right binariz.	36.58	36.56	36.41	35.70	36.20*	31.06	25.94	23.47	27.48	25.88*
exp10	exp04 + forest binariz.	37.03	37.27	37.09	36.62	36.98*	31.20	25.99	23.59	28.09	26.15*

Table 1: Case-insensitive BLEU[%] scores of various systems. += means incrementally adding methods/features to the previous system. * means that a system is significantly different than the exp01 baseline at $p < 0.01$.

applied to any spans when they respect the tree constituents of the input tree. It can be regarded as the simplest way of using source syntax in Hiero-style systems. Seen from Table 1, removing the span limit shows modest BLEU improvements. It agrees with the previous result that loosening the constraints on spans is helpful to systems based on the hard syntactic constraints (Li et al., 2013).

- GHKM+Hiero (exp03 and exp04)
  The results of our proposed approach (w/o new features) are reported in exp03 and exp04. We see that incorporating tree-to-string rules yields +0.6 and +0.5 improvements on the collected newswire and web test sets (exp03 vs exp01). The new features (Section 3.3) give a further improvement (exp04 vs exp03). This result confirms that the system can learn a preference for certain types of rules using the new features.

- Impact of Search Space (exp05)
  We also study the impact of search space on system performance. To do this, we force the improved system (exp04) to respect source tree constituents and to discard any hypotheses which violate the tree constituent constraints. Seen from exp05, this system has a lower BLEU score than both the Hiero baseline (exp01) and GHKM+Hiero system (exp04), strongly suggesting that restricting MT systems to a smaller space of hypotheses is harmful.

- GHKM+Hiero, Spans > 10 Only (exp06)
  Another interesting question is whether tree-to-string rules and features are more helpful to larger spans. We restricted our approach to spans > 10 only and conducted another experiment. As is shown in exp06, applying tree-to-string rules and features for large spans is beneficial (exp06 vs. exp01). But it underperforms the system with the full use of tree-to-string rules (exp06 vs. exp04). This interesting observation implies that applying tree-to-string rules on smaller spans introduces good hypotheses that can be selected with our additional features.

- Impact of Failed Parses (exp07)
  As noted in Section 3, the parser fails to parse some of the sentences in our experiments. In this case our approach generates the baseline result using the Hiero model (i.e., type 1 rules only). To investigate the effect of failed parse trees on system performance, we also report the BLEU score including null translations for which the parser fails. As shown in exp07, there are significantly lower BLEU scores when null translations are included. It indicates that our approach is more robust than standard tree-to-string systems which would generate an empty translation if the source language parser fails.

- Results on Binarization (exp08-10)
  Tree binarization is a widely used method to improve syntactic MT systems (Wang et al., 2010). exp08-10 show the results of our improved system with left-heavy, right-heavy and forest-based bina-

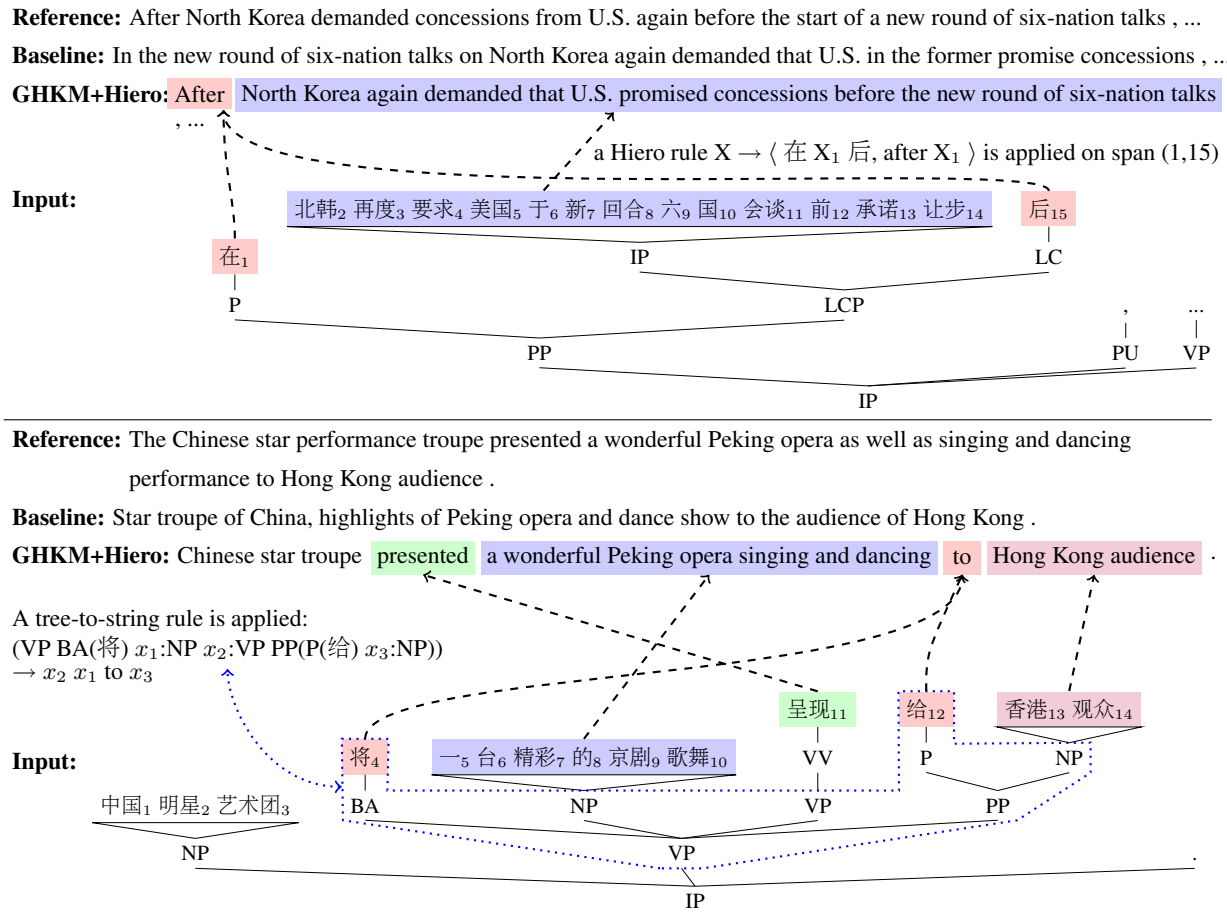


Figure 5: Comparison of translations generated by the baseline and improved systems.

rization[3]. We see that left-heavy binarization is very helpful and exp08 achieves overall improvements of 1.2 and 0.8 BLEU points on the newsire and web data. In contrast, right-heavy binarization does not yield promising performance. This agrees with the previous report (Wang et al., 2010) that MT systems prefer to use certain ways of binarization in most cases. exp10 shows that the additional trees introduced in our forest-based scheme are not sufficient to make a big impact on BLEU scores. Possibly larger gains can be obtained if taking a forest of parse trees from the source parser, but this is outside the scope of this paper.

4.3 Analysis

We then analyse rule usage in the 1-best derivations for our improved system on the tuning set. We find that type 2 rules represent 13.97% of the rules used in the 1-best derivations. Also, 44.45% of the applied rules are available from the tree-to-string model (i.e., rules that use the features described in Section 3.3). These numbers indicate that the tree-to-string rules are beneficial and our decoder likes to use them.

Finally, we discuss two real translation examples from our tuning set. See Figure 5 for translations generated by different systems. In the first example, the Chinese input sentence contains 在 ... 后 which is usually translated into *after* ... (i.e., a Hiero rule X → ⟨ 在 X_1 后, after X_1 ⟩). However, because the " 在 ... 后" pattern spans 15 words and that is beyond the span limit, our baseline is unable to apply this desired rule and chooses a wrong translation *in* for the Chinese word 在. When the source parse tree

[3]We found that the CTB-style parse trees usually have a very flat top-level IP (i.e., single clause) tree structure. As the IP structure in Chinese is very complicated, the system might prefer a flexible binarization scheme. Thus we considered both left and right-heavy binarization to form a binarization forest for IPs in Chinese parse trees, and binarized other tree constituents in a left-heavy fashion.

is available, our approach removes the span limit for spans that agree with the tree constituents. In this case, the MT system successfully applies the rule on span (1, 15) and generates a much better translation.

In the second example, the translation of the input sentence requires complex reordering of adjacent constituents. The baseline system cannot handle this case and generates a monotonic translation using the glue rules. This results in a wrong order for the translation of Chinese verb 呈现 (*show*). By contrast, the improved system chooses a tree-to-string rule with three non-terminals (some of which are adjacent in the source language) and perfectly performs a syntactic movement of the required tree constituents.

5 Related Work

Recently linguistically-motivated models have been intensively investigated in MT. In particular, source tree-based models (Liu et al., 2006; Huang et al., 2006; Eisner, 2003; Zhang et al., 2008; Liu et al., 2009a; Xie et al., 2011) have received growing interest due to their good abilities in modelling source language syntax for better lexicon selection and reordering. Alternatively, the hierarchical phrase-based approach (Chiang, 2005) considers the underlying hierarchical structures of sentences but does not require linguistically syntactic trees on either language side.

There are several lines of work for augmenting hierarchical phrase-based systems with the use of source language phrase-structure trees. Liu et al. (2009b) describe novel approaches to translation under multiple translation grammars. Their approach is very much motivated by system combination, and they develop procedures for joint decoding and optimisation within a single system that give the benefit of combining hypotheses from multiple systems. They demonstrate their approach by combining full tree-to-string and Hiero systems. Our approach is much simpler and emphasises changes to the grammar rather than the decoder or its parameter optimisation (MERT). Our aim is to augment the search space of Hiero with linguistically-motivated hypotheses, and not to develop a new decoder that is capable of translation under multiple grammars. Moreover, we consider Hiero as the backbone model and only introduce tree-to-string rules where they can contribute; we show that extracting tree-to-string rules from just 10% of the data suffices to get good gains. This results in a small number of tree-to-string rules and does not slow down the decoder.

Another related line of work is to introduce syntactic constraints or annotations to hierarchical phrase-based systems. Marton and Resnik (2008) and Li et al. (2013) proposed several soft or hard constraints to model syntactic compatibility of Hiero derivations and input source language parse trees. We note that, despite significant development effort, we were not able to improve our baseline through the use of these soft syntactic constraints; it was this experience that led us to develop the hybrid approach described in this paper.

Several research groups used syntactic labels as non-terminal symbols in their SCFG rules and develop new features (Zollmann and Venugopal, 2006; Zhao and Al-Onaizan, 2008; Chiang, 2010; Hoang and Koehn, 2010). However, all these methods still resort to rule extraction procedures similar to that of the standard phrase/hierarchical rule extraction method. In contrast, we use the GHKM method which is a mature technique to extract rules from tree-string pairs but does not impose those Hiero-style constraints on rule extraction. More importantly, we consider the hierarchical syntactic tree structure to make use of well-formed rules in decoding, while such information is not used in standard SCFG-based systems. We also keep to the simpler non-terminals of Hiero, and do not 'decorate' any non-terminals with syntactic or other information.

6 Conclusion

We have presented an approach to improving Hiero-style systems by augmenting the SCFG with tree-to-string rules and syntax-based features. The input parse trees are used to introduce new linguistically-sensible hypotheses into the translation search space while maintaining the Hiero robustness qualities and avoiding computational explosion. We obtain significant improvements over a strong Hiero baseline in Chinese-to-English. Further improvements are achieved when applying tree binarization.

Acknowledgements

This work was done while the first author was visiting the speech group at University of Cambridge, and was supported in part by the National Science Foundation of China (Grants 61272376 and 61300097), and the China Postdoctoral Science Foundation (Grant 2013M530131). We would like to thank the anonymous reviewers for their pertinent and insightful comments. We also would like to thank Juan Pino, Rory Waite, Federico Flego and Gonzalo Iglesias for building parts of the baseline system.

References

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large Language Models in Machine Translation. In *Proceedings of EMNLP-CoNLL*, pages 858–867, Prague, Czech Republic.

David Chiang. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proceedings of ACL*, pages 263–270, Ann Arbor, Michigan, USA.

David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33:45–60.

David Chiang. 2010. Learning to Translate with Source and Target Syntax. In *Proceedings of ACL*, pages 1443–1452, Uppsala, Sweden.

Adrià de Gispert, Gonzalo Iglesias, Graeme Blackwood, Eduardo R. Banga, and William Byrne. 2010. Hierarchical Phrase-Based Translation with Weighted Finite-State Transducers and Shallow-n Grammars. *Computational Linguistics*, 36(3):505–533.

Yonggang Deng and William Byrne. 2008. HMM Word and Phrase Alignment for Statistical Machine Translation. *IEEE Transactions on Audio, Speech & Language Processing*, 16(3):494–507.

Jason Eisner. 2003. Learning Non-Isomorphic Tree Mappings for Machine Translation. In *Proceedings of ACL*, pages 205–208, Sapporo, Japan.

Yang Feng, Yang Liu, Qun Liu, and Trevor Cohn. 2012. Left-to-Right Tree-to-String Decoding with Prediction. In *Proceedings of EMNLP-CoNLL*, pages 1191–1200, Jeju Island, Korea.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable Inference and Training of Context-Rich Syntactic Translation Models. In *Proceedings of COLING-ACL*, pages 961–968, Sydney, Australia.

Hieu Hoang and Philipp Koehn. 2010. Improved translation with source syntax labels. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 409–417, Uppsala, Sweden.

Liang Huang and David Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of ACL*, pages 144–151, Prague, Czech Republic.

Liang Huang and Haitao Mi. 2010. Efficient Incremental Decoding for Tree-to-String Translation. In *Proceedings of EMNLP*, pages 273–283, Cambridge, MA, USA.

Liang Huang, Knight Kevin, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, pages 66–73, Cambridge, MA, USA.

Gonzalo Iglesias, Adrià de Gispert, Eduardo R. Banga, and William Byrne. 2009. Rule Filtering by Pattern for Efficient Hierarchical Translation. In *Proceedings of EACL*, pages 380–388, Athens, Greece.

Gonzalo Iglesias, Cyril Allauzen, William Byrne, Adrià de Gispert, and Michael Riley. 2011. Hierarchical Phrase-based Translation Representations. In *Proceedings of EMNLP*, pages 1373–1383, Edinburgh, Scotland, UK.

Junhui Li, Philip Resnik, and Hal Daumé III. 2013. Modeling Syntactic and Semantic Structures in Hierarchical Phrase-based Translation. In *Proceedings of NAACL-HLT*, pages 540–549, Atlanta, Georgia.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proceedings of COLING-ACL*, pages 609–616, Sydney, Australia.

Yang Liu, Yajuan Lü, and Qun Liu. 2009a. Improving Tree-to-Tree Translation with Packed Forests. In *Proceedings of ACL-IJCNLP*, pages 558–566, Suntec, Singapore.

Yang Liu, Haitao Mi, Yang Feng, and Qun Liu. 2009b. Joint decoding with multiple translation models. In *Proceedings of ACL-IJCNLP*, pages 576–584, Suntec, Singapore.

Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based Minimum Error Rate Training for Statistical Machine Translation. In *Proceedings of EMNLP*, pages 725–734, Honolulu, Hawaii.

Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. In *Proceedings of EMNLP*, pages 44–52, Sydney, Australia.

Yuval Marton and Philip Resnik. 2008. Soft Syntactic Constraints for Hierarchical Phrased-Based Translation. In *Proceedings of ACL-HLT*, pages 1003–1011, Columbus, Ohio.

Haitao Mi and Liang Huang. 2008. Forest-based Translation Rule Extraction. In *Proceedings of EMNLP*, pages 206–214, Honolulu, Hawaii, USA.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-Based Translation. In *Proceedings of ACL-HLT*, pages 192–199, Columbus, Ohio.

Kishore Papineni, Salim Roukos, Todd Ward, and Weijing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL*, pages 311–318, Philadelphia, PA, USA.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HLT-NAACL*, pages 404–411, Rochester, New York, USA.

Juan Pino, Aurelien Waite, Tong Xiao, Adrià de Gispert, Federico Flego, and William Byrne. 2013. The University of Cambridge Russian-English system at WMT13. In *Proceedings of WMT*, pages 200–205, Sofia, Bulgaria.

David Vilar, Daniel Stein, Stephan Peitz, and Hermann Ney. 2010. If i only had a parser: poor man's syntax for hierarchical machine translation. In *Proceedings of IWSLT*, pages 345–352.

Wei Wang, Jonathan May, Kevin Knight, and Daniel Marcu. 2010. Re-structuring, Re-labeling, and Re-aligning for Syntax-Based Machine Translation. *Computational Linguistics*, 36(2):247–277.

Tong Xiao, Jingbo Zhu, Hao Zhang, and Qiang Li. 2012. NiuTrans: An Open Source Toolkit for Phrase-based and Syntax-based Machine Translation. In *Proceedings of ACL: System Demonstrations*, pages 19–24, Jeju Island, Korea.

Jun Xie, Haitao Mi, and Qun Liu. 2011. A novel dependency-to-string model for statistical machine translation. In *Proceedings of EMNLP*, pages 216–226, Edinburgh, Scotland.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A Tree Sequence Alignment-based Tree-to-Tree Translation Model. In *Proceedings of ACL-HLT*, pages 559–567, Columbus, Ohio, USA.

Bing Zhao and Yaser Al-Onaizan. 2008. Generalizing Local and Non-Local Word-Reordering Patterns for Syntax-Based Machine Translation. In *Proceedings of EMNLP*, pages 572–581, Honolulu, Hawaii.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In *Proceedings of WMT*, pages 138–141, New York City.

BEL: Bagging for Entity Linking

Zhe Zuo, Gjergji Kasneci, Toni Gruetze, Felix Naumann
Hasso Plattner Institute
Prof.-Dr.-Helmert-Straße 2-3, 14482 Potsdam, Germany
{firstname.lastname}@hpi.uni-potsdam.de

Abstract

With recent advances in the areas of knowledge engineering and information extraction, the task of linking textual mentions of named entities to corresponding ones in a knowledge base has received much attention. The rich, structured information in state-of-the-art knowledge bases can be leveraged to facilitate this task. Although recent approaches achieve satisfactory accuracy results, they typically suffer from at least one of the following issues: (1) the linking quality is highly sensitive to the amount of textual information; typically, long textual fragments are needed to capture the context of a mention, (2) the disambiguation uncertainty is not explicitly addressed and often only implicitly represented by the ranking of entities to which a mention could be linked, (3) complex, joint reasoning negatively affects the efficiency.

We propose an entity linking technique that addresses the above issues by (1) operating on a textual range of relevant terms, (2) aggregating decisions from an ensemble of simple classifiers, each of which operates on a randomly sampled subset from the above range, (3) following local reasoning by exploiting previous decisions whenever possible. In extensive experiments on hand-labeled and benchmark datasets, our approach outperformed state-of-the-art entity linking techniques, both in terms of quality and efficiency.

1 Introduction

Named-entity linking (NEL) is the task of establishing a mapping from textual mentions of named entities to canonical representations of those entities in a knowledge base. Often, textual mentions are ambiguous; that is, a mention could refer to multiple named entities, but only one of them is correct in the given textual context. Resolving these ambiguities is often referred to as *named entity disambiguation* (NED), which is a highly challenging aspect of an NEL process. More specifically, a robust NEL algorithm has to robustly resolve ambiguities and thus build on robust NED methods. The NED problem, however, is often ill-posed, as only the right context and background knowledge can help disambiguate entities. In many cases, the contextual information is implicit in nature and may be latently spread across various passages or documents, and background knowledge may not be sufficient, which makes the disambiguation task challenging even for human readers. As an example, consider the sentence: "*London spent $80,000 ($2,040,000 in current value) to build a 15,000-square-foot stone mansion ('Wolf House') on the property.*" A human reader knows that in general money is spent by people, but sometimes also city councils can spend money, and hence, in the above sentence "London" may refer to a person or to the city of London. However, when considering the contextual information, especially the key phrase "Wolf House", and the fact that this was the name of the mansion of the writer Jack London, the disambiguation of "London" becomes obvious.

The NED problem is abundant, and the above subtleties place it right at the heart of many artificial intelligence applications, such as semantic search, machine translation, business intelligence, topic detection, text summarization, machine vision, and many more. In the context of information systems, the problem has been addressed in many different flavors and settings, e.g., in the structured setting of

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2075–2086, Dublin, Ireland, August 23-29 2014.

record-linkage and *duplicate detection*, where the goal is find database records that refer to the same named entity (Bhattacharya and Getoor, 2007; Naumann and Herschel, 2010), in the semi-structured setting of cleaning XML data (Weis and Naumann, 2005) or annotating Web tables (Limaye et al., 2010), in the context of enriching Wikipedia information boxes (Wu and Weld, 2008), for the alignment of knowledge bases (Aumueller et al., 2005; Lacoste-Julien et al., 2013), and most prominently, in the setting of Natural Language Processing (Bagga and Baldwin, 1998; Mann and Yarowsky, 2003; Fleischman and Hovy, 2004; Bunescu and Pasca, 2006; Cucerzan, 2007), which is also the setting of this work.

In the latter setting, the proliferation of clean knowledge bases with rich semantic relations between Web entities, e.g., DBpedia (Auer et al., 2007), Freebase (Bollacker et al., 2008), or YAGO (Suchanek et al., 2007), has given rise to novel, reliable NEL techniques (see Section 2) that exploit the semantic relatedness between entities for the linking process (Shen et al., 2012; Hoffart et al., 2011b; Hoffart et al., 2012).

Our disambiguation model builds on a majority-voting strategy that employs a bagging of multiple ranking classifiers, thus the name BEL: Bagging for Entity Linking. Each ranking classifier operates on a randomly sampled subset of terms surrounding the mention in focus. These terms are sampled from a so-called textual *range of relevant terms*, i.e., terms that are most promising for determining the context of the mention. Finally, based on the sampled terms, each ranking classifier proposes a ranked list of candidate entities and the mention is linked to the entity that is proposed as top-ranked candidate by the majority of the classifiers.

In summary, the main contributions of this work are:

1. A novel ensemble-based disambiguation approach that exploits the terms that surround a textual mention to best capture its context; a parsimonious linking model that combines the above method with a prior probability (similar to the one presented in (Fader et al., 2009), (Hoffart et al., 2011b), or (Lin et al., 2012)) of a candidate named entity being referred to by a given mention yields a highly efficient linking process.

2. An analysis of the disambiguation impact of the components used in BEL on the final linking decision.

3. A detailed quality and efficiency comparison with the state-of-the-art methods of Cucerzan (2007), Hoffart et al. (2011b), and Hoffart et al. (2012) on multiple real-world and synthetic datasets; apart from being more efficient, BEL also achieves a linking quality that is comparable to or even better than that of the above methods.

The remainder of the paper is organized as follows: The next section discusses related work. Section 3 is devoted to our NEL approach. The experimental evaluation is presented in Section 4, before we conclude in Section 5.

2 Background and Related Work

There is a vast array of literature on the topic of resolving ambiguous mentions of named entities. We focus on relevant disambiguation strategies for the NEL problem and leave aside natural language processing (NLP) techniques on named entity recognition and part-of-speech tagging; although, admittedly, for the recognition of entity mentions, such techniques are indispensable. In this work, we assume such NLP techniques are given and use the Stanford NER Tagger (Finkel et al., 2005) to reliably recognize textual mentions of named entities. Another field that we bypass is that of *record linkage* or *duplicate detection*, where the focus is on comparing sets of database records and identifying mappings between records referring to the same entity. Obviously, record linkage methods operate on structured data, such as database entries with a predefined set of attributes (commonly with a known value range), which is different from our NLP setting.

In traditional methods, each mention and each named entity is represented by a vector of terms occurring in its textual context. Vector-based similarity measures are applied to capture the affinity between a mention and a named entity. The feature values can go beyond simple unigram terms and consist

of compound terms, such as bigrams, key phrases, encyclopedic facts, or categorical descriptions. For example, Pedersen et al. (2005) employed salient bigrams to represent the context of a mention; Mann and Yarowsky (2003) included biographic facts into the vector representation of a named entity, whereas Cucerzan (2007) extended the term-based feature set of a Wikipedia entity by information from other articles linking to it, but instead of using the whole article text, only some key phrases and immediate Wikipedia categories were included. Bunescu and Pasca (2006), after deriving an entity dictionary from Wikipedia, for a given mention, rank entities by a kernel-based similarity between the textual context of the mention and the Wikipedia text and categories of the candidate entity. The mention is linked to the most similar entity.

The disambiguation problem has also been formulated as a probabilistic reasoning problem. For example, Fleischman and Hovy (2004) trained a maximum entropy model to infer the probability that two mentions represent the same entity and used a modified agglomerative clustering algorithm to cluster mentions using the probabilistic similarity measure. Similarly, Sil et al. (2012) used a log-linear model to represent the probability of a named entity being referred to by a mention. For both above methods, the selection of features and efficient strategies for learning their weights are crucial, as ideally all feature weights should be learned in a joint fashion, which can be computationally expensive and is often impeded by the "curse of dimensionality".

Note that many of the above techniques model the implicit relatedness between terms (and term compounds), where the general idea is that two terms are related if many Web pages contain both of them. Measures building on this idea were refined and extended in (Milne and Witten, 2008) and (Huang et al., 2012), especially for the relatedness between Wikipedia articles. Such implicit relatedness can lead to a large candidate space; to effectively prune this space, entity prominence priors have been integrated in various recent disambiguation models, e.g., (Fader et al., 2009; Hoffart et al., 2011b; Lin et al., 2012).

Other techniques model explicit, relationship-based similarities between entities; for example, Du et al. (2013) employed similarity measures that captured the average pair-wise proximity between candidate entities in the knowledge graph, as well as their average pair-wise conceptual similarity by means of the lowest-common-ancestor classes. (Hoffart et al., 2011b; Hoffart et al., 2012) exploited the hypernymy- and key-phrase-based relatedness, between the k candidate entities in the knowledge base, to jointly link k mentions occurring in the same paragraph. A prior probability of a candidate entity being referred to by a mention was combined with the above relatedness measures in an objective maximization function. The intuition behind the hypernymy-based relatedness was that in order for k mentions (that occur in the same textual context) to be linked correctly to $l \leq k$ named entities in the knowledge base, the l entities should jointly exhibit a high "semantic" relatedness, which in (Hoffart et al., 2011b) is referred to as *coherence*. Despite this principled modeling of the NEL problem in (Hoffart et al., 2011b; Hoffart et al., 2012) and the impressive quality results reported in those works, efficiency seems to be the main bottleneck of such collective inference models. We argue that a Web-scale NEL process should avoid complex reasoning strategies wherever possible. Concerns along these lines have been also expressed in (Lin et al., 2012), where the authors highlight the need for the application of NEL techniques at Web scale.

The approach presented in this paper, BEL, avoids complex, coherence-based joint reasoning. It also avoids the processing of long textual passages, where multiple mentions have to occur. Instead, we show that a careful light-weight, independent reasoning on the linking of mentions can lead to a linking quality that is comparable to and sometimes even better than the one achieved by the above methods.

3 The BEL Algorithm

In this work, the focus is not on the recognition of named entity mentions in a text but rather on their disambiguation once the mentions are known. Throughout this work we assume that a reliable named entity recognition tool is available. BEL relies on the Stanford NER Tagger (Finkel et al., 2005) to recognize textual mentions of named entities. Once the mentions have been recognized, BEL retrieves promising candidate entities from the knowledge base and employs a careful, majority-voting algorithm to take the best possible linking decision based on the textual context of the mentions. The method is

described in the following subsections.

3.1 High-Level Overview of the BEL Algorithm

Algorithm 1 gives a high-level overview of the BEL approach. The only assumption we make is that the textual corpus from which the knowledge base has been derived is freely available. For example, the textual corpus of knowledge bases such as YAGO or DBpedia is Wikipedia, which is an open source of information about the entities in the two knowledge bases.

Exemplarily, in Algorithm 1, we use the YAGO knowledge base to highlight the main idea of the algorithm. YAGO is a clean knowledge base with structured information about a large proportion of the entities contained in Wikipedia, thus being a popular representative of many state-of-the-art knowledge bases derived from Wikipedia.

Once the set of mentions has been derived from a given document (line 1), for each mention, a list of promising candidates is derived from Wikipedia. The candidates are ranked by a so-called "prominence" score, representing the probability of a Wikipedia article (i.e., the entity represented by the article) being referred to by the mention (lines 2, 3). In case the list of candidates is empty, the corresponding mention is linked to a designated entity, E_{NULL}, meaning that the mention cannot refer to a YAGO entity (lines 4, 5). The same holds for the case that the top-ranked candidate occurs in Wikipedia but not in YAGO (lines 7 - 9). Otherwise, the joint majority decision of multiple bagged ranking classifiers is computed (lines 11 - 13). Only if there is a majority consensus about a candidate (i.e., the candidate is ranked as top candidate by the majority of the classifiers), the mention is linked to that candidate; otherwise, the mention is linked to E_{NULL} (lines 14 - 18).

Algorithm 1 Bagging for Entity Linking Algorithm

Input: document file $\mathbf{D} = (t_1, t_2, \ldots)$, HashMap $\mathbf{V}$ that maps the ID of a ranking classifier to the top-ranked candidate entity by that classifier.
Output: linkage between mentions $\mathbf{M} = \{m_1, m_2, \ldots\}$ in $\mathbf{D}$ and corresponding entities $\mathbf{E} = \{e_1, e_2, \ldots\}$ in YAGO.

1: $\mathbf{M} :=$ *recognizeMentions*($\mathbf{D}$)
2: **for** each mention $m_i \in \mathbf{M}$ **do**
3: $\quad \mathbf{L}_{m_i} :=$ *getTopKCandidates*(k,m_i) /*according to the "prominence" score $S_{PR}(e, m_i)$*/
4: $\quad$ **if** $\mathbf{L}_{m_i}$ is empty **then**
5: $\quad\quad$ link m_i to E_{NULL} /*i.e.,mention cannot be linked*/
6: $\quad$ **else**
7: $\quad\quad$ $e' := arg\,max_{e \in L_{m_i}} S_{PR}(e, m_i)$
8: $\quad\quad$ **if** e' is not in YAGO **then**
9: $\quad\quad\quad$ link m_i to non-YAGO entity E_{NULL}
10: $\quad\quad$ **else**
11: $\quad\quad\quad$ **for** each ranking classifier S_n **do**
12: $\quad\quad\quad\quad$ $\mathbf{V}$.put($n, arg\,max_e(SimScore(e, S_n, m))$)
13: $\quad\quad\quad$ **end for**
14: $\quad\quad\quad$ **if** an e^* occurs more than $\frac{|\mathbf{V}|}{2}$ in $\mathbf{V}.values()$ **then**
15: $\quad\quad\quad\quad$ link m_i to e^*
16: $\quad\quad\quad$ **else**
17: $\quad\quad\quad\quad$ link m_i to non-YAGO entity E_{NULL}
18: $\quad\quad\quad$ **end if**
19: $\quad\quad$ **end if**
20: $\quad$ **end if**
21: **end for**

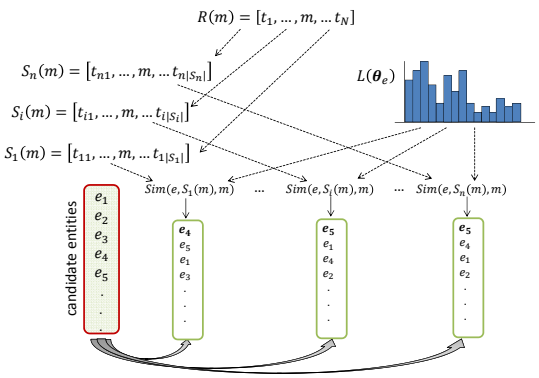


Figure 1: Strategy for generating ranking classifiers, each of which operates on a randomly sampled subset $S_i(m)$ from a set $R(m)$ of relevant terms surrounding m and assigns contextual similarity scores to the candidate entities based on that subset.

In addition, for efficiency reasons, BEL exploits previous disambiguation decisions whenever possible. If a mention occurs multiple times in a document and was already reliably linked b times to the same named entity in YAGO, the previous linking decisions (for that mention) are reused without rerunning the disambiguation process (i.e., in the experimental evaluation of BEL, the default value of b is 2). This heuristics may lead to incorrect linkings, but in empirical evaluations on real-world datasets the algorithm has shown a robust quality behavior, while being highly efficient (see Section 4).

The runtime of the algorithm is dominated by the computation of the contextual similarity scores of each ranking classifier. More specifically, since the parameters needed by each classifier are precomputed, each classifier needs only $O(N \log N)$ steps to propose a context-based ranking of the N candidates derived by the "prominence" score. Since the classifiers operate independently from each other,

the algorithm allows parallel computation of the contextual similarity scores. However, in this work we have implemented a sequential version, which for K different ranking classifiers has a complexity of $O(KN \log N)$.

3.2 Bagging of Ranking Classifiers for Majority Voting

The main idea behind BEL is to leverage different contextual representations of a mention. Each such representation is used by one ranking classifier to rank the candidate entities by their similarity to that representation. As shown in lines 14 and 15 of Algorithm 1, the mention is linked to the candidate that is ranked as top entity by the majority of the classifiers. This idea gives rise to several questions: (1) How to derive the different contextual representations of a mention? (2) How to compute the similarity between a contextual representation and a candidate entity? (3) How to prune the candidate space in such a way that only the most promising entities are considered by each of the ranking classifiers?

Obviously, the latter question is focused on efficiency; to address it, we exploit a precomputed index, constructed by exploiting the intra-Wikipedia links and the Wikipedia Redirect Pages. For every mention m, the index contains entities that might refer to it along with a probabilistic "prominence" score $P(e|m)$ by which a candidate entity e might refer to the mention. To compute this "prominence" score, we collect all terms that occur in a Wikipedia article or redirect-page and are hyperlinked to another Wikipedia article. From this collection we derive relative frequencies by which a Wikipedia entity (i.e., specific article) was hyperlinked from a given term. For example, Jordan is hyperlinked to the article about the country of Jordan 60% of the time, 20% of the time it is hyperlinked to the basketball player Michael Jordan, etc. These relative frequencies are estimations of the probability of an entity given a mention $P(e|m)$. Empirically (see also Figure 2a in the evaluation section) we have found out that when ranked by this score, the top-40 candidate entities already yield a overwhelming coverage rate of ≈92.4% for the correct entity. For top-100 this coverage rate increases only marginally (by only ≈0.5%). Thus considering only the top-40 entities in the candidate lists (which in general might contain hundreds or even thousands of entities), is not only an efficient but also an effective pruning strategy.

The second question involves the semantics of the similarity score. Suppose that $S_i(m)$ stands for the i'th contextual representation of the mention m. Our model is probabilistic in nature and holistic in the sense that the above "prominence" score $P(e|m)$ just falls off the model by following principled mathematical derivations. We start by reasoning about the probability of a candidate entity e given the mention m and its context $S_i(m)$:

$$P(e|S_i(m), m) = \frac{P(S_i(m)|m, e)P(m|e)P(e)}{P(S_i(m)|m)P(m)} \tag{1}$$

$$= \frac{P(e|m)P(S_i(m)|m, e)}{P(S_i(m)|m)} \tag{2}$$

$$\propto P(e|m)P(S_i(m)|m, e) \tag{3}$$

$$\propto logP(e|m) + logP(S_i(m)|m, e) \tag{4}$$

The last two steps in the above derivation mean that ranking the candidate entities by the similarity score $P(e|m)P(S_i(m)|m, e)$ or by $logP(e|m) + logP(S_i(m)|m, e)$ yields the same ranking as $P(e|S_i(m), m)$. Note that in general $S_i(m)$ depends on the entity e and not on the mention m. Hence, we can estimate $P(S_i(m)|m, e)$ as $P(S_i(m)|e)$. So the final similarity score is given by:

$$Sim(e, S_i(m), m) := logP(e|m) + logP(S_i(m)|e) \tag{5}$$

We estimate $P(S_i(m)|e)$ as the probability of $S_i(m)$ being generated by a language model (Zhai and Lafferty, 2004) on the terms describing e in Wikipedia. Those terms are collected from the Wikipedia article of e after removing stop words. Such a language model is described by means of frequency parameters $\boldsymbol{\theta}_e$. We construct it by indexing the terms and their frequencies in the corresponding Wikipedia articles. Figure 1 depicts the general idea behind our approach. For different contextual representations $S_1(m), ..., S_n(m)$ of a mention m, the ranking classifier responsible for $S_i(m)$ computes $Sim(e, S_i(m), m)$ for each candidate entity e and ranks the candidates by this score. Finally m is linked to the candidate that is ranked as the top entity by the majority of the ranking classifiers. This majority

voting strategy reduces the uncertainty of the linking process and leads to higher precision than a single ranking classifier, while still maintaining a high recall.

The final question concerns the computation of the contextual representations $S_i(m)$ of a mention m. We derive such representations by randomly sampling terms that occur in the local vicinity of a mention in the text. More specifically, to generate a contextual representation $S_i(m)$ from a range of N relevant terms around a mention m, we uniformly sample N times with replacement. We run the same procedure for all n representations. This sampling technique is known as bootstrapping (Breiman, 1996) and has been shown to have several advantages over other sampling procedures, such as increasing the contextual diversity and mitigating strong dependencies between features. Indeed, in the experiments, the bagging of the ranking classifiers lead to a significant improvement of $\approx$2.5% in terms of precision compared to the simple case where no bagging is used (see Section 4).

3.3 Recognizing Non-YAGO Entities

For an improved accuracy of the linking process, it is also crucial to reliably recognize true negatives, i.e., mentions that refer to entities that are not present in the underlying knowledge base. In case of the YAGO knowledge base, we first check whether the most prominent Wikipedia entity for a given mention is presented in YAGO; if this is not the case, the mention is classified as a non-YAGO entity. Note that many entities from Wikipedia are not present in YAGO, either due to recently added articles, or to articles that represent concepts[1]. Furthermore, a flexible threshold is used to recognize a non-YAGO entity. It is calculated as the maximum similarity score among the Wikpedia entities in the candidate list that are not present in the knowledge base, or as a default "prominence" score, when there is no such entity. If none of the candidates has a higher score than the threshold, the corresponding classifier proposes E_{NULL} as the best candidate. Also, in the simple case that the retrieved list of candidates is empty, the mention is classified as a non-YAGO entity. Although, these strategies are relatively straight-forward, they lead to a notable improvement in the recognition of true negatives. Further investigation of more elaborate strategies for the reliable detection of true negatives is part of our future work agenda.

3.4 Efficiency Aspects

For a better overview of the key efficiency aspects that are leveraged by BEL, we give here a succinct summary:

- Early pruning of the candidate space while maintaining a high coverage of promising candidates
- Local and independent reasoning strategy based on sliding windows and bootstrapping aggregation for the disambiguation process
- Highly efficient, in-memory processing of randomly sampled subsets
- Previous disambiguation decisions are exploited whenever possible; e.g., for people, locations, or company names that reoccur in a similar form in a document, the disambiguation process is run only once.

As it will be shown in the next section, the above considerations lead to a highly efficient linking process that often outperforms the evaluated state-of-the-art techniques, both in terms of quality and efficiency.

4 Experimental Evaluation

4.1 Datasets

Three datasets were used to evaluate the BEL approach. As a knowledge base for evaluation, we used YAGO2 (Hoffart et al., 2011a).

[1] In YAGO, the concepts have been derived from WordNet.

Table 1: Datasets overall information

	CoNLL-YAGO	CUCERZAN	KORE
articles	76	336	50
mentions (total)	1431	5343	148
mentions (non-YAGO)	279	936	7
word count (avg.)	173	384	12

CoNLL-YAGO: This dataset contains 76 randomly picked Reuters news articles of CoNLL 2003 data (Tjong Kim Sang and De Meulder, 2003). We have manually labeled the mentions, which are recognized by the Stanford NER Tagger (Finkel et al., 2005), to the corresponding entities in YAGO2.
CUCERZAN: This dataset consists of 350 Wikipedia articles that were randomly selected by S. Cucerzan to evaluate his approach (Cucerzan, 2007). The annotated entities in this corpus are named entities derived from the hyperlinks of mentions in these 350 Wikipedia articles. Since some of the articles are not available anymore in the Wikipedia archive, we have recovered 336 out of the 350 articles of the original corpus.
KORE: This small dataset was produced in the realm of AIDA (Hoffart et al., 2012). It is a synthetic corpus consisting of 50 very short articles, where each article contains one or more hand-crafted sentences about different ambiguous mentions of named entities. This dataset is quite difficult, as the named entities in this corpus are ambiguous with sparse context.

4.2 Evaluated Approaches

We compared BEL to three other prominent approaches (Hoffart et al., 2011b; Hoffart et al., 2012; Cucerzan, 2007), which, as reported in the corresponding papers, outperform many state-of-the-art algorithms in terms of disambiguation and linking quality. Experience-wise, we can confirm that the very recent AIDA approaches (Hoffart et al., 2011b; Hoffart et al., 2012) have indeed raised the bar for many entity linking methods. In our experiments, these algorithms showed a highly reliable behavior, even with respect to difficult disambiguation tasks.

The AIDA approach comes in different versions: In its original version (Hoffart et al., 2011b), it exploits a graph-based connectivity between candidate entities of multiple mentions (i.e., graph coherence, e.g., derived from the *type, subclassOf* edges of the knowledge graph or from the incoming links in Wikipedia articles) to determine the most promising linking of the mentions. We refer to this version of AIDA as AIDA-GRAPH. In another version that has been optimized for datasets such as KORE (Hoffart et al., 2012), AIDA's coherence model has been extended to recognize key-phrases for named entities, which are then used to determine a similarity score based on key-phrase overlap between candidate entities. We refer to this version as AIDA-KORE.

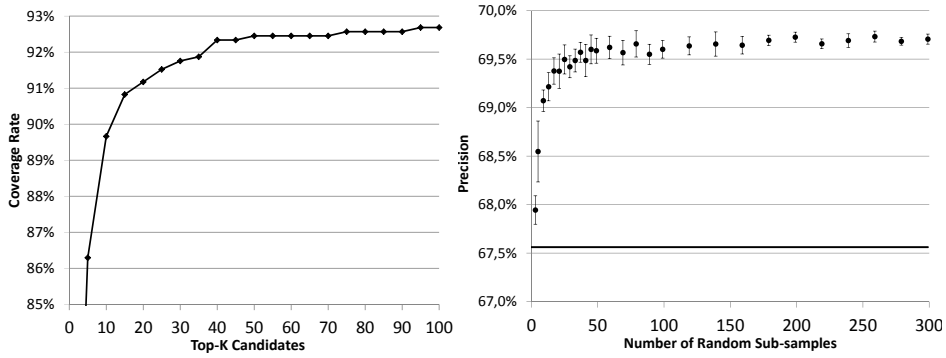
Cucerzan (2007) finds a linking of mentions to Wikipedia entities, such that the sum of vector-based similarities between the candidate entities and the document (containing the mentions) as well as the similarities between pairs of candidate entities is maximized. We refer to this method as LED (Large-scale Entity Disambiguation). The original work has been conducted at Microsoft and the code is proprietary. Hence, we had to re-implement the algorithm according to the descriptions in the paper. To make sure that algorithm was correctly implemented, we evaluated it on the original dataset, and achieved results comparable to those presented in the original paper. Note that, since many entities from Wikipedia are not present in YAGO, the task of linking mentions of the CUCERZAN dataset to YAGO is different from the original task addressed in (Cucerzan, 2007), where mentions were linked to Wikipedia articles.

4.3 Parameter Analysis for BEL

For BEL, the parameters are optimized to deal with common natural-language articles on the Web (e.g., articles from encyclopedic pages or news sites). The same parameter settings are used on all three datasets described above to show the performance of BEL on different types of corpora. To achieve such a common setting of the parameters, we trained BEL on articles sampled from the above datasets, each of which exhibits specific textual characteristics.

4.3.1 Pruning Candidate Lists

In BEL, each mention is assigned a list of candidates. In general, such a list could contain hundreds or even thousands of entities. However, the mention should be linked to at most one entity in the list.



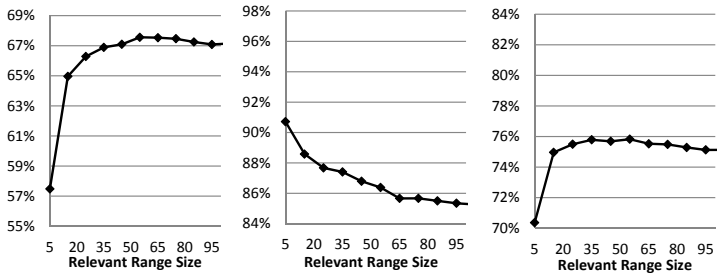
(a) (b)

Figure 2: Parameter analysis experiments (in %): (a) Correct Entity Coverage Rate. (b) Performance of bagging strategy in precision in comparison to the performance of a single classifier.

We randomly picked 1000 mentions from the three datasets to analyze the impact of the candidate list size. The coverage rate (i.e., the relative frequency by which the correct entity is contained in the list) in relation to the list size is shown in Figure 2a. The lists are sorted by decreasing "prominence" scores (see Section 3). In this experiment, 139 mentions have no corresponding entity in YAGO, while 61 correct entities are missing, which means that the maximum coverage rate that a candidate selection strategy can achieve is $800/861 \approx 92.92\%$. As the curve shows, most of the correct entities are indeed located within the top positions of the candidate lists. Therefore, we prune the ranked lists by selecting the top-40 candidates for further processing.

4.3.2 Range of Relevant Terms

As mentioned earlier, the bagging of classifiers is aimed at capturing the contextual information of a mention by randomly sampling terms surrounding it, a process that is repeated several times, once for every ranking classifier. As a sampling procedure we employ bootstrapping (Breiman, 1996), which captures the diversity of contextual information derived from the original range, while mitigating dependencies between terms. We analyze the quality of this bagging strategy mainly based on two criteria: (1) the size of the range of relevant terms, and (2) the bagging size (i.e., number of randomly sampled subsets).



(a) Precision (b) Recall (c) F1-measure

Figure 3: Performance of the language model by increasing the size of the range of relevant terms.

The impact of the range size on the linking quality is intricate, in the sense that a larger range contains more noise, while a smaller one has sparser context. In BEL, the range of relevant terms is empirically calibrated, by evaluating the performance of BEL on different range sizes after removing stop words and non-English terms. To avoid any bias from the "prominence" score and to focus only on the context, we set the $logP(e|m)$-component of the scoring function to 0. Figure 3 shows the average performance

based on 10-fold cross validation on all evaluation datasets. The F1-measure achieves maximum when the range of relevant terms contains 55 terms, which is also in agreement with the optimal setting found by Gooi and Allan (2004). Thus, we define the size of the range to be 55. When a document contains less terms, BEL takes the whole text into account.

The bagging strategy of BEL encourages the contextual diversity, while it reduces the linking uncertainty by employing majority voting. Here, to show the impact of our bagging strategy, we randomly pick 80% articles from our datasets for different bagging sizes. Since the bagging strategy is mainly affected by contextual information, we turn off the component responsible for the "prominence" score and run BEL 10 times for each bagging size on these articles. In Figure 2b, the black horizontal line with precision 67.56% is the baseline derived from the single language model classifier. The black dots denote the average precisions and the error bars show the corresponding standard deviation. As the figure shows, by increasing the bagging size the precision increases, while the standard deviation decreases. Considering the precision, efficiency, and stability of the algorithm, we use 199 subsets as the default setting (an odd number of voters is more likely to avoid ties when there are two top-ranked candidates by the voters). Note that, the linking process is stricter and thus leads to a decreased recall. However, the experimental result shows that the impact of the bagging strategy on the increase of precision is consistently higher than its impact on the decrease of the F1-measure; the precision increases from 67.56% to 69.73%, while the F1-measure decreases from 75.82% to 75.32%. Moreover, in our opinion, it is better to suggest that a mention is not in the knowledge base than link it to a wrong entity.

BEL has been evaluated with respect to its linking quality and efficiency. The employed evaluation measures and the results are presented in the following subsections.

4.3.3 Evaluation Measures

For the quality evaluation, we have measured precision, recall, and the F1-measure of each of the above approaches on the mentioned datasets.

A *true positive (tp)* is a mention that has been correctly linked to a YAGO entity. An incorrect linking is defined to be a *false positive (fp)*. Furthermore, a *true negative (tn)* refers to a mention that is correctly identified as an entity that does not occur in YAGO (i.e., non-YAGO entity). The remaining cases are defined as *false negatives (fn)*. Precision is then defined as $P = tp/(tp + fp)$ and recall as $R = tp/(tp + fn)$. The F1-measure is obtained from the harmonic mean of precision and recall as $F = 2PR/(P+R)$.

For the efficiency evaluation, we have measured the runtime (in seconds) of each approach on each dataset.

4.3.4 Evaluation of Linking Quality

The results of the quality evaluation are shown in Table 2, along with the corresponding confidence intervals, which are calculated by repeating 30 times a random sampling of subsets containing 60% of the documents from each dataset. For each dataset, the results computed on all documents are within the intervals that correspond to a confidence level of 99% according to the Student's t-distribution to show that although some of the datasets are of moderate size, the 99% confidence interval of the scores computed on the sampled subsets is relatively small.

As it can be seen, BEL significantly outperforms all the other approaches on the CoNLL-YAGO dataset, especially on precision. Also, for the CUCERZAN dataset, the quality of BEL is comparable to that of AIDA-GRAPH and AIDA-KORE, and it significantly outperforms LED. Moreover, in terms of precision, BEL performs also on this latter dataset significantly better than the other approaches (i.e., from a statistical point of view). Together with BEL's impressive efficiency (see Section 4.3.5), the precision-related quality is a crucial scalability aspect, since when processing a high throughput of documents it is highly important that the produced linkings be rather correct.

For the KORE dataset, AIDA-KORE outperforms other approaches. However, it should be noted that KORE is a very challenging dataset and that the AIDA-KORE approach has been specifically tailored to such datasets. Also note that although the AIDA-KORE algorithm shows a high linking quality in the experiments, it is the least efficient approach, since it performs complex joint reasoning over groups of candidate entities and mentions. In our experiments, we had to wait more than 15 hours for the

evaluation results of this approach for KORE dataset, since the mentions contained in this corpus are highly ambiguous.

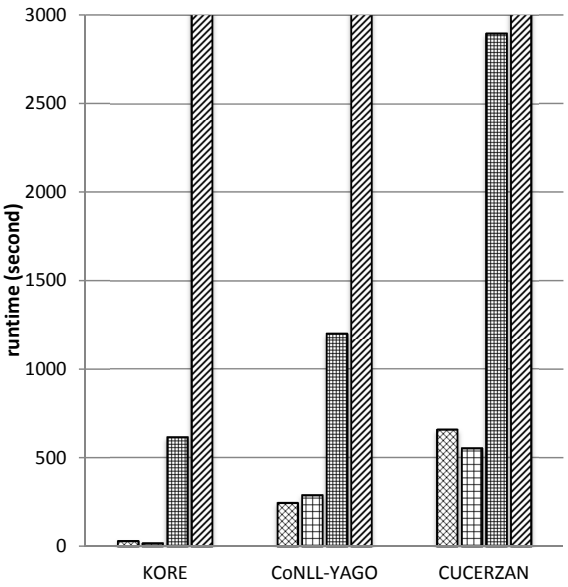
In comparison to a greedy linking strategy, where a mention is simply linked to the most prominent entity according to the "prominence" score, which is our baseline BEL-PROM, BEL performs much better on all three datasets. This fact highlights the importance of the contextual similarity component in the model.

Table 2: Evaluation results (in %).

	Method	Precision	Recall	F1
CoNLL-YAGO	LED	62.35 (-1.92,+0.25)	96.13 (-0.43,+0.27)	75.63 (-1.50,+0.18)
	AIDA-GRAPH	78.67 (-0.80,+1.23)	96.29 (-0.20,+0.64)	86.59 (-0.41,+0.82)
	AIDA-KORE	77.11 (-0.86,+0.80)	96.21 (-0.64,+0.25)	85.61 (-0.67,+0.47)
	BEL-PROM	68.37 (-0.89,+1.30)	97.40 (-0.25,+0.32)	80.30 (-0.61,+0.97)
	BEL	**81.40** (-1.33,+0.78)	95.72 (-0.38,+0.25)	**87.98** (-0.85,+0.46)
CUCERZAN	LED	63.47 (-0.40,+1.01)	96.94 (-0.11,+0.24)	76.72 (-0.28,+0.75)
	AIDA-GRAPH	81.30 (-0.57,+0.16)	94.64 (-0.28,+0.17)	87.47 (-0.40,+0.11)
	AIDA-KORE	81.35 (-0.83,+0.03)	97.31 (-0.25,+0.10)	**88.61** (-0.57,+0.03)
	BEL-PROM	73.92 (-0.53,+0.29)	98.83 (-0.11,+0.06)	84.58 (-0.37,+0.20)
	BEL	**82.37** (-0.31,+0.25)	93.46 (-0.71,+0.27)	87.56 (-0.35,+0.12)
KORE	LED	40.14 (-3.30,+0.88)	100.00 (-0.00,+0.00)	57.28 (-3.52,+0.79)
	AIDA-GRAPH	62.33 (-1.83,+0.93)	100.00 (-0.00,+0.00)	76.79 (-1.43,+0.68)
	AIDA-KORE	**66.67** (-2.29,+1.91)	94.95 (-0.87,+1.82)	**78.33** (-1.60,+1.48)
	BEL-PROM	31.29 (-1.83,+1.47)	100.00 (-0.00,+0.00)	47.67 (-2.23,+1.61)
	BEL	54.55 (-2.40,+2.53)	76.61 (-0.76,+2.20)	63.72 (-1.72,+2.08)

Table 3: Efficiency comparison

Method	KORE	CoNLL-YAGO	CUCERZAN
BEL	30.02s	244.34s	657.44s
LED	17.70s	288.52s	552.26s
AIDA-GRAPH	615.35s	1,202.66s	2,897.43s
AIDA-KORE	>15h	>11h	>25h



4.3.5 Efficiency Evaluation

The lean algorithmic design of BEL enables a highly efficient linking process. For the efficiency comparison, we used a Pentium 3.1GH machine with 8GB of main memory. The indexes for the language models of YAGO2 entities, as well as the indexed YAGO2 knowledge base (that was used by all approaches for the linking) were maintained in a PostgreSQL 9.1 database.

For each dataset, Table 3 shows the runtime of each approach. Obviously, the joint reasoning strategy of AIDA-GRAPH comes at high efficiency costs; on all datasets it has been outperformed by the other approaches. While LED is slightly more efficient than BEL on the KORE and CUCERZAN datasets, as shown in Table 2, it often pays a high cost in terms of quality, and BEL also outperforms it in terms of efficiency on the CoNLL-YAGO dataset. Note that the runtime of LED and BEL are both practically viable from a user's perspective. The AIDA-KORE approach, on the other hand, lacks practical viability, since it needs several hours to process even moderately sized datasets (e.g., approx. 11 hours for the CoNLL-YAGO dataset). For the CUCERZAN dataset, which is the largest one, although the F1-measure of AIDA-KORE is approximately 1% higher than BEL, one needs to wait more than 25 hours to get the result. Instead, BEL can finish the linking process in around 11 minutes.

4.3.6 Discussion

Both, the "prominence" score and the contextual score derived from the proposed bagging strategy have advantages and limitations; they are orthogonal in nature, and their individual strengths are manifested in different ways in the final decision of the algorithm. The value of the "prominence" score has a high impact on the final decision, when BEL is run on articles about famous people, organizations, locations, products, events etc. Typical examples of articles that contain such entities are news reports,

scholarly articles containing encyclopedic knowledge, and product descriptions. In contrast, the bagged language models have a high impact on the final decision in cases where the occurring mentions are highly ambiguous but contain valid key information surrounding the mention. Examples for such articles can be found in all three datasets we have used.

Although in many cases, linking the mention to the most prominent candidate entity leads to the correct decision (e.g., "Australia" refers most probably to the country), this strategy is not reliable for many ambiguously used mentions. For example, in one article of CoNLL-YAGO dataset, the named entity "Australia National Cricket Team" in YAGO, was also often referred to by "Australia". Nevertheless, BEL was able to establish a linking to the correct named entity.

The datasets we have annotated and a preliminary online-demo of the algorithm are available online[2].

5 Conclusion

The focus of this work has been on lean and light-weight classification algorithms, which as an ensemble provide a reliable and efficient linking strategy. The comparison of our approach, BEL, with state-of-the-art techniques on manually-labeled, benchmark datasets shows that BEL indeed fulfills the above criteria. Especially on longer, real-world texts, BEL shows an unprecedented quality and efficiency behavior. Further research is needed to understand how such an approach can be optimized for short texts containing highly ambiguous mentions of named entities. We are convinced that BEL provides a robust basis for further research in this area.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. *DBpedia: A nucleus for a web of open data*. The Semantic Web. Springer.

David Aumueller, Hong-Hai Do, Sabine Massmann, and Erhard Rahm. 2005. Schema and ontology matching with COMA++. In *Proceedings of the International Conference on Management of Data*, pages 906–908. ACM.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the annual meeting of the Association for Computational Linguistics*, pages 79–85.

Indrajit Bhattacharya and Lise Getoor. 2007. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data*, 1(1):5.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the International Conference on Management of Data*, pages 1247–1250.

Leo Breiman. 1996. Bagging predictors. *Machine learning*, 24(2):123–140.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, pages 9–16.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 708–716.

Fang Du, Yueguo Chen, and Xiaoyong Du. 2013. Linking entities in unstructured texts with RDF knowledge bases. In *Web Technologies and Applications*, pages 240–251. Springer.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2009. Scaling Wikipedia-based named entity disambiguation to arbitrary web text. In *IJCAI Workshop: User Contributed Knowledge and Artificial Intelligence: An Evolving Synergy*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the annual meeting of the Association for Computational Linguistics*, pages 363–370.

[2] `http://hpi-web.de/naumann/projekte/bel.html`

Michael Fleischman and Eduard Hovy. 2004. Multi-document person name resolution. In *Proceedings of the Workshop on Reference Resolution and its Applications*.

Chung Heong Gooi and James Allan. 2004. Cross-document coreference on a large scale corpus. In *Proceedings of the Conference on Natural Language Learning at HLT-NAACL*, pages 9–16.

Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, Edwin Lewis-Kelham, Gerard de Melo, and Gerhard Weikum. 2011a. YAGO2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the International World Wide Web Conference*, pages 229–232.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011b. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792.

Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2012. KORE: keyphrase overlap relatedness for entity disambiguation. In *Proceedings of the International Conference on Information and Knowledge Management*, pages 545–554.

Lan Huang, David Milne, Eibe Frank, and Ian H. Witten. 2012. Learning a concept-based document similarity measure. *Journal of the American Society for Information Science and Technology*, 63(8):1593–1608.

Simon Lacoste-Julien, Konstantina Palla, Alex Davies, Gjergji Kasneci, Thore Graepel, and Zoubin Ghahramani. 2013. Sigma: Simple greedy matching for aligning large knowledge bases. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 572–580. ACM.

Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and searching web tables using entities, types and relationships. *Proceedings of the VLDB Endowment*, 3(1-2):1338–1347, September.

Thomas Lin, Mausam, and Oren Etzioni. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 84–88.

Gideon S. Mann and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Proceedings of the Conference on Natural Language Learning at HLT-NAACL*, volume 4, pages 33–40.

David Milne and Ian H. Witten. 2008. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy*, pages 25–30. AAAI Press.

Felix Naumann and Melanie Herschel. 2010. *An Introduction to Duplicate Detection*. Morgan and Claypool Publishers.

Ted Pedersen, Amruta Purandare, and Anagha Kulkarni. 2005. Name discrimination by clustering similar contexts. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics*, volume 3406 of *Lecture Notes in Computer Science*, pages 226–237. Springer.

Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2012. LINDEN: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the International World Wide Web Conference*, pages 449–458.

Avirup Sil, Ernest Cronin, Penghai Nie, Yinfei Yang, Ana-Maria Popescu, and Alexander Yates. 2012. Linking named entities to any database. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 116–127.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the International World Wide Web Conference*, pages 697–706. ACM.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In *Proceedings of the Conference on Natural Language Learning at HLT-NAACL*, volume 4, pages 142–147.

Melanie Weis and Felix Naumann. 2005. DogmatiX tracks down duplicates in XML. In *Proceedings of the International Conference on Management of Data*, pages 431–442, Baltimore, MD.

Fei Wu and Daniel S. Weld. 2008. Automatically refining the wikipedia infobox ontology. In *Proceedings of the International World Wide Web Conference*, pages 635–644.

Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, 22(2):179–214, April.

Exploratory Relation Extraction in Large Text Corpora

Alan Akbik **Thilo Michael** **Christoph Boden**
Database Systems and Information Management Group
Einsteinufer 17, 10587 Berlin, Germany
{firstname.lastname}@tu-berlin.de

Abstract

In this paper, we propose and demonstrate *Exploratory Relation Extraction* (ERE), a novel approach to identifying and extracting relations from large text corpora based on user-driven and data-guided incremental exploration. We draw upon ideas from the information seeking paradigm of Exploratory Search (ES) to enable an exploration process in which users begin with a vaguely defined information need and progressively sharpen their definition of extraction tasks as they identify relations of interest in the underlying data. This process extends the application of Relation Extraction to use cases characterized by imprecise information needs and uncertainty regarding the information content of available data.

We present an interactive workflow that allows users to build extractors based on entity types and human-readable extraction patterns derived from subtrees in dependency trees. In order to evaluate the viability of our approach on large text corpora, we conduct experiments on a dataset of over 160 million sentences with mentions of over 6 million FREEBASE entities extracted from the CLUEWEB09 corpus. Our experiments indicate that even non-expert users can intuitively use our approach to identify relations and create high precision extractors with minimal effort.

1 Introduction

1.1 Motivation and Problem Statement

Relation Extraction (RE) is the task of creating extractors that automatically find instances of semantic relations in unstructured data such as natural language text (Riloff, 1996). An example extraction task might be to find instances of the EDUCATEDAT relation, which relates persons to their educational institution and may include the entity pair <*Sigmund Freud, University of Vienna*> as relation instance. Motivated by an explosion of readily available sources of text data such as the Web, RE offers intriguing possibilities for querying and analyzing data as well as extracting and organizing the contained information (Sarawagi, 2008). As scalable computing architectures capable of processing ever larger amounts of data are being developed (Dean and Ghemawat, 2004) and dependency parsers are becoming more accurate and more robust (Petrov and McDonald, 2012), so rises the potential of developing means to directly access the structured information contained in natural language text.

In spite of such positive trends however, currently established methods of creating relation extractors suffer from a number of limitations. The first is one of *cost*; the process of creating extractors requires either labeled data to be produced at sufficient quality and quantity in order to train a supervised machine learning algorithm (Culotta and Sorensen, 2004; Mintz et al., 2009), or the manual creation of a complex set of extraction rules (Strötgen and Gertz, 2010; Reiss et al., 2008). In either case, the process is tedious and time-consuming and requires trained specialists with an extensive background in NLP, rule-writing or machine learning (Chiticariu et al., 2013). Worse, this process needs to be repeated for every relation and domain of interest. Due to this cost, great care must be taken when deciding which relation types to look for in a given text corpus.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2087–2096, Dublin, Ireland, August 23-29 2014.

This leads to the second limitation, namely the necessary *a priori specification* of relations. Current methods generally require a careful upfront definition of the RE tasks in order to start producing labeled training data or extraction rule-sets. Practical scenarios, however, are often characterized by imprecise and rapidly changing information needs and uncertainty regarding the type of information contained in large, given text corpora (Chiticariu et al., 2013). This severely limits the practicability of currently established RE methods.

1.2 Exploratory Search for Relations

To address these limitations, we propose a process of *exploration* for relations of interest in available data. We propose to substantially reduce entry barriers into RE so that extraction tasks no longer need to be exactly pre-specified and expensively prepared by generating labeled training data in advance. Instead, we propose a manual, rule-based approach in which extraction rules are kept very simple so that users can formulate natural language-like patterns as exploratory queries for relations against a text corpus.

We draw inspiration from the information seeking paradigm of *Exploratory Search (ES)* (Marchionini, 2006; White and Roth, 2009), where users start with a vaguely defined information need and - with a mix of look-up, browsing, analysis and exploration - progressively discover information available to address it and simultaneously concretize their information need. One of the challenges associated with the often desired capability of ES is the design of interactive interfaces to support users as they navigate through complex environments. Similarly, our challenge is to create an intuitive workflow that allows non-experts in NLP to engage in relation exploration.

We propose to simplify the search for information by using natural language-like queries that match subtrees in large corpora of dependency parsed data while hiding the complexity from the users. Explorative queries return matching relation instances and source sentences, as well as suggestions for further queries computed from the available data. By following a process of experimental querying and accepting or rejecting pattern suggestions, users identify relations of interest and group patterns into extractors. Our goal is to make use of such data-guidance to facilitate exploration while giving as much explicit control to a user as possible.

1.3 Contributions

In this paper, we propose and demonstrate *Exploratory Relation Extraction (ERE)*, a user-driven and data-guided incremental exploration approach to Relation Extraction. We give details on our relation extraction pattern language and introduce a guided, interactive workflow aimed at allowing users to explore parsed text corpora for relations at minimal effort. We conduct two experiments on a large corpus of over 160 million sentences from the CLUEWEB09 to determine in how far non-experts can use ERE to discover and extract relations. We discuss the results of the user study, as well as strengths and weaknesses of our proposed approach.

2 Exploratory Relation Extraction

In this section, we present our approach for Exploratory Relation Extraction. We provide details on how we define extraction patterns and how we preemptively extract all subtrees in dependency trees from a given text corpus (Section 2.1). We then outline a data-guided incremental workflow to explore the indexed data for relations (Section 2.2) and illustrate this with an exemplary execution (Section 2.3).

2.1 Human-Readable Relation Extraction Patterns

Like much previous work in RE (Culotta and Sorensen, 2004; Schutz and Buitelaar, 2005; Uszkoreit, 2011), we define extraction patterns using features from dependency-parsed sentences. As recent work has shown (Del Corro and Gemulla, 2013; Akbik et al., 2013b), patterns in dependency trees are well-suited to manual rule based RE, as they enable more succinct and thus more human-readable rule sets. Following this work, we define RE patterns as subtrees in dependency trees.

In our work, we follow the idea of Preemptive Information Extraction (Shinyama and Sekine, 2006) in which all possible relations for a given text corpus are preemptively generated in advance. Applied

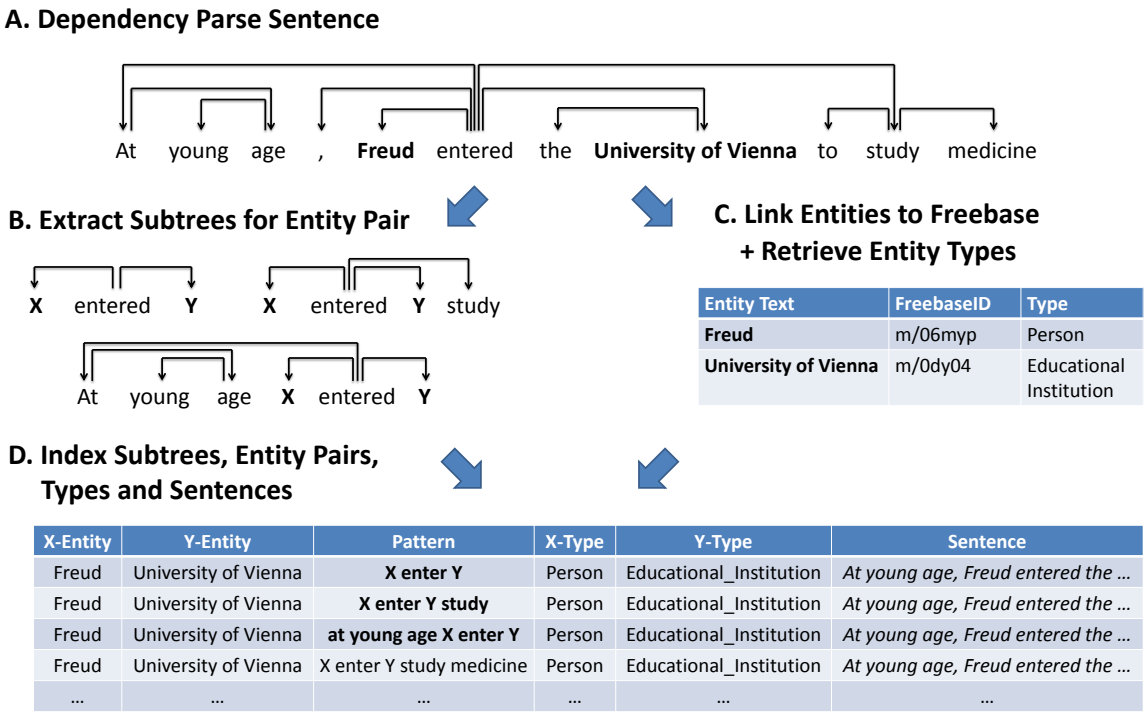


Figure 1: Illustration of the subtree generation process. We parse each sentence in a given document collection using a dependency parser and annotate all entities (**A**). Then, we generate all possible subtrees in the dependency tree that span pairs of annotated entities, three of which are illustrated in **(B)**, and link entities to their FREEBASE IDs to determine their entity types (**C**). We then generate a lexical, lemmatized representation of these subtrees which we store along with the entity pair, their entity types and sentence they are observed with **(D)**.

to our problem this means that we generate all possible dependency subtrees, arguing that depending on the user's information need, any such pattern may be valuable. Since we are interested in binary relations only, we generate only those subtrees that span two named entities in a sentence. In addition, we also determine the fine-grained entity types for named entities in order to allow users to optionally restrict patterns to match only entities of certain types. Previous work has shown the benefit of including fine-grained type restrictions into patterns (Akbik et al., 2013a).

We illustrate this process with an example sentence in Figure 1, for which we determine all subtrees that span the indicated entity pair. In the subtrees, we replace the entity tokens with the placeholders "*X*" and "*Y*", where the former is the placeholder for the X-entity and the latter the placeholder for the Y-entity. For better human-readability, we lexicalize the patterns by lemmatizing the words and discarding information on typed dependencies. We also link the entities in the sentence to entries in the FREEBASE knowledge base (Bollacker et al., 2008), allowing us to retrieve their fine grained entity types.

We then index the information on lexicalized patterns, the entities they span and their types, as well as the sentences in which the patterns were found (Figure 1D). This allows users to query for any combinations of patterns and entity type restrictions and retrieve matching entity pairs and sentences from the index. For instance, a user may query for all entity pairs that match the "*at young age X enter Y*" pattern, and optionally restrict the Y-entity to be only of type ORGANIZATION, or more specific types such as CHURCH or UNIVERSITY. We argue that because patterns are lexicalized variants of dependency subtrees and entity type restrictions can have human readable names, such queries are intuitive to users even without an NLP background. The use and preemptive indexing of human-readable patterns decreases the entry barriers into the ERE process, as this enables users to exploratively query parsed text corpora.

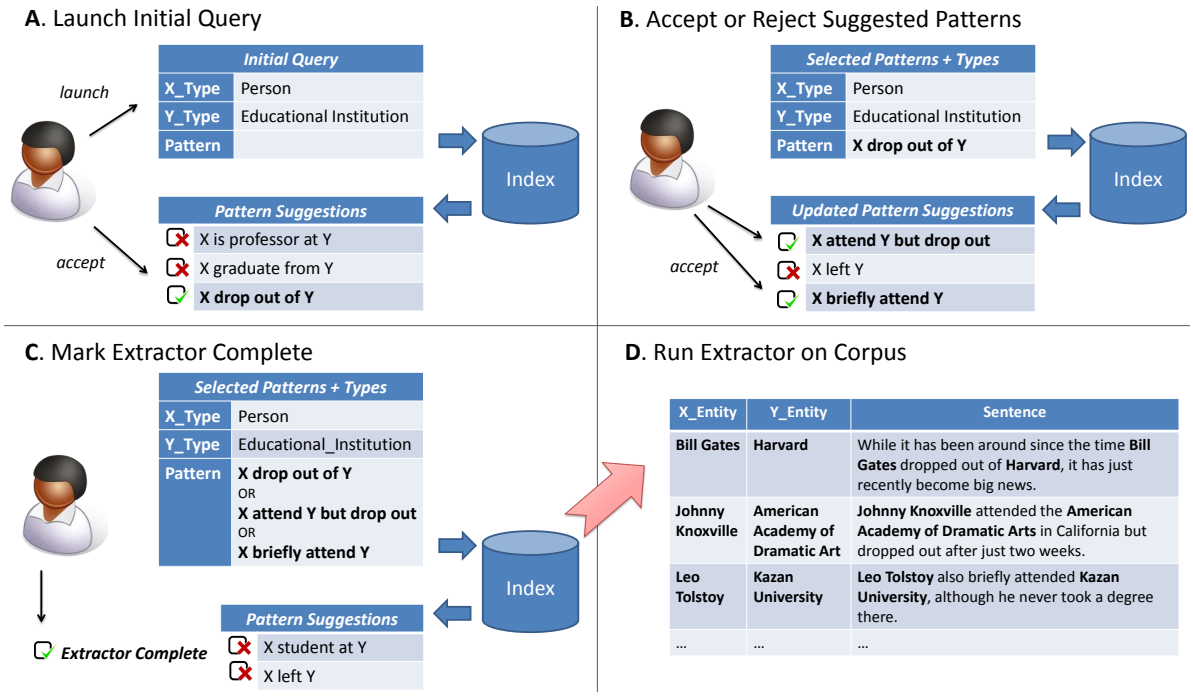


Figure 2: Illustration of the exploratory relation extraction process. The user begins with specifying entity types of interest and receives a set of pattern suggestions (**A**). Intrigued by the pattern "X drop out from Y", the user affirms this pattern. This prompts updated pattern suggestions which the user affirms or rejects (**B**). When no more interesting patters are offered, the user marks the extractor as complete (**C**) and runs it on a corpus, retrieving relation instances and matching sentences (**D**).

2.2 Guidance From Available Data

A second key component is to provide guidance in the exploration process by computing suggestions for patterns from user input and enabling an interactive workflow that allows users to work with available data. Such guidance is needed for two reasons: First, though much effort is invested in human-readable extraction patterns, users may need support in formulating patterns and choosing entity type restrictions. This is especially the case when users are non-experts in the domain of interest and they strive to identify a range of appropriate patterns. Second, users may be uncertain of the information content of a given text corpus. By providing guidance through automatic pattern suggestions that reflect available information, we help users find patterns for their information need.

Users formulate an *entry point* to launch the exploration process, either by providing entity types, patterns or both. We guide the formulation of this initial query through autocomplete options. If the user enters only types for the entities, the system offers the most common patterns that are observed between entities of these types. The user can also search for patterns that contain a certain keyword.

In either case, the system suggests patterns that meet the user-defined entry point. Patterns are ordered by their absolute count in the corpus so that more common patterns are displayed at the top of the list. In addition, verb-based patterns are favored using a scoring metric that assigns extra points to patterns that include verbs. To assist a user in understanding a pattern, we optionally display example sentences and entity pairs in which it matches.

The user then starts a process of selecting (and de-selecting) entity type restrictions and pattern, thus refining the extractor while being guided by constantly updated pattern suggestions. The user continues this process until satisfied with the created extractor at which point it can be saved and the discovered relation instances downloaded. The user can now repeat the workflow to create more extractors.

2.3 Exploration Workflow Example

Suppose we have a user who is given a large text corpus and is asked to link persons to their respective educational institutions, but is unsure of what type of relevant information may be found in the corpus. Knowing only that relations should hold between entities of type PERSON and entities of type EDUCATIONAL_INSTITUTION, the user starts an exploration process by providing only these entity type restrictions. This is illustrated in Figure 2A).

A query is run against the index that identifies common patterns that hold between entities of such types, including "*X be professor at Y*", "*X study at Y*" and "*X drop out from Y*". Recall that each pattern is a human-readable version of a subtree in a dependency tree with two placeholders for entities, namely "X" and "Y". These placeholders may match named entities of any type, or can be restricted to matching only entities of certain types such as persons, organizations or locations. By clicking on a pattern, the user retrieves entity pairs and sentences in which a pattern matches; For example, the user is informed that the pattern "*X study at Y*" finds the relation instance <*Bill Gates, Harvard University*> in the sentence "***Bill Gates** briefly studied at **Harvard University***.".

Intrigued by the pattern "*X drop out from Y*", the user affirms this pattern and rejects all other suggestions. This causes a new query to be run against the parsed data, this time consisting of the entity restrictions as well as the pattern. As the query is now more concrete, the pattern suggestions are updated to reflect this new information. The user is presented with similar patterns such as "*Y dropout X*" and "*X attend Y but drop out*". This is illustrated in Figure 2B).

The user repeats this, selecting or de-selecting patterns (Figure 2B). At each interaction, suggestions are updated to reflect the current selection. When the user is satisfied with the identified relation, the selected set of patterns and restrictions is saved as an extractor (Figure 2C) and executed against the entire text corpus (Figure 2D). This returns lists of matching relation instances and sentences. The user has thus started with an imprecise information need and identified a relation of interest in a given text corpus, namely a relation for persons that attended an educational institution but did not graduate.

3 Experiments

In order to examine in how far our approach indeed contributes to overcoming the limitations of RE outlined in Section 1.1, namely the significant cost and the necessary a-priory specification of relations, we conduct a user study with 10 subjects that have little or no NLP background. We ask the users to apply the workflow for two separate tasks: An *extraction task* in which users are given four clearly defined semantic relations and an *exploration task* in which users are asked to identify relations for more vaguely defined information needs. We only provide the users with a brief introduction into the workflow. For the extraction task, we measure the time spent per extractor and estimate the quality of the created extractors in terms of precision and recall. For the exploration as well as for the extraction task we also qualitatively inspect discovered relations and evaluate user feedback.

3.1 Datasets

ClueWeb09. As source of text data, we use the English language portion of the well-known CLUEWEB09[1] reference corpus, consisting of roughly 5 billion crawled Web pages. We use boilerplating to remove HTML markup and sentence splitting to determine English language sentences.

FACC1. We use the recently released FACC1 (Gabrilovich et al., 2013) resource, a high quality named entity linking effort that was executed on the CLUEWEB09 corpus, linking over 6 billion entity mentions to their corresponding FREEBASE entries. Using this data, we identify over 160 million sentences in CLUEWEB09 that contain at least two entities we can link to FREEBASE. We parse all such sentences using the ClearNLP toolkit (Choi and McCallum, 2013).

Gold Standard Relation Annotations. As gold standard, we use the FREEBASE relation annotations as well as annotations from the "*Relation Extraction Corpus*"[2] a large, human-judged dataset of five relations about public figures on Wikipedia that was released by Google. Four of these relations involve

[1] http://lemurproject.org/clueweb09/

[2] http://code.google.com/p/relation-extraction-corpus/

	EDUCATEDAT					GRADUATEDWITHDEGREE				
	#INST	P	R	#PAT	TIME	#INST	P	R	#PAT	TIME
USER 1	**58,611**	0.99	0.2	**51**	12 min	17,698	1.0	0.27	34	17 min
USER 2	48,782	0.99	**0.31**	34	15 min	12,180	1.0	0.27	27	14 min
USER 3	25,435	0.88	0.12	12	8 min	**54,371**	0.93	0.53	24	8 min
USER 4	33,095	0.99	0.23	25	12 min	7,196	1.0	0.22	9	10 min
USER 5	47,668	0.76	0.16	29	13 min	34,942	1.0	0.48	3	5 min
USER 6	20,356	0.99	0.15	18	14 min	10,290	1.0	0.25	12	14 min
USER 7	22,889	0.62	0.01	8	4 min	37,119	0.71	0.6	19	4 min
USER 8	31,412	0.98	0.19	13	15 min	1,251	0.46	0.04	10	14 min
USER 9	14,169	0.99	0.1	6	8 min	13,104	0.6	0.17	13	12 min
USER 10	29,289	0.99	0.19	17	15 min	35	1.0	0.02	4	20 min
AVERAGE	33,171	0.92	0.17	21	11.6 min	18,819	0.87	0.29	16	11.8 min

	BORNIN					DIEDIN				
	#INST	P	R	#PAT	TIME	#INST	P	R	#PAT	TIME
USER 1	**158,222**	0.7	0.26	18	9 min	**25,779**	0.7	0.14	32	9 min
USER 2	72,888	0.79	0.21	23	17 min	13,582	0.86	0.13	12	12 min
USER 3	89,825	0.84	0.22	21	7 min	15,849	0.86	0.13	12	7 min
USER 4	66,899	0.81	0.21	19	14 min	13,542	0.86	0.13	11	8 min
USER 5	65,213	0.82	0.19	19	15 min	21,105	0.85	0.13	10	9 min
USER 6	131,275	0.83	0.25	16	13 min	14,423	0.85	0.13	8	9 min
USER 7	7,851	0.85	0.03	5	4 min	15,980	0.85	0.14	17	4 min
USER 8	52,927	0.82	0.17	10	15 min	25,090	0.74	0.14	8	14 min
USER 9	56,724	0.84	0.18	10	12 min	15,728	0.85	0.14	8	9 min
USER 10	58,347	0.94	0.22	10	15 min	14,112	0.86	0.13	8	10 min
AVERAGE	76,017	0.82	0.19	15	12.1 min	33,171	0.82	0.13	13	9.1 min

Table 1: Evaluation results for the 4 well-defined relations in the extraction task. We note differences from user to user, especially with regards to the number of found instances (**#INST**), the number of selected patterns (**#PAT**) and the time spent per relation. Extractors generally find large amounts of relation instances at high precision (**P**), while recall values (**R**) are lower. Users are ordered by the total number of patterns they selected. User 1 selected the most patterns overall and found the most instances for the BORNIN, DIEDIN and EDUCATEDAT relations (highlighted bold). User 10 both spent the most time overall while selecting the fewest patterns. User 7 spent the least amount of time overall.

FREEBASE entities, namely BORNIN, DIEDIN, EDUCATEDAT and GRADUATEDWITHDEGREE. We use these relations in the extraction task.

3.2 Extraction Task

We evaluated the user-created extractors against the gold standard annotations. However, even with relatively large sources of annotations, only roughly 5% of entity pairs in our 160 million sentences have a known FREEBASE relation. We therefore compute precision and recall only for labeled entity pairs, and separately list the absolute number of extracted relation instances.

Large amounts of relation instances at high precision. As Table 1 indicates, many users were able to create extractors that find very large amounts of instances (over 100.000 instances in some cases) at high precision in an average time of 9 to 12 minutes, while recall values tend to be lower. This tendency to favor precision at the cost of recall has been observed in previous works on rule-based RE (Wang et al., 2012). Nevertheless, we analyzed precision and recall in greater detail by manually evaluating a sample of 200 false positives and 200 false negatives by hand to discover the reasons for precision and recall

loss.

Mismatch between gold standard and results. As Table 2 shows, false positives are most commonly due to inconsistencies between extraction results and the gold standard annotations concerning the level of granularity of a relation instance. For example, we found BORNIN and DIEDIN relation instances that indicated a person's place of birth or death at lower or higher granularity than FREEBASE records. An example of this is given in Table 2 for Abraham Lincoln's place of death; we find the more granular <Lincoln, Hildene>, while the gold standard expects <Lincoln, Vermont>. While different from the gold standard, such instances are not false, which suggests that actual precision may be higher than the measured values indicate.

Missed patterns and entity types. The most common causes of recall loss are patterns that users failed to select. In Table 2, we distinguish between "common" patterns that were found by at least one user and "long tail" patterns that were found by none. While we did not expect a user-driven approach to identify long tail patterns, we were surprised that some users failed to find more common patterns. Similarly, the second most common cause of precision loss are entity type restrictions that users failed to correctly select, again to our surprise. We proceeded to interview the users to determine reasons for this.

3.3 Exploration Task

We also asked users to explore the corpus for a vaguely defined information need, namely for relations that pertain to "celebrities", as well as one arbitrary relation. Users spent widely varying amounts of time (between 5 and 50 minutes) on this task due to differences in motivation, as some users had interpreted the search for "interesting" relations as a challenge. For each relation, users provided a short description.

Some relations not in Freebase. While the most common types of relations found for entities of type CELEBRITY regarded different types of romantic involvements with other celebrities such as marriages and divorces, some relations were identified that are not found in FREEBASE. This included a relation that connects a celebrity to the sports team they support or the car they drive (see Table 3). This indicates a potential for using ERE to identify new relations for addition to existing knowledge bases.

Closed-class words can be relevant. Interestingly, one user also worked with patterns that involved closed-class word classes, such as "if" and "whether". Table 3 shown an example of a relation that indicates speculative birthplaces using such words.

3.4 User Feedback and Discussion

Approach more suited to exploration than extraction. When interviewing the users, we found that they generally favored the exploration over the extraction tasks as here the search could be directed to more fine-granular and specialized relations. One of the main problems encountered was the "halting problem", i.e. the question of when to stop adding patterns to an extractor. For some relations, such as BORNIN, users already found thousands of relation instances after selecting the first pattern, which caused two problems; First, they were unsure of the quality of the selected pattern(s), as they were unable to manually check thousands of relation instances for their validity. Second, they were unsure if more patterns were even needed if the first few already found such amounts of relation instances. These problems were not encountered in the exploration tasks, as here users could decide the information need for themselves and select patterns accordingly.

Difficulties concerning entity types. Another main difficulty related to the precise meaning of FREEBASE entity types; For instance, there are several location types, such as LOCATION.LOCATION, LOCATION.DATED_LOCATION and LOCATION.STATISTICAL_REGION, which users found to be confusing, a problem that was compounded by occasional entity linking errors. Many users expressed the desire to specify custom entity types as restrictions in order to have a similar level of control here as over the choice of patterns.

Low entry barriers but allow additional complexity. Overall, we found that users were generally able to start exploring the corpus using our workflow immediately after the brief introduction. Users stated the natural language-like representation of patterns to be intuitively readable, although for some it required a trial and error process to understand how patterns matched entities in sentences. Similarly, some users wished to understand in greater detail how entity types are determined and whether this could

FALSE POSITIVES		
CLASS	COUNT	EXAMPLE SENTENCE
FB Mismatch	95	**Lincoln** died at **Hildene** , his Vermont home, on July 26, 1926.
Type Error	82	[..] the scene where **Boromir** is killed in The **Fellowship of the Ring**.
FB Incomplete	14	Later that year, on December 27, **Dorr** died in **Providence**, in his native Rhode Island.
Other	9	Brieven van liederen **Rascal Flatts** die in het schijfcd album omvatten **Feels Like Today**.

FALSE NEGATIVES		
CLASS	COUNT	EXAMPLE SENTENCE
Common	87	**Klein** holds a **Bachelor of Arts**.
Long Tail	79	**Roger Blandford** is a native of England and took his **BA**, MA and [..].
Other	34	[..], 1974; **MS**, 1976; PhD, University of Pierre and **Marie Curie**, 1982.

Table 2: Analysis of 200 false positives and 200 false negatives to determine error classes for precision and recall loss. Each error class is listed with an example sentence. Main reasons for false positives included a mismatch in granularity between extraction results and annotations, wrongly specified types by the users or cases in which instances were found that were not in FREEBASE. Main reasons for false negatives were mostly patterns that users failed so select, either common patterns, or more rare patterns from the long tail.

NAME	DESCRIPTION	EXAMPLE PATTERNS	EXAMPLE INSTANCES
CELEBRITYDIVORCE	Divorce between two celebrities	"X and Y divorce", "X divorce Y",	<Nicole Kidman, Tom Cruise> <Federline, Spears>
CELEBRITYDRIVESCAR	Finds the cars that celebrities drive	"X drives Y", "X 's car Y",	<Arnold Schwarzenegger, H1> <Leonardo DiCaprio, Toyota Prius>
CONTESTEDBITHPLACE	Relates persons to their speculative birthplace	"if X born in Y", "whether X born in Y",	<Barack Obama, Kenya> <Barack Obama, Nigeria>

Table 3: Examples for relations discovered in the exploration task. CELEBRITYDIVORCE represents a commonly discovered relation, while CELEBRITYDRIVESCAR represents a relation that is presently not part of Freebase. CONTESTEDBITHPLACE is an example of a relation that utilizes closed-world words in patterns.

be influenced. This indicates the need for adding options in future work that give more experienced users more technical information (and control) on dependency trees and FREEBASE types.

4 Previous Work

While no directly comparable approach to Exploratory Relation Extraction is known to us, we take inspiration from a number of previous works.

Exploratory Search (Marchionini, 2006; White and Roth, 2009) is an information seeking paradigm in the field of Information Retrieval, where - like in our proposed approach - users begin an exploration process with an imprecise information need and progressively discover available information to address and sharpen it. Unlike our approach, users search for documents and must consume the unstructured information themselves. We instead apply this paradigm to RE and strive to find structured, relational information in text corpora of unknown content as well as generate Realtion Extractors in the process.

Preemptive Information Extraction (Shinyama and Sekine, 2006), as well as much work in Open Information Extraction (Yates et al., 2007) that builds on this idea, is the preemptive (or open) extraction of all possible relations in a text corpus. We draw inspiration from this idea in our preemptive subtree generation approach; however, while we extract all possible subtrees for each relation regardless of whether they point to a relation or not, Preemptive and OpenIE approaches aim to produce facts and therefore much more narrowly extract predicates using rule-sets (Del Corro and Gemulla, 2013), classifiers (Schmitz et al., 2012) or both (Etzioni et al., 2011).

Manual Rule-Based RE. We also build our work on the field of manual, rule-based RE, which has been observed to be predominantly preferred industry solution due to interpretability of extraction rules and

easy adaption to changing domains (Chiticariu et al., 2013; Chiticariu et al., 2010). The lack of tools to assist rule developers in exploring and choosing between different automatically generated rules has been stated to be one of the major challenges associated with rule-based RE systems. Recent research has moved towards more guided (Li et al., 2012) and more interactive (Akbik et al., 2013b) workflows for the creation of rule-based extractors. Our proposed approach follows this direction, but is the first approach to combine both with automatic suggestions and enable exploratory search for relations.

Precomputing Resources of Relational Patterns. Our work also bears some resemblance to previous work that have grouped similar extraction patterns into clusters (Li et al., 2011) or arranged them in a taxonomy (Nakashole et al., 2012), with the goal of facilitating relation extraction efforts. Contrary to these works, we do not precompute a static resource but rather continuously re-compute pattern suggestions on the basis of user interactions and the text corpus that the user is working with. In addition, our suggestions are based on both user-selected patterns as well as entity type restrictions.

5 Conclusion and Future Work

In this paper, we proposed Exploratory Relation Extraction as a method of exploring text corpora of uncertain content for relations of interest given an imprecise information need. We have presented and evaluated a user-driven and data-guided incremental exploration workflow that enables non-expert users to identify relations and create high precision extractors with minimal effort. Our results indicate that applying ideas from Exploratory Search to RE is beneficial and can extend the application of RE to use cases characterized by more imprecise information needs and uncertainty regarding the information content of available data. In order to facilitate the discussion of our approach with the research community, we release our work publicly through a Web demonstrator[3].

Future work will investigate extending the approach to relations that hold between an arbitrary number of entities as well as the detection of custom entity types. We aim to allow users to store and combine extractors - for example relation extractors that use custom entity type detectors - to address more complex information needs and distribute the exploration and extraction processes along larger groups of users. This way we seek to enable collaborative RE approaches for creating large knowledge bases from text.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This research is funded by the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no ICT-2009-4-1 270137 'Scalable Preservation Environments' (SCAPE) and the German Federal Ministry of Education and Research (BMBF) under grant no. 01ISI2033 RADAR'.

References

A. Akbik, L. Visengeriyeva, J. Kirschnick, and A. Löser. 2013a. Effective selectional restrictions for unsupervised relation extraction. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*.

Alan Akbik, Oresti Konomi, and Michail Melnikov. 2013b. Propminer: A workflow for interactive information extraction and exploration using dependency trees. In *ACL System Demonstrations*. Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick R Reiss, and Shivakumar Vaithyanathan. 2010. Systemt: an algebraic approach to declarative information extraction. In *ACL*, pages 128–137. Association for Computational Linguistics.

Laura Chiticariu, Yunyao Li, and Frederick R Reiss. 2013. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 827–832.

Jinho D Choi and Andrew McCallum. 2013. Transitionbased dependency parsing with selectional branching. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria*.

[3]The demonstrator is available at http://lucene.textmining.tu-berlin.de/

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics.

Jeffrey Dean and Sanjay Ghemawat. 2004. MapReduce: simplified data processing on large clusters. In *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, OSDI'04, pages 137–150, Berkeley, CA, USA. USENIX Association.

Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*, pages 355–366. International World Wide Web Conferences Steering Committee.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open information extraction: The second generation. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume One*, pages 3–10. AAAI Press.

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. FACC1: freebase annotation of ClueWeb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0).

Yunyao Li, Vivian Chu, Sebastian Blohm, Huaiyu Zhu, and Howard Ho. 2011. Facilitating pattern discovery for relation extraction with semantic-signature-based clustering. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1415–1424. ACM.

Yunyao Li, Laura Chiticariu, Huahai Yang, Frederick R Reiss, and Arnaldo Carreno-fuentes. 2012. Wizie: a best practices guided development environment for information extraction. In *Proceedings of the ACL 2012 System Demonstrations*, pages 109–114. Association for Computational Linguistics.

Gary Marchionini. 2006. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46.

M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL/AFNLP*, pages 1003–1011.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. Patty: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1135–1145. Association for Computational Linguistics.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*, volume 59.

Frederick Reiss, Sriram Raghavan, Rajasekar Krishnamurthy, Huaiyu Zhu, and Shivakumar Vaithyanathan. 2008. An algebraic approach to rule-based information extraction. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 933–942. IEEE.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049.

Sunita Sarawagi. 2008. Information extraction. *Foundations and trends in databases*, 1(3):261–377.

Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534. Association for Computational Linguistics.

Alexander Schutz and Paul Buitelaar. 2005. Relext: A tool for relation extraction from text in ontology extension. In *The Semantic Web–ISWC 2005*, pages 593–606. Springer.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 304–311. Association for Computational Linguistics.

Jannik Strötgen and Michael Gertz. 2010. Heideltime: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324. Association for Computational Linguistics.

Hans Uszkoreit. 2011. Learning relation extraction grammars with minimal human intervention: strategy, results, insights and plans. In *Computational Linguistics and Intelligent Text Processing*, pages 106–126. Springer.

Chang Wang, Aditya Kalyanpur, James Fan, Branimir K Boguraev, and DC Gondek. 2012. Relation extraction and scoring in deepqa. *IBM Journal of Research and Development*, 56(3.4):9–1.

Ryen W White and Resa A Roth. 2009. Exploratory search: Beyond the query-response paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 1(1):1–98.

Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 25–26. Association for Computational Linguistics.

An Analysis of Causality between Events and its Relation to Temporal Information

Paramita Mirza
Fondazione Bruno Kessler,
University of Trento
Trento, Italy
paramita@fbk.eu

Sara Tonelli
Fondazione Bruno Kessler
Trento, Italy
satonelli@fbk.eu

Abstract

In this work we present an annotation framework to capture causality between events, inspired by TimeML, and a language resource covering both temporal and causal relations. This data set is then used to build an automatic extraction system for causal signals and causal links between given event pairs. The evaluation and analysis of the system's performance provides an insight into explicit causality in text and the connection between temporal and causal relations.

1 Introduction

Causality is a concept that has been widely investigated from a philosophical, psychological and logical point of view, but how to model its recognition and representation in NLP-centered applications is still an open issue. However, information on causality could be beneficial to a number of natural language processing tasks such as question answering, text summarization, decision support, etc. The lack of information extraction systems focused on causality may depend also on the lack of unified annotation guidelines and standard benchmarks, which usually foster the comparison of different systems performances. Specific phenomena related to causality, such as causal arguments (Bonial et al., 2010), causal discourse relations (The PDTB Research Group, 2008) or causal relations between nominals (Girju et al., 2007), have been investigated, but no unified framework has been proposed to capture causal relations between events, as opposed to the existing TimeML standard for temporal relations (Pustejovsky et al., 2010).

The work presented in this paper copes with this issue by *i)* proposing an annotation framework to model causal relations between events and *ii)* detailing the development and the evaluation of a supervised system based on such framework.

We take advantage of the formalization work carried out for the TimeML standard, in which events, temporal relations and temporal signals have been carefully defined and annotated. We propose to model causal relations in a similar way to temporal relations, inheriting from TimeML the notion of event, relation and signal, even though our approach to causality is well rooted in the *force dynamic* model by Talmy (1985).

Besides, we focus our preliminary annotation on TimeBank (Pustejovsky et al., 2006), a corpus widely used by the research community working on temporal processing. This should possibly enable the adaptation of existing temporal processing systems to the analysis of causal information, given that we rely on well-known standards and data. On the other hand, this makes it easier for us to straightforwardly investigate the relation between temporal and causal information, given that a causing event should always take place *before* a resulting event.

2 Related Work

Research on the extraction of event relations has concerned both the analysis of the temporal ordering of events and the recognition of causality relations. However, the two research lines have progressed quite independently from each other. Recent works on temporal relations mostly revolve around the last

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2097–2106, Dublin, Ireland, August 23-29 2014.

TempEval-3[1] shared task on temporal and event processing. The task organizers released some data sets annotated with events, time expressions and temporal relations in TimeML format (Pustejovsky et al., 2003), mainly used for training and evaluation purposes. The results of TempEval-3 reported by UzZaman et al. (2013) show that, even though the performance of systems for extracting TimeML events and time expressions is quite good (>80% F-score), the overall performance of end-to-end event extraction pipelines is negatively affected by the poor performance of modules for temporal relation extraction. In fact, the state-of-the-art performance on the temporal relation extraction task yields only around 36% F-score (Bethard, 2013).

The problem of detecting causality between events is as challenging as recognizing their temporal order, but less analyzed from an NLP perspective. Besides, it has mostly focused on specific types of event pairs and causal expressions in text, and has failed to provide a global account of causal phenomena that can be captured with NLP techniques. SemEval-2007 Task 4 "Classification of Semantic Relations between Nominals" (Girju et al., 2007) gives access to a corpus containing nominal causal relations among others, as causality is one of the considered semantic relations in the task. Bethard et al. (2008) collected 1,000 conjoined event pairs connected by *and* from the Wall Street Journal corpus. The event pairs were annotated manually with both temporal (BEFORE, AFTER, NO-REL) and causal relations (CAUSE, NO-REL). They use 697 event pairs to train a classification model for causal relations, and use the rest for evaluating the system, which results in 37.4% F-score. Rink et al. (2010) perform textual graph classification using the same corpus, and make use of manually annotated temporal relation types as a feature to build a classification model for causal relations between events. This results in 57.9% F-score, 15% improvement in performance compared with the system without the additional feature of temporal relations.

The interaction between temporal and causal information, and the contribution of temporal information to the identification of causal links, are also one of the issues investigated in this paper. However, we aim at providing a more comprehensive account of how causal relations can be explicitly expressed in a text, and we do not limit our analysis to specific connectives.

Do et al. (2011) developed an evaluation corpus by collecting 20 news articles from CNN, allowing the detection of causality between *verb-verb*, *verb-noun*, and *noun-noun* triggered event pairs. Causality between event pairs is measured by taking into account Point-wise Mutual Information (PMI) between the cause and the effect. They also incorporate discourse information, specifically the connective types extracted from the Penn Discourse TreeBank (PDTB), and achieve a performance of 46.9% F-score. Unfortunately, the data set is not freely available, hence, comparing our work with theirs is not possible.

The most recent work of Riaz and Girju (2013) focuses on the identification of causal relations between verbal events. They rely on the unambiguous discourse markers *because* and *but* to automatically collect training instances of cause and non-cause event pairs, respectively. The result is a knowledge base of causal associations of verbs, which contains three classes of verb pairs: *strongly causal*, *ambiguous* and *strongly non-causal*.

The lack of a standard benchmark to evaluate systems for the extraction of causal relations between events makes it difficult to compare the performance of different systems, and to identify the state-of-the-art approach to this particular task. For this reason, we annotated TimeBank, a freely available corpus, with the aim of making it available to the research community for further evaluations.

3 Data annotation

In order to develop a classifier for the detection of causal relations between events, we first define annotation guidelines for explicit causality and then manually annotate a data set for training and testing.

3.1 Annotation scheme

Since one of the goals of this work is to investigate the interaction between temporal and causal information, we define an annotation scheme strongly inspired by the TimeML standard for events, time expressions and temporal relations. First, we inherit from TimeML the definition of events, which includes all types

[1] http://www.cs.york.ac.uk/semeval-2013/task1/

of actions (punctual and durative) and states. Hence, we do not limit our annotation only to specific PoS such as verbal or nominal events.

Similar to the <TLINK> tag in TimeML for temporal relations, we introduce the <CLINK> tag to mark a causal relation between two events. Both TLINKs and CLINKs mark directional relations, i.e. they involve a source and a target event. However, while a list of relation types is part of the attributes for TLINKs (e.g. BEFORE, AFTER, INCLUDES, etc.), for CLINKs only one relation type is foreseen, going from a *source* (the cause, indicated with $_{S}$ in the examples) to a *target* (the effect, indicated with $_{T}$).

We also introduce the notion of causal signals through the <C-SIGNAL> tag. <SIGNAL>s have been introduced in TimeML to annotate temporal prepositions and other temporal connectives and subordinators. If a SIGNAL marks the presence of a temporal relation in a text, its ID is added to the attributes of such TLINK. In a similar way, C-SIGNALs are used to mark-up textual elements signalling the presence of causal relations, which include all causal uses of *prepositions* (e.g. because of, as a result of, due to), *conjunctions* (e.g. because, since, so that), *adverbial connectors* (e.g. so, therefore, thus) and *clause-integrated expressions* (e.g. the reason why, the result is, that is why). Also for CLINKs it is possible to assign a *c-signalID* attribute, in case a C-SIGNAL marks the causal relation between two events in text.

Concerning the notion of causality, it is particularly challenging to provide guidelines that clearly define how to identify it in text, since causality exists as a psychological tool for understanding the world independently of language and it is not necessarily grounded in text (van de Koot and Neeleman, 2012). There have been several attempts in the psychology field to model causality, including the counterfactual model (Lewis, 1973), the probabilistic contrast model (Cheng and Novick, 1991; Cheng and Novick, 1992) and the dynamics model (Wolff and Song, 2003; Wolff et al., 2005; Wolff, 2007), which is based on Talmy's force dynamic account of causality (Talmy, 1985; Talmy, 1988). We choose to lean our guidelines on the latter model, since it accounts also for different ways in which causal concepts are lexicalized.

Specifically, Wolff (2007) claims that causation covers three main types of causal concepts, i.e. CAUSE, ENABLE and PREVENT. These causal concepts are lexicalized through three types of verbs listed in Wolff and Song (2003): *i)* CAUSE-type verbs, e.g. *cause, prompt, force*; *ii)* ENABLE-type verbs, e.g. *allow, enable, help*; and *iii)* PREVENT-type verbs, e.g. *block, prevent, restrain*. These categories of causation and the corresponding verbs are taken into account in our guidelines (Tonelli et al., 2014).

We assign a CLINK if, given two annotated events, there is an explicit causal construction linking them. Such construction can be expressed in one of the following ways:

1. Expressions containing **affect verbs** (*affect, influence, determine, change, etc.*), e.g. *Ogun ACN crisis* $_{S}$ ***influences*** *the launch* $_{T}$ *of the All Progressive Congress*.
2. Expressions containing **link verbs** (*link, lead, depend on, etc.*), e.g. *An earthquake* $_{T}$ *in North America was* ***linked*** *to a tsunami* $_{S}$ *in Japan*.
3. **Basic constructions involving causative verbs** of CAUSE, ENABLE and PREVENT type, e.g. *The purchase* $_{S}$ ***caused*** *the creation* $_{T}$ *of the current building*.
4. **Periphrastic constructions involving causative verbs** of CAUSE, ENABLE and PREVENT type, e.g. *The blast* $_{S}$ ***caused*** *the boat to heel* $_{T}$ *violently*. With "periphrastic" we mean constructions where a causative verb (*caused*) takes an embedded clause or predicate as a complement expressing a particular result (*heel*).
5. Expressions containing **CSIGNALs**, e.g. *Its shipments declined* $_{T}$ ***as a result of*** *a reduction* $_{S}$ *in inventories by service centers*.

We annotate both intra- and inter-sentential causal relations between events, provided that one of the above constructions is present. We do not annotate causal relations that are implicit and must be inferred by annotators, because they may be highly ambiguous and would probably affect inter-annotator agreement.

3.2 Corpus statistics

Based on the guidelines above, we manually annotated causality in the TimeBank corpus taken from TempEval-3, containing 183 documents with 6,811 annotated events in total.[2] We chose this corpus because gold events were already present, between which we could add causal links. Besides, one of our research goals is the analysis of the interaction between temporal and causal information, and TimeBank already presents full manual annotation of temporal information according to TimeML standard.

However, during annotation, we noticed that some events involved in causal relations were not annotated, probably because the corpus was originally built focusing on events involved in temporal relations. Therefore, we annotated also 137 new events, which led to around 56% increase in the number of annotated CLINKs.

The total number of annotated CSIGNALs is 171 and there are 318 CLINKs, much less than the number of TLINKs found in the corpus, which is 5,118. Besides, not all documents contain causality relations between events. From the total number of documents in TimeBank, only 109 (around 60%) of them contain explicit causal links and only 87 (around 47%) of them contain CSIGNALs. We also found that there is no temporal signal (marked by <SIGNAL> tag) annotated in TimeBank, which is unfortunate since it could help in disambiguating causal signals from temporal signals.

Annotation was performed using the CAT tool (Bartalesi Lenzi et al., 2012), a web-based application with a plugin to import annotated data in TimeML and add information on top of it. The agreement reached by two annotators on a subset of 5 documents is 0.844 Dice's coefficient on C-SIGNALs (micro-average over markables) and of 0.73 on CLINKs. The built corpus is then used as training and test data in the experiments for the classification of CSIGNALs and CLINKs, as described in Section 4. This preliminary analysis on the corpus, however, shows that explicit causal relations between events are less frequently found in texts than temporal ones. This may lead to data sparseness problems.

4 Experiments

Using the 183 documents from TimeBank manually enriched with causal information for training and testing, we implement two different classifiers: the first one is a CSIGNAL labeler, that takes in input information on events and temporal expressions as annotated in the original TimeBank, and classifies whether a token is part of a causal signal or not (Section 4.1). The second one is a CLINK classifier, which given an event pair detects whether they are connected by an explicit causal link (Section 4.2). Both experiments are carried out based on five-fold cross-validation. The overall approach is largely inspired by our existing framework for the classification of temporal relations (Mirza and Tonelli, 2014).

4.1 Automatic Extraction of CSIGNALs

The task of recognizing CSIGNALs can be seen as a text chunking task, i.e. using a classifier to determine whether a token is part of a causal signal or not. Since the extent of causal signals can be expressed by multi-word expressions, we employ the IOB tagging convention to annotate the data, where each token can either be classified into B-CSIGNAL, I-CSIGNAL or O (for other). We build our classification model using the Support Vector Machine (SVM) implementation provided by YamCha[3], a generic, customizable, and open source text chunker. In order to provide the classifier a feature vector to learn from, we perform the two following steps:

1. Run the *TextPro* tool (Pianta et al., 2008) to get information on base NP chunking and whether a token is part of named entity or not.
2. Run *Stanford CoreNLP* tool[4] to get information on lemma, part-of-speech (PoS) tags and dependency relations between tokens.

In the end, the feature vector includes *token*, *lemma*, *PoS tag*, *NP chunking*, *dependency path*, and *several binary features*, indicating whether a token is: *i)* an event or part of a temporal expression,

[2]The annotated data set is available at `http://hlt.fbk.eu/technologies/causal-timebank`

[3]`http://chasen.org/~taku/software/yamcha/`

[4]`http://nlp.stanford.edu/software/corenlp.shtml`

according to gold TimeML annotation; *ii)* part of a named entity or not; and *iii)* part of a specific discourse connective type.

Dependency information is encoded as the dependency path between the current token and its *governor*. For example, in *"He fell because the chair was broken"*, there is a dependency relation *mark (broken, because)*, where *mark* indicates the presence of a finite clause subordinate to another clause (de Marneffe and Manning, 2008). Thus, we encode the dependency feature for the token *because* as *mark (broken)*. If the governor is an event, e.g. *broken* is annotated as an event, the dependency feature is represented as *mark (*EVENT*)* instead.

The mentioned binary features are introduced to exclude the corresponding token as a candidate token for a causal signal. In other words, if a token is part of a named entity or an event, it is very unlikely that it will be part of a causal signal. The same holds for all connective types that do not express causal relations, e.g. temporal or concessive ones. In order to obtain this information, we include in the feature vector the information about discourse connectives acquired using the *addDiscourse* tool (Pitler and Nenkova, 2009), which identifies connectives and assigns them to one of four semantic classes in the framework of the Penn Discourse Treebank (The PDTB Research Group, 2008): TEMPORAL, EXPANSION, CONTINGENCY and COMPARISON. Note that causality is part of the CONTINGENCY class.

System	Precision	Recall	F-score
Rule-based (baseline)	54.33%	40.35%	46.31%
Supervised chunking	91.03%	41.76%	57.26%

Table 1: Evaluation of CSIGNAL extraction system

Table 1 shows the performance of our classification model in a five-fold cross-validation setting, which yields a good precision but a poorer recall, summing up into 57.26% F-score. We also compare our supervised model with a baseline rule-based system, which labels as CSIGNALs all causal connectors listed in our annotation guidelines and those appearing in specific syntactic constructions. For instance, *from* and *by* are always labeled as CSIGNAL when they are governed by a passive verb annotated as event and govern another event, as in the sentence *"The building was* damaged $_{\text{T}}$ ***by*** *the* earthquake $_{\text{S}}$*."* Note that this is quite a strong baseline, since the rule-based algorithm embeds some of the intuitions on syntactic dependencies expressed also as features in the supervised approach.

4.2 Automatic Extraction of CLINKs

Similar to causal signal extraction, we approach the problem of detecting causal links between events as a supervised classification task. Given an ordered pair or events (e_1,e_2), the classifier has to decide whether there is a causal relation between them or not. However, since we also consider the directionality of the causal link, an event pair (e_1,e_2) is classified into 3 classes: CLINK (where e_1 is the source and e_2 is the target), CLINK-R (with the reverse order or source and target) or NO-REL. Again, we use YamCha to build the classifier. This time, a feature vector is built for each pair of events and not for each token as in the previous classification task.

As candidate event pairs, we take into account every possible combination of events in a sentence in a forward manner. For example, if we have e_1, e_2 and e_3 in a sentence (in this order), the candidate event pairs are (e_1,e_2), (e_1,e_3) and (e_2,e_3). We also include as candidate event pairs the combination of each event in a sentence with events in the following one. This is necessary to account for inter-sentential causality, under the simplifying assumption that causality may occur only between events in two consecutive sentences.

We implement a number of features, some of which are computed independently based on either e_1 or e_2, e.g. lemma, PoS, while some others are pairwise features, which are computed based on both elements, e.g. dependency path, signals in between, etc. The implemented features are as follows:

String and grammatical features. The tokens and lemmas of e_1 and e_2, along with their PoS and a binary feature indicating whether e_1 and e_2 have the same PoS tags.

Textual context. The sentence distance and event distance of e_1 and e_2. Sentence distance measures

how far e_1 and e_2 are from each other in terms of sentences, i.e. 0 if they are in the same sentence. The event distance corresponds to the number of events occurring between e_1 and e_2 (i.e. if they are adjacent, the distance is 0).

Event attributes. Event attributes as specified in TimeML annotation, which consist of *class*, *tense*, *aspect* and *polarity*. Events being a noun, adjective and preposition do not have tense and aspect attributes in TimeML. Therefore, we retrieve this information by extracting the tense and aspect of the verbs that govern them, based on their dependency relation. We also include four binary features representing whether e_1 and e_2 have the same event attributes or not. These features, especially the *tense* and *aspect* one, are very relevant for detecting causality. For instance, if e_1 is in the future tense and e_2 in the past tense, there cannot be a causal relation connecting e_1 (as source) and e_2 (as target or result).

Dependency information. We include as features *i)* the dependency path that exists between e_1 and e_2, *ii)* the type of causative verb connecting them (if any) and *iii)* binary features indicating whether e_1/e_2 is the *root* of the sentence. This information is based on the collapsed representation of dependency relations provided by the parsing module of Stanford CoreNLP. Consider the sentence *"Profit from coal fell* $_{\text{T}}$ *to $41 million from $58 million, partly* ***because of*** *a miners' strike* $_{\text{S}}$*."* Based on the collapsed typed dependencies, we would obtain a direct relation between *fell* and *strike*, which is *prep_because_of (fell, strike)*. This information combined with the classification of *because of* as a causal signal would straightforwardly identify the relation connecting the two events as causal.

Causal signals. We take into account the annotated CSIGNALs connecting two candidate events. We look for causal signals occurring between e_1 and e_2, or before e_1. We also include the position of the signals (*between* or *before*) as feature, since it is crucial to determine the direction of the causality of a given ordered event pair. This is particularly evident if you consider the position of causal signals in the following examples: *i)* "The building collapsed $_{\text{T}}$ **because of** the earthquake $_{\text{S}}$" vs. *ii)* "**Because of** the earthquake $_{\text{S}}$ the building collapsed $_{\text{T}}$." This feature is also very relevant in connection with the *Textual context*, since two events being in two different sentences are linked by an explicit causal relation only in specific cases, for instance if there is a CSIGNAL in between, typically at the beginning of the second sentence. Note that in case of several CSIGNALs occurring between e_1 and e_2, we take the closest CSIGNAL to e_2, as in the sentence "The building was damaged $_{\text{S}}$ **by** the earthquake , **thus**, people moved $_{\text{T}}$ away". The dependency path between the causal signal and e_1/e_2 is also important to determine the correct involved events in the causal relations. For instance, in the sentence "They decided $_{\text{T}}$ to move **because of** the earthquake $_{\text{S}}$", the involved event is *decided* instead of *move*.

Temporal relations (TLINKs). Rink et al. (2010) showed that including temporal relation information in detecting causal links results in improving classification performance. Nevertheless, they only analyze this phenomenon when causality is expressed by the conjunction *and*. We decided to include this information in the feature set by specifying the temporal relation type connecting e_1 and e_2, if any, to see whether TLINKs help in improving causality detection also in a more comprehensive setting.

We evaluate our approach in a five-fold cross-validation setting, and we compare the performance of our classifier with a baseline rule-based system. This relies on an algorithm that, given a term t belonging to *affect*, *link*, *causative* verbs (basic and periphrastic constructions) or *causal signals* (as listed in the annotation guidelines), looks for specific dependency constructions where t is connected to two events. If such dependencies are found, a CLINK is automatically set between the two events identifying the source and the target of the relation. Further details on the baseline system and its evaluation can be found in Mirza et al. (2014).

In our experimental setting, we evaluate two versions of the CLINK classifier: the first includes as features the *gold annotated* CSIGNALs in the classification model, while the second takes in input the CSIGNALs *automatically annotated* by the classifier described in Section 4.1. We also evaluate the contribution of *dependency*, *CSIGNAL* and *TLINK* features by excluding each of them from the classification model.

Evaluation results are reported in Table 2. We observe that the baseline is always outperformed by the other classifiers. CSIGNAL is the most important feature, with a particularly high impact on recall. The

intuition behind this result is that, if a CSIGNAL is present, it is a strong indicator of a causal relation being present in the surrounding context. This is similar to what Derczynski and Gaizauskas (2012) report for temporal information, showing that temporal signals provide useful information in TLINK classification. Dependency information contributes to the performance of the classifier, but is less relevant than TLINK information. A more detailed analysis of the relation between temporal and causal information is reported in the following section. The significantly decreasing recall of the classifier using the automatic extracted CSIGNALs as features is most probably caused by the low recall of the CSIGNAL extraction system.

System	Precision	Recall	F-score
Rule-based (baseline)	36.79%	12.26%	18.40%
Supervised classification (with gold CSIGNALs)	74.67%	35.22%	47.86%
- without dependency feature	65.77%	30.82%	41.97%
- without CSIGNAL feature	57.53%	13.21%	21.48%
- without TLINK feature	61.59%	29.25%	39.66%
Supervised classification (with automatic CSIGNALs)	67.29%	22.64%	33.88%

Table 2: Performance of CLINK extraction system

5 Discussion

We further analyse the output of the automatic extraction systems, in order to understand some phenomena triggering the results.

5.1 Recognizing CSIGNALs

When we manually inspect the output of the CSIGNAL extraction system, we find that the false positives are actually the causal signals that annotators missed in the corpus, and not ambiguous connectives. The system surprisingly yields better precision than human annotation, finding new correct signals.

The recall, however, suffers most probably from data sparseness. It is possible that during the cross-validation experiments some splits do not have enough data to learn from, recalling that only around 47% of the documents contain annotated CSIGNALs. Furthermore, 20% of the false negative cases are due to classifier's mistakes in detecting the causal signal *by*, which is highly ambiguous. Our assumption with the rule-based system that "*by* is likely to be a causal signal when it is used to modify a passive verb" is too restrictive, since *by* can convey a causal meaning even if the target event is not in the passive voice, as in the example *"The embargo is meant to cripple* $_{\text{T}}$ *Iraq* ***by*** *cutting* $_{\text{S}}$ *off its exports of oil and imports of food and military supplies."*

Another ambiguous causal signal that the classifier fails to detect is the conjunction *and*. We believe that more training data, and perhaps more lexical information on the tokens connected by the conjunction *and*, are needed for the classifier to be able to disambiguate them.

5.2 Detecting CLINKs

We found that most of the mistakes done by the classifier, as well as by the rule-based system, are caused by the dependency parser output that tends to establish a dependency relation between a causative verb or causal signal and the closest verb. For example, in the sentence *"StatesWest Airlines withdrew* $_{\text{T}}$ *its offer to acquire Mesa Airlines* ***because*** *the Farmington carrier did not respond* $_{\text{S}}$ *to its offer"*, the dependency parser identify *because* as the mark of *acquire* instead of *withdrew*.

Moreover, also for this task data sparseness is definitely an issue. One possible solution would be to annotate more data, for instance the AQUAINT data set used for TempEval-3 competition (UzZaman et al., 2013). Another possibility would be to automatically generate additional data from the Penn Discourse TreeBank corpus, where causality is one of the discourse relations annotated between argument pairs. However, a further processing step would be needed to identify inside the argument spans the events between which a relation holds, which may introduce some errors.

Regarding the directionality of causal relations, the classifier is generally quite precise. 112 out of 150 CLINKs detected by the classifier actually match a causal relation present in the gold annotated data. Only 8 of them have been classified with the wrong direction. We believe that using the TLINK types as features contributes to this good performance in disambiguating causality direction (CLINK vs. CLINK-R).

5.3 Interaction between temporal and causal information

We provide in Table 3 some statistics on the overlaps between causal links and temporal relation types from the gold data. The *Others* class in the table includes SIMULTANEOUS, IS_INCLUDED, BEGUN_BY and DURING_INV relations. These counts were obtained by overlapping the temporal information in TimeBank with the causal information manually added for our experiments. In total, only 32% of the gold causal links have the underlying temporal relations. Note that the annotators could not see the temporal links already present in the data, therefore they were not biased by TLINKs when assessing causal links.

	BEFORE	AFTER	IBEFORE	IAFTER	Others	Total
CLINK	15	5	0	0	4	24
CLINK-R	1	67	0	3	8	79

Table 3: Statistics of CLINKs overlapping with TLINKs

The data confirm our intuition that temporal information is a strong constraint when detecting causal relations, with the BEFORE class having the most overlaps with CLINK and AFTER with CLINK-R. This is in line with the outcome of our feature analysis reported in Table 2, suggesting that feeding temporal information into a causal relation classifier yields an improvement in performance. However, the converse would be less effective, since the occurrences of explicit causal relations are by far less frequent than temporal ones. Besides, we found that the few cases where CLINKs overlap with AFTER relation are not due to annotation mistakes, as in the example *"But some analysts questioned* $_{\text{T}}$ *how much of an impact the retirement package will have,* ***because*** *few jobs will end* $_{\text{S}}$ *up being eliminated."*

Finally, the performance achieved by our system in causal relation extraction (with gold C-SIGNALs) is 47.86% F-score, which is better than the performance of the state-of-the-art temporal relation extraction system with 36.26% (Bethard, 2013). This probably depends on the fact that extracting CLINKs is a simpler task compared with TLINK extraction: in the first case 3 classes are considered, while temporal relation types are classified into 14 classes.

6 Conclusions

In this paper, we presented a framework for annotating causal signals and causal relations between events. Besides, we implemented and evaluated two supervised systems, one classifying C-SIGNALs and the other CLINKs.

With the first task, we showed that while recognizing unambiguous causal signals is very trivial, ambiguous signals such as *by* and *and* are very difficult to identify because they occur in diverse syntactic constructions. We definitely need more data to learn from, and perhaps use more lexical information on the words connected by such causal signals as features. The knowledge base of causal associations between verbs developed by Riaz and Girju (2013) may be a useful resource to provide such information, and we will explore this possibility in the future.

We found that the low recall achieved by the CLINK classifier is probably affected by wrong dependencies identified by the Stanford parser. In the future, we would like to test also the C&C tool (Curran et al., 2007) to extract dependency relations, since it has a better coverage of long-range dependencies. We have also shown that causal signals are very important in detecting explicit causal links holding between two events. Finally, we showed that temporal relation types help in disambiguating the direction of causality, i.e. to determine the source and target event. However, the converse may not hold, since the causal links in the data set are very sparse, and only 2% of the total TLINKs overlap with CLINKs.

Acknowledgements

The research leading to this paper was partially supported by the European Union's 7th Framework Programme via the NewsReader Project (ICT-316404). We also thank Rachele Sprugnoli and Manuela Speranza for their contribution in defining the annotation guidelines.

References

Valentina Bartalesi Lenzi, Giovanni Moretti, and Rachele Sprugnoli. 2012. CAT: the CELCT Annotation Tool. In *Proceedings of LREC 2012*.

Steven Bethard, William Corvey, Sara Klingenstein, and James H. Martin. 2008. Building a Corpus of Temporal-Causal Structure. In European Language Resources Association (ELRA), editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may.

Steven Bethard. 2013. ClearTK-TimeML: A minimalist approach to TempEval 2013. In *Proceedings of the Seventh International Workshop on Semantic Evaluation*, SemEval '13, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Claire Bonial, Olga Babko-Malaya, Jinho D. Choi, Jena Hwang, and Martha Palmer. 2010. PropBank Annotation Guidelines, Version 3.0. Technical report, Center for Computational Language and Education Research, Institute of Cognitive Science, University of Colorado at Boulder. `http://clear.colorado.edu/compsem/documents/propbank_guidelines.pdf`.

Patricia W. Cheng and Laura R. Novick. 1991. Causes versus enabling conditions. *Cognition*, 40(1-2):83 – 120.

Patricia W. Cheng and Laura R. Novick. 1992. Covariation in natural causal induction. *Psychological Review*, 99(2):365–382.

James Curran, Stephen Clark, and Johan Bos. 2007. Linguistically Motivated Large-Scale NLP with C&C and Boxer. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 33–36, Prague, Czech Republic, June. Association for Computational Linguistics.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8. Association for Computational Linguistics.

Leon Derczynski and Robert J. Gaizauskas. 2012. Using Signals to Improve Automatic Classification of Temporal Relations. *CoRR*, abs/1203.5055.

Quang Xuan Do, Yee Seng Chan, and Dan Roth. 2011. Minimally Supervised Event Causality Identification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Stroudsburg, PA, USA. Association for Computational Linguistics.

Roxana Girju, Preslav Nakov, Vivi Nastase, Stan Szpakowicz, Peter Turney, and Deniz Yuret. 2007. SemEval-2007 Task 04: Classification of Semantic Relations between Nominals. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 13–18, Prague, Czech Republic, June. Association for Computational Linguistics.

David Lewis. 1973. Causation. *The Journal of Philosophy*, 70(17):pp. 556–567.

Paramita Mirza and Sara Tonelli. 2014. Classifying Temporal Relations with Simple Features. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 308–317, Gothenburg, Sweden, April. Association for Computational Linguistics.

Paramita Mirza, Rachele Sprugnoli, Sara Tonelli, and Manuela Speranza. 2014. Annotating Causality in the TempEval-3 Corpus. In *Proceedings of the EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)*, pages 10–19, Gothenburg, Sweden, April. Association for Computational Linguistics.

Emanuele Pianta, Christian Girardi, and Roberto Zanoli. 2008. The TextPro Tool Suite. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco. European Language Resources Association.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 13–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003. TimeML: Robust specification of event and temporal expressions in text. In *Proceedings of the Fifth International Workshop on Computational Semantics (IWCS-5)*.

James Pustejovsky, Jessica Littman, Roser Saurí, and Marc Verhagen. 2006. Timebank 1.2 documentation. Technical report, Brandeis University, April.

James Pustejovsky, Kiyong Lee, Harry Bunt, and Laurent Romary. 2010. ISO-TimeML: An international standard for semantic annotation. In *Proceedings o the Fifth International Workshop on Interoperable Semantic Annotation*.

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *Proceedings of the SIGDIAL 2013 Conference*, pages 21–30, Metz, France, August. Association for Computational Linguistics.

Bryan Rink, Cosmin Adrian Bejan, and Sanda M. Harabagiu. 2010. Learning Textual Graph Patterns to Detect Causal Event Relations. In *Proceedings of the Twenty-Third International FLAIRS Conference*.

Leonard Talmy. 1985. Force dynamics in language and thought. *Chicago Linguistic Society*, 21:293–337.

Leonard Talmy. 1988. Force dynamics in language and cognition. *Cognitive science*, 12(1):49–100.

The PDTB Research Group. 2008. The PDTB 2.0. Annotation Manual. Technical Report IRCS-08-01, Institute for Research in Cognitive Science, University of Pennsylvania.

Sara Tonelli, Rachele Sprugnoli, and Manuela Speranza. 2014. Newsreader guidelines for annotation at document level. Technical Report NWR-2014-2, Fondazione Bruno Kessler. http://www.newsreader-project.eu/files/2013/01/NWR-2014-2.pdf.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Proceedings of the Seventh International Workshop on Semantic Evaluation*, SemEval '13, pages 1–9, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

H. van de Koot and A. Neeleman, 2012. *The Theta System: Argument Structure at the Interface*, chapter The Linguistic Expression of Causation, pages 20 – 51. Oxford University Press: Oxford.

Phillip Wolff and Grace Song. 2003. Models of causation and the semantics of causal verbs. *Cognitive Psychology*, 47(3):276–332.

Phillip Wolff, Bianca Klettke, Tatyana Ventura, and Grace Song. 2005. Expressing causation in english and other languages. *Categorization inside and outside the laboratory: Essays in honor of Douglas L. Medin*, pages 29–48.

Phillip Wolff. 2007. Representing causation. *Journal of experimental psychology: General*, 136(1):82.

Exploring Fine-grained Entity Type Constraints for Distantly Supervised Relation Extraction

Yang Liu Kang Liu Liheng Xu Jun Zhao
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
Zhongguancun East Road #95, Beijing 100190, China
{yang.liu, kliu, lhxu, jzhao}@nlpr.ia.ac.cn

Abstract

Distantly supervised relation extraction, which can automatically generate training data by aligning facts in the existing knowledge bases to text, has gained much attention. Previous work used conjunction features with coarse entity types consisting of only four types to train their models. Entity types are important indicators for a specific relation, for example, if the types of two entities are "PERSON" and "FILM" respectively, then there is more likely a "DirectorOf" relation between the two entities. However, the coarse entity types are not sufficient to capture the constraints of a relation between entities. In this paper, we propose a novel method to explore fine-grained entity type constraints, and we study a series of methods to integrate the constraints with the relation extracting model. Experimental results show that our methods achieve better precision/recall curves in sentential extraction with smoother curves in aggregated extraction which mean more stable models.

1 Introduction

Relation Extraction is the task of extracting semantic relations between a pair of entities from sentences containing them. It can potentially benefit many applications, such as knowledge base construction, question answering (Ravichandran and Hovy, 2002), textual entailment (Szpektor et al., 2005), etc. Traditional supervised approaches for relation extraction (Zhou et al., 2005)(Zhou et al., 2007) need to manually label training data, which is expensive and limits the ability to scale up. Due to the shortcoming of supervised approaches mentioned above, recently, a more promising approach named distantly supervised relation extraction (or distant supervision for relation extraction) (Mintz et al., 2009) has become popular. Instead of manual labeling, it automatically generates training data by aligning facts in existing knowledge bases to text.

However, the paradigm of distant supervision also causes new problems of noisy training data both in positive training instances and negative training instances. To overcome the false positive problem caused by the distant supervision assumption, researches in (Riedel et al., 2010)(Hoffmann et al., 2011)(Surdeanu et al., 2012) proposed multi-instance models to model noisy positive training data, where they assumed that at least one sentence in those containing an entity pair is truly positive. Takamatsu et al. (Takamatsu et al., 2012) claimed that the at-least-one assumption in multi-instance models would fail when there was only one sentence containing both entities. They proposed a method to learn and filter noisy pattern features from training instances to overcome the false positive problem. Researchers (Xu et al., 2013)(Zhang et al., 2013)(Ritter and Etzioni, 2013) tried to address the problem of false negative training data caused by the incomplete knowledge base. Xu el al. (Xu et al., 2013) used the pseudo-relevance feedback method trying to find out the false negative instances and add them into positive training instances. Zhang et al. (Zhang et al., 2013) employed some rules to select negative training instances carefully, hoping not to include the false negative instances. And Ritter et al. (Ritter and Etzioni, 2013) used hidden variables to model the missing data in databases based on a graphical model. The training data generation process for all the above work is under the framework of (Mintz et al., 2009),

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2107–2116, Dublin, Ireland, August 23-29 2014.

one important step of which is to recognize entity mentions from text and assign them entity types which are used to compose features for training the model. The entity types they used are very coarse only consisting of four categories (PERSON, ORGANIZATION, LOCATION, NONE). We argue that the coarse entity types are not sufficient to indicate relations.

A specific relation constrains the entity types of its two entities. For instance, the *SingerOf* relation limits the entity type of its first entity as PERSON or more fine-grained ARTIST, and the entity type of its second entity as ART or more fine-grained MUSIC. Therefore, when extracting a relation instance, the entity types of its two entities are important indicators for a specific relation. Previous work used conjunction features (Details in Section 3.3) by combining the coarse entity types of entity mentions with its contextual lexical and syntactic features. However, the conjunction features may fail to distinguish the relations. For example, the following two sentences contain two relation instances, one is *DirectorOf(Ang Lee, Life of Pi)*, and the other is *AuthorOf(George R.R. Martin, A Song of Ice and Fire)*.

1. **Ang Lee**'s **Life of Pi** surprised many by scoring a leading four Oscars on Sunday night...
2. Westeros is the premiere fansite for **George R.R. Martin**'s **A Song of Ice and Fire**.

Only using the above conjunction features, we cannot tell the difference between the two entity pairs, and are probable to incorrectly classify them as the same relation. By contrast, if we can assign each entity with fine-grained entity types, for example, *Ang Lee* as the entity type ARTIST and *George R.R. Martin* as AUTHOR, we may succeed in classifying the two entity pairs correctly.

To achieve the goal mentioned above, there are mainly three challenges: (1) how to define the fine-grained type set; (2) how to assign the types to entity mentions; (3) how to integrate the fine-grained entity type constraints with the relation extracting model. To address these challenges, in this paper, we propose a novel approach to explore the fine-grained entity type constraints for distantly supervised relation extraction. First, we use the types defined in (Ling and Weld, 2012) stemmed from Freebase[1] as the fine-grained entity type set (introduced in Section 3.1). Second, we leverage Web knowledge to train a fine-grained entity type classifier and predict entity types for each entity mention. Third, we study several methods to integrate the type constraints with an existing system *MULTIR*, a multi-instance multi-label model in (Hoffmann et al., 2011), to train the extractor.

In summary, the contribution of this paper can be concluded as follows.

(a) We explore the effect of fine-grained entity type constraints on distantly supervised relation extraction. A novel method is proposed to leverage Web knowledge to automatically train a fine-grained entity type classifier, which is used to predict the fine-grained types of each entity mention.

(b) We study a series of methods for integrating the fine-grained entity type constraints with the extracting model and compare their performance with different parameter settings.

(c) We conduct experiments to demonstrate the effects of the newly exploited fine-grained entity type constraints. It shows that our method achieves a much better precision/recall curves over the baseline system in sentential extraction, and improves the performance with a smoother precision/recall curve in aggregated extraction, which means a more stable model.

2 Distant Supervision for Relation Extraction

We define a relation instance (or a fact), which means a binary relation, as $r(e_1, e_2)$. r is the relation, and e_1 and e_2 mean the two entities in the relation instance, for example, $BornIn(Yao\ Ming, Shanghai)$. Distant supervision supplies a method to automatically generate training data. In this part, we will introduce the general steps in distant supervision for relation extraction. First, we define the notations we use. Σ denotes sentences comprising the corpus, E denotes entity mentions in the corpus which are consecutive words with the same named entity tags assigned by an NER system, Δ denotes the facts (or relation instances) in the existing knowledge base. R denotes the relations in Δ.

[1] http://www.freebase.com/

person	doctor	**organization**	terrorist_organization		
actor	engineer	airline	government_agency		
architect	monarch	company	government		
artist	musician	educational_institution	political_party		
athlete	politician	fraternity_sorority	educational_department		
author	religious_leader	sports_league	military		
coach	soldier	sports_team	news_agency		
director	terrorist				
location	body_of_water	**product**	camera	**art**	written_work
city	island	engine	mobile_phone	film	newspaper
country	mountain	airplane	computer	play	music
county	glacier	car	software	**event**	military_conflict
province	astral_body	ship	game	attack	natural_disaster
railway	cemetery	spacecraft	instrument	election	sports_event
road	park	train	weapon	protest	terrorist_attack
bridge					
building	time	chemical_thing	website		
airport	color	biological_thing	broadcast_network		
dam	award	medical_treatment	broadcast_program		
hospital	educational_degree	disease	tv_channel		
hotel	title	symptom	currency		
library	law	drug	stock_exchange		
power_station	ethnicity	body_part	algorithm		
restaurant	language	living_thing	programming_language		
sports_facility	religion	animal	transit_system		
theater	god	food	transit_line		

Figure 1: Fine-grained entity type set.

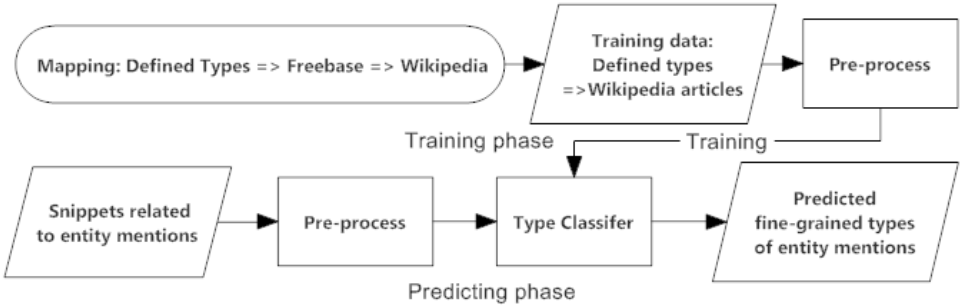


Figure 2: Framework of fine-grained entity type classifier.

To generate training data, we align pairs of entity mentions in the same sentence with Δ. The aligned entity mentions E_{train} and their sentences Σ_{train} along with R_{train} are used as training data. Features are extracted from them to train the relation extracting model.

To predict the unknown data for extracting new relation instances, we input pairs of entity mentions $E_{predict}$ and the sentences containing them $\Sigma_{preidct}$ into the trained extracting model for extracting new relation instances.

3 Fine-grained Entity Type Constraints

Entity mentions in sentences are considered consecutive words with the same entity types (Section 2). The entity types are part of the lexical and syntactic features(Mintz et al., 2009), and the feature setting is followed by other related work. Their entity types are assigned by an NER system and consist of four categories (PERSON, ORGANIZATION, LOCATION, NONE). The types of entity mentions in a relation are important indicators for the very type of relation. However, the coarse (only four types) entity types may not capture sufficient constraints to distinguish a relation. In this section, we explore fine-grained entity type constraints and study different methods to integrate them with the extracting model.

This section first introduces the fine-grained entity type set(Section 3.1), and then describes our method which leverages Web knowledge to train the fine grained entity type classifier and assign entity mentions with the fine-grained entity types (Section 3.2). At last, we illustrate methods to integrate fine-grained entity type constraints with the relation extracting model.

Entity pair	[Hank Ratner], [Cablevision]					
Sentence	Cablevision's $600 million offer came in the form of a letter to Peter S.Kalikow, chairman of the M.T.A., from the Garden's vice chairman, Hank Ratner.					
Conjunction	Reverse	Left	NE1	Middle	NE2	Right
Feature examples	False		PER		ORG	
	False	Hank[NMOD]	PER	[NMOD]chairman ... offer[SBJ]	ORG	
	True	B_-1	ORG	POS $ ... NN NN,	PER	.B_1

Table 1: Examples of conjunction features.

3.1 Fine-grained Entity Types

Figure 1 is the type set we use. It was introduced in (Ling and Weld, 2012) and was derived from Freebase types. The bold types in each small box of Figure 1 are upper-class types for others in that small box. For example, $/actor$ is a lower-class type of $/person$ which is denoted as $/person/actor$. And $/person$ and $/person/actor$ coexist in the type set.

3.2 Fine-grained Entity Type Classifier

In this section, we describe our method that leverages Web knowledge to train a fine-grained entity type classifier and predict entity types of each entity mention. Its architecture is shown in Figure 2.

3.2.1 Training

The training data are obtained from Wikipedia. Because the defined fine-grained types are tailored based on Freebase types, we can find the mappings between the two type sets, for example, $/person/doctor$ maps to two Freebase types $/medicine/physician$ and $/medicine/surgeon$. And Freebase WEX[2] supplies a mapping between Freebase types to Wikipedia articles. As a result, we can map Wikipedia articles to defined fine-grained types.

Based on the mappings, we obtain Wikipedia articles for each type as training data and negative training examples are sampled from articles not contained in the mappings. We preprocess the articles by: stop words filtering, stemming, and term frequency filtering and use a maxent model to train the classifier.

3.2.2 Predicting

To predict types of each entity mention, we first use search engines to expand entity mentions. Specifically, each entity mention is used as a query sent to the search engine[3]. Titles and descriptions of top k returned snippets are selected (We keep the top 20 in the experiments). The obtained text are preprocessed with the same method as training examples. Then we use the trained fine-grained entity type classifier to predict the types of each entity mention.

After predicting, we obtain a ranked list of types for each entity mention, which are ranked by the predicting scores.

3.3 Integrating Fine-grained Entity Type Constraints into the Extracting Model

This section introduces our methods to integrate the fine-grained entity type constraints with the extracting model. First of all, we briefly review the features used in previous models which derived from (Mintz et al., 2009) and (Riedel et al., 2010). Their features mainly comprise two types: lexical features (POS tags, words and entity types) and syntactic features (dependency parsing tags, words and entity types). Each feature is a conjunction with several parts: entity types of two entity mentions, the left context window of the first entity mention, the right context window of the second entity mention and the part between them (the window contains none or one or two words). Table 1 shows an example of the conjunction features.

[2]http://wiki.freebase.com/wiki/WEX

[3]We use Bing search API. http://datamarket.azure.com/dataset/bing/search

To integrate the exploited fine-grained entity type constraints with the extracting model, we proposed three methods (substitution, augment and selection) to make the type constraints take effects.

3.3.1 Substitution Method

In this method, we substitute coarse entity types of the features with the entity mentions' fine-grained types, and use the new features to train the model. Instead of substituting directly, an entity mention is first represented by its fine-grained types and the upper-class of the fine-grained type, for example, $/person/politician$ derives two types $/person$ and $/person/politician$ itself. The reason is that the extracting model can benefit from the related types like the upper-class types. And then we use the obtained entity types to substitute the old coarse entity types as new features greedily, which means that all the possible combinations of types between the entity pair are considered. For example, "$Barack\ Obama$" has the fine-grained type $/person/politician$ and his birth place "$Hawaii$" has the type $/location/island$, then there are 4 combinations between the two entities, they are ($/person$, $/location$), ($/person$, $/location/island$), ($/person/politician$, $/location$) and ($/person/politician$, $/location$).

3.3.2 Augment Method

In this method, we generate new features by substituting the coarse entity types with predicted fine-grained types, and expand the old features with new features. Different from the substitution method, we do not add the upper-class types, for that we think the coarse types in old features have the same effect. In this method, we use the fine-grained constraints as a complementary.

3.3.3 Selection Method

The selection method is similar to the augment method. The difference is that we do not expand all old features with new features. We select some of them to expand. The reason is that some of the conjunction features are of high-precision themselves, it can clearly indicate the relations with its left, middle and right parts, even without the entity types (informative ones). If we expand these features, it may cause more noisy features. So we expect to only expand the ones that lack of the indicating abilities (non-informative ones). In this paper, we employ a simple method to distinguish between the informative ones and non-informative ones by the length of the features, which means that the longer is more informative than the shorter. In our experiments, the length threshold is set as 20.

In the predicting phase (Section 3.2), we obtain a ranked type list for each entity mention. The top list types are considered in our methods. Experiments in Section 4.3 are conducted on top k $\{k \in 1, 2, 3\}$ type/types in the obtained ranked list. And they are combined with a greedy method similar to that in the substitution method explained above.

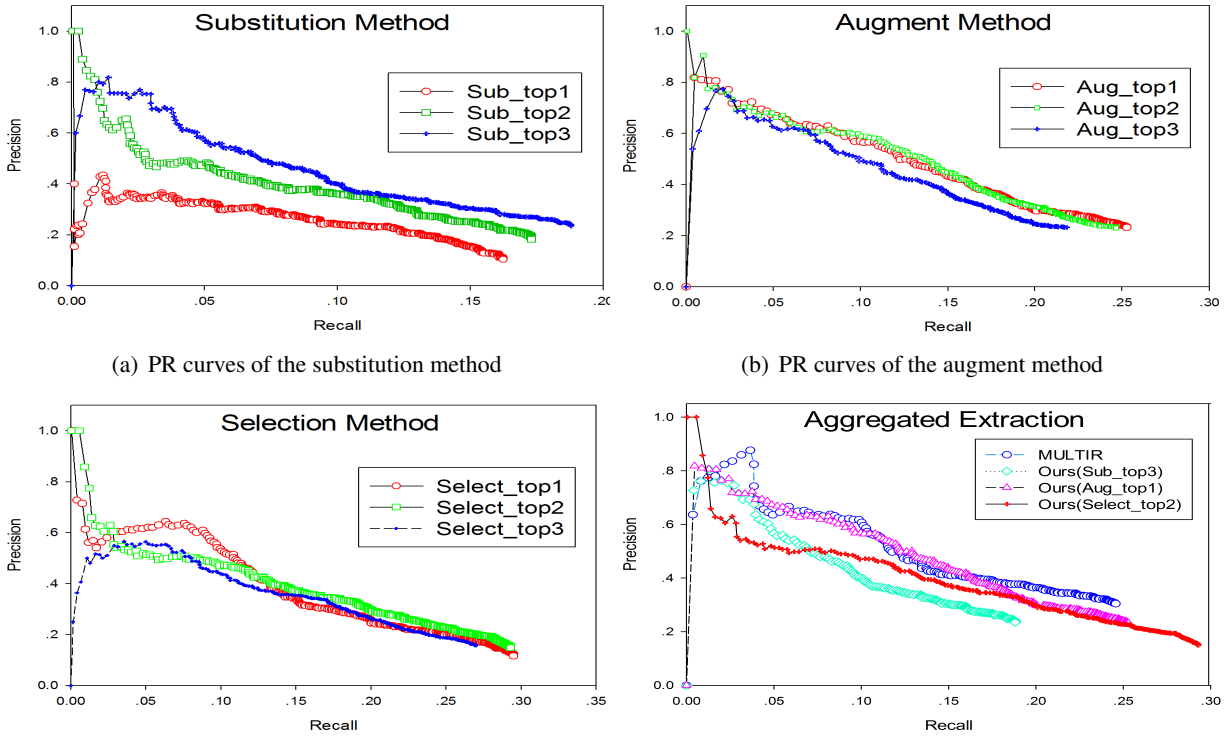
4 Experiments

4.1 Settings

We use the same data sets as (Riedel et al., 2010) and (Hoffmann et al., 2011), where NYTimes sentences in the years 2005-2006 are used as training corpus Σ_{train} for distant supervision and sentences in 2007 are used as testing corpus $\Sigma_{predict}$. The data was first tagged with an NER system (Finkel et al., 2005) and consecutive words with the same tag are extracted as entity mentions. And then, entity mentions E_{train} in training corpus are aligned to facts Δ in Freebase as training examples to train the models.

We integrate our fine-grained entity type constraint with MULTIR, an existing multi-instance multi-label extracting model in (Hoffmann et al., 2011). Following their setttings, we conduct experiments on aggregated extraction and sentential extraction to show the effect of fine-grained entity type constraints.

- **Aggregated extraction**: Aggregated extraction is corpus-level extraction. When given an entity pair, it predicts its relation types based on the whole corpus. After extraction, the precision and recall are computed by comparing the results with facts in Freebase. The evaluation underestimates the accuracy because there may be correct facts in the extracted results but not existing in Freebase, these facts are labeled as incorrect by mistake here. Because aggregated extraction is an automatic evaluation, it is used to tune parameters like held-out evaluation in (Mintz et al., 2009).



(a) PR curves of the substitution method

(b) PR curves of the augment method

(c) PR curves of the selection method

(d) Comparison with other methods

Figure 3: Precision-recall (PR) curves of the aggregated extraction.

- **Sentential extraction**: Sentential extraction predicts an entity pair only based on a specified sentence containing the pair of entities. We use manually labeled data in (Hoffmann et al., 2011) as benchmark. The data consist of 1,000 sentences and are sampled from the results their system outputs and sentences aligned with facts in Freebase. As they stated in their paper, these results provide a good approximation to the true precision but can overestimate the actual recall.

4.2 Experimental Results

In aggregated extraction, we first evaluate the three type-constraint integration methods (substitution, augment and selection) with the top k $\{k \in 1, 2, 3\}$ type/types (Section 3.3). And then, we compare the best parameter setting methods with previous work. In sentential extraction, we compare methods tuned in aggregated extraction with $MULTIR$.

4.2.1 Aggregated Extraction

Figure 3 shows the precision-recall (PR) curves of the aggregated extraction. In it, Sub_topk $\{k \in 1, 2, 3\}$ means using the substitution method (Section 3.3) with top k fine-grained entities types returned by the type classifier in Section 3.2. Correspondingly, Aug_topk is for the augment method and $Select\_topk$ is for the selection method.

Figure 3(a) shows that Sub_top3 outperforms the other two settings of k in the substitution method, it seems that more fine-grained types produce better curves. In Figure 3(b), Aug_top1 and Aug_top2 achieve similar performances. However, when adding one more type with $k = 3$, we obtain a lower curve, which contradicts the trend showed in the curves of the substitution method (Figure 3(a)). Figure 3(c) shows the PR curves of three selection methods, $Select\_top1$ has a better performance at the beginning. Then $Select\_top2$ exceeds it a bit consistently.

In Figure 3(d), we demonstrate the comparison of best tuned methods above with previous work. They are Sub_top3, Aug_top1 and $Select\_top2$. From Figure 3(d), it shows that, among the three of our methods, Aug_top1 achieves better precisions along the PR curves, and $Select\_top2$ reaches the best

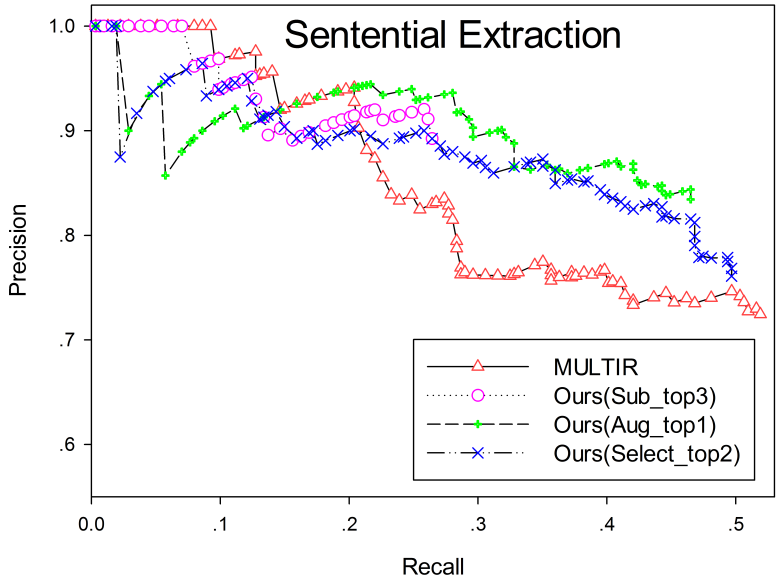


Figure 4: Comparison with MULTIR

recall at the highest recall point. Comparing to other methods, the PR curve of Aug_top1 reaches a higher recall with 29.3% at the highest recall point than *MULTIR* (24.5%). $Select\_top2$ achieves 29.3% at the highest recall point, best among all methods. And by integrating the fine-grained entity type constraints, they improve the PR curve of *MULTIR* with a more smoother curve without most of the depressions seen in *MULTIR*. As stated in (Hoffmann et al., 2011), the smoother curve indicated a more stable model.

4.2.2 Sentential Extraction

Figure 4 shows the precision-recall (PR) curves of the sentential extraction. In the evaluation, we compare the three best integration methods tuned in aggregated extraction with original *MULTIR*. Among our three method, Aug_top1 outperforms in precision and achieves a better curve in general among the three methods, however, $Select\_top2$ gains a better recall at the end. Sub_top3 has the worst recall. In general, our methods have much better precisions than *MULTIR*. Aug_top1 and $Select\_top2$ achieve better curves than *MULTIR*. Since the evaluation of sentential extraction is a good approximation of precision, it implies that the proposed methods are effective.

4.2.3 Analysis

On one hand, among the three proposed integration methods, generally, the augment method and selection method get better performance. The reason is that substitution method uses predicted fine-grained entity types to replace the old coarse features in the conjunction features completely, and the conjunction features are sensitive to entity types for different entity types indicate different conjunction features, as a result, if we can not promise a good accuracy in the type classification which is hard to achieve in classifying hundreds of fine-grained types, the performance will be badly influenced. Different from the substitution method, augment method and selection method keep the old features with coarse features, they use the features with fine-grained entity type constraints as extra information to help the extraction and achieve better results.

On the other hand, comparing to other methods, by integration the exploited fine-grained entity type constraints, our methods achieve improvements in both aggregated and sentential extraction. It proves that the fine-grained entity type constraints we exploit are effective, and our proposed integration methods succeed in integrating the constraints into the extracting model. Our augment method outperforms *MULTIR* in precision along the PR curves in sentential extraction and improve it performance with a more smoother PR curve in aggregated extraction, which indicates a more stable model. Moreover, the method gets a better recall. And our selection method consistently outperforms *MULTIR* in sentential

	k=1	k=2	k=3
Recall@k	0.596	0.740	0.806

Table 2: Evaluation of the fine-grained type classifier.

extraction. In aggregated extraction, it also achieves a smoother curve and an impressive promotion at the highest recall point. Since the evaluation of aggregated extraction only considers the facts existing in Freebase which may incorrectly label the right extracting results and underestimate the true precision, and based on its better performance of precision in sentential extraction, we consider it is a more promising method. This paper only employs very naive method to select the non-informative features by its length (Section 3.3.3), a more effective selecting method may lead further improvements.

4.3 Performance of Entity Type Classifier

We evaluate the performance of the fine-grained entity type classifier (Section 3.2). In section 3.2, we sample the training examples from a collection of Wikipeida articles mapped with the fine-grained types. To generate test entity mentions, we first remove the sampled training articles from the collection, and then sample the articles from it, where the titles of sampled articles are used as the test entity mentions (we sample 12,000 test entity mentions) and their mapped fine-grained types are used as benchmark. After that, the predicting method in Section 3.2.2 is used to expand mentions and predict the types of each test entity mention. After predicting, we obtain a ranked list of types for each test entity mention.

To evaluate, we define a notation of *Hit@k*, which equals 1 if the true type of an entity mention is hit in the top k predicted types, otherwise equals 0. And then we evaluate it by the $Recall@k$ defined bellow.

$$Recall@k = \frac{\Sigma_{i=1}^{12000} Hit@k_i}{12000} \quad (1)$$

In equation (1), i means the ith test entity mention. Table 2 shows the results for the top 3 predicted types.

5 Related Work

Distant supervision (also known as weak supervision or self supervision) is used to a broad class of methods in information extraction which aims to automatically generate labeled data by aligning with data in knowledge bases. It is introduced by Craven and Kumlien (Craven et al., 1999) who used the Yeast Protein Database to generate labeled data and trained a naive-Bayes extractor. Bellare and McCallum (Bellare and McCallum, 2007) used BibTex records as the source of distant supervision. The KYLIN system in (Wu and Weld, 2007) used article titles and infoboxes of Wikipedia to label sentences and trained a CRF extractor aiming to generate infoboxes automatically. The Open IE systems TEXTRUNNER (Yates et al., 2007) and WOE (Wu and Weld, 2010) trained their extractors with the automatic labeled data from Penn Treebank and Wikipedia infoboxes respectively.

Mintz (Mintz et al., 2009) first introduced their work that performed distant supervision for relation extraction. It used Freebase as the knowledge base to align sentences in Wikipedia as training data and trained a logistic regression classifier to extract relations between entities.Distant supervision supplied a method to generate training data automatically, however it also bring the problem of noisy labeling. After their work, a variety of methods focused to solve this problem. Riedel (Riedel et al., 2010) proposed a multi-instance model to model the false positive noise in training data with the assumption that at least one of the labeled sentences truly expressed their relation. After their work, Hoffmann (Hoffmann et al., 2011) and Surdeanu (Surdeanu et al., 2012) tried to not only model the noisy training data, but also overcame the problem of multi-label where two entities may exist more than one relation, they proposed graphic models as kinds of multi-instance multi-label learning methods and made improvements over previous work. The at-least-one assumption would fail when encountering entity pairs with only one aligned sentence. Takamatsu (Takamatsu et al., 2012) employed an alternative approach without the mentioned assumptions. Their work predicted negative patterns using a generative model and remove labeled data containing negative patterns to reducing noise in labeled data.

Besides the problem of false positive training examples caused by distant supervision. There were a bunch of researches trying to solve the problem of false negative training examples caused by incomplete knowledge bases. Zhang (Zhang et al., 2013) made heuristic rules to filter the false negative training examples. And Xu (Xu et al., 2013) tried to overcom this problem by pseudo-relevance feedback. Min (Min et al., 2013) improved MIML in (Surdeanu et al., 2012) by adding a new layer in their 3-layer graphic model to model the incomplete knowledge base. Ritter (Ritter and Etzioni, 2013) employed similar intuition with (Xu et al., 2013) that they thought rear entities missing in the database would be often mentioned in the text. They proposed a latent-variable approach to model it and showed its improvement over aggregate and sentential extraction.

6 Conclusion

In this paper, we propose a novel approach to explore the fine-grained entity type constraints for distantly supervised relation extraction. We leverage Web knowledge to automatically train a fine-grained entity type classifier and predict entity types of each entity mention. And we study a series of methods to integrate the type constraints with a relation extraction model. At last, thorough experiments are conducted. The experimental results imply our methods are effective with better precision/recall curves in sentential extraction and smoother precision/recall curves in aggregated extraction, which indicate more stable models.

In the future we hope to explore more details of integration methods that integrates fine-grained entity type constraints with relation extraction models, especially the selection integration method. We consider that a more effective method to distinguish between the informative and non-informative features will lead more improvements.

Acknowledgements

This work was sponsored by the National Basic Research Program of China (No. 2014CB340503) and the National Natural Science Foundation of China (No. 61202329). This work was supported in part by Noahs Ark Lab of Huawei Tech. Co. Ltd.

References

Kedar Bellare and Andrew McCallum. 2007. Learning extractors from unlabeled text using relevant databases. In *Sixth International Workshop on Information Integration on the Web*.

Mark Craven, Johan Kumlien, et al. 1999. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 77–86. Heidelberg, Germany.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 541–550.

Xiao Ling and DS Weld. 2012. Fine-Grained Entity Recognition. In *AAAI*.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *Proceedings of NAACL-HLT*, pages 777–782.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 41–47. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Alan Ritter and Oren Etzioni. 2013. Modeling Missing Data in Distant Supervision for Information Extraction. *Transactions of the Association for Computational Linguistics*, 1:367–378.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465. Association for Computational Linguistics.

Idan Szpektor, Hristo Tanev, Ido Dagan, Bonaventura Coppola, et al. 2005. *Scaling Web-based aquisition of entailment relations*. Ph.D. thesis, Tel Aviv University.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 721–729. Association for Computational Linguistics.

Fei Wu and Daniel S Weld. 2007. Autonomously semantifying wikipedia. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 41–50. ACM.

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

W Xu, RH Le Zhao, and R Grishman. 2013. Filling Knowledge Base Gaps for Distant Supervision of Relation Extraction. *Proceedings of Association for Computational Linguistics*.

Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 25–26. Association for Computational Linguistics.

Xingxing Zhang, jianwen Zhang, Junyu Zeng, Jun Yan, Zheng Chen, and Zuifang Sui. 2013. Towards Accurate Distant Supervision for Relational Facts Extraction. In *Proceedings of Association for Computational Linguistics*.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 427–434. Association for Computational Linguistics.

GuoDong Zhou, Min Zhang, Dong Hong Ji, and Qiaoming Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Using Collections of Human Language Intuitions to Measure Corpus Representativeness

Reinhard Rapp
Aix-Marseille Université
Laboratoire d'Informatique Fondamentale
163 Avenue de Luminy, 13288 Marseille, France
reinhardrapp@gmx.de

Abstract

In corpus linguistics there have been numerous attempts to compile balanced corpora, resulting in text collections such as the Brown Corpus or the British National Corpus. These corpora are meant to reflect the average language use a native speaker typically encounters. But is it possible to measure in how far these efforts were successful? Assuming that humans' language intuitions are based on our brain's capability to statistically analyze perceived language and to memorize these statistics, we suggest a method for measuring corpus representativeness which compares corpus statistics to three types of human language intuitions as collected from test persons: Word familiarity, word association, and word relatedness. We compute a representativeness score for a corpus by extracting word frequency, word co-occurrence, and contextual statistics from it and by comparing these statistics to the human data. The higher the similarity, the more representative the corpus should be for the language environments of the test persons. Our findings confirm the expectation that corpus size and corpus balancing matter.

1 Introduction

Balanced corpora, i.e. corpora consisting of a carefully sampled mix of texts, have often been considered important for providing a standard of average language use. Well known examples of such corpora include the *Brown Corpus* (Francis & Kuçera, 1989) and the *British National Corpus* (Burnard & Aston, 1998). But to obtain a balance many decisions concerning the corpus design have to be made. Biber (1993) mentions, among other things, that it has to be decided for what target population a corpus is meant to be representative, that estimates concerning the quantities of various text types are required, and that decisions with regard to the number of individual text samples and their sizes have to be made.

However, there is no easy and well established way to verify the success of these measures. Current suggestions include, for example, to consider a corpus as representative if it is not dominated by sublanguage (Temnikova et al., 2014), or to more or less give up on the concept of representativeness and to concentrate on considering the suitability of a corpus for particular tasks. Saldanha (2009) comes to the conclusion that "The problem with making representativeness the defining characteristic of a corpus is that it is very difficult to evaluate."

Our goal here is to make an attempt to measure corpus representativeness in a standardized way, thereby avoiding to observe test persons' average language input as this would not be very practical. Our starting point is that a representative corpus should reflect as well as possible average language use as encountered by native speakers. We also assume that human language acquisition is essentially corpus-based (Rapp, 2011). This implies the following: The human brain analyzes particular statistical properties of perceived language and memorizes them. During language production these properties

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2117–2128, Dublin, Ireland, August 23-29 2014.

are reproduced. It has been shown that in certain test situations it is possible to isolate intuitions related to some specific statistics. These include the following three which we will utilize for measuring corpus representativeness: Word frequency, word co-occurrence, and common context of words. In terms of human language intuitions, these three statistical properties relate to word familiarities, word associations, and word relatedness.

What we suggest is to extract data relating to these three types of statistical properties from a corpus and to compare it to the respective experimental data as obtained from test persons. The higher the average agreement, the more representative the corpus should be for the language environment of the test persons.[1]

Related work has been conducted by Brisbaert & New (2009), which is mentioned in section 2.1, and in our own previous studies (Rapp, 2014a and Rapp, 2014c), of which the current work is an extension. A nice summary of how to measure corpus representativeness through psycholinguistic measures is provided in a presentation by Francom & Ussishkin (2011). Gries (2010), though in a slightly different context, emphasizes the need of external validation: "For corpus linguists, that means that our measures must be validated against corpus-external evidence because, strictly speaking, as long as we corpus linguists do not show that our dispersions and adjusted frequencies correspond to something outside of our corpora, we have failed to provide the most elementary aspect of a new measure – its validation."

The remainder of this paper is structured as follows: We first describe the experimental data used, i.e. the familiarity norms, the association norms, and the synonym data (describing word relatedness). Next we present the algorithms used to extract the corresponding statistics from the corpora. By comparing the human and the corpus-derived data, we introduce three quantitative measures of corpus representativeness, which we subsequently combine. The paper concludes with a discussion and an outlook on future work.

2 Human language intuitions

2.1 Word familiarities

Psychologists have collected word familiarity ratings from test persons. For this purpose, the subjects were asked to come up with subjective familiarities for given words. Usually a scale between 1 and 7 was used, whereby 1 means unfamiliar and 7 means very familiar. The outcome of such experiments are the so-called familiarity norms, i.e. large tables listing the subjects' familiarity ratings. In the current work we used the familiarity data for 4920 words from an online version[2] of the *MRC Psycholinguistic Database* (Coltheart, 1981).

In previous studies (e.g. Rapp, 2005) it has been shown that there is a strong correlation between the human familiarity judgments and the log occurrence frequencies of the words in corpora. For illustration, Table 1 shows the top five most familiar words in the MRC database together with their frequencies in the Brown corpus and compares them to some of the least familiar words. As can be seen, the familiar words have consistently much higher corpus frequencies. To explain this finding, Rapp (2005) hypothesized that human familiarity ratings are based on the word frequencies as observed by the test persons in the language they perceive in everyday life.

However, if we assume that the familiarity norms reflect word frequencies in perceived language, then it should be possible to use them as a standard for measuring the frequency aspect of corpus representativeness. A corpus whose word frequencies are highly correlated to the familiarity norms is more likely to be a good surrogate for everyday language, although word frequency of course reflects only one of many properties of a corpus. Nevertheless, for a corpus to be representative, it is a necessary (though not sufficient) condition that its word frequencies are similar to those in everyday language.

[1] Let us mention that there is some analogy to automatic MT evaluation, namely when computing the BLEU score: There a machine translation is compared to a human translation (which is based on human intuitions) by identifying matches between n-grams of various lengths. Then a combined score is computed from the results obtained for each n-gram length.

[2] http://websites.psychology.uwa.edu.au/school/MRCDatabase/uwa_mrc.htm

We should mention that instead of using word familiarity data it is also possible to use reaction times as obtained in the word recognition task.[3] Brisbaert & New (2009) did so and related the reaction times to the corpus frequencies of the words for the purpose of measuring corpus representativeness. In essence, although they tested on other corpora, their findings seem to be similar to what we report here based on word familiarities.

FAMILIAR WORDS			UNFAMILIAR WORDS		
WORD	FAMILIARITY	BROWN FREQUENCY	WORD	FAMILIARITY	BROWN FREQUENCY
BREAKFAST	6.6	53	LOQUACITY	1.4	1
AFTERNOON	6.5	106	MIEN	1.4	1
CLOTHES	6.5	89	YUCCA	1.4	1
BEDROOM	6.5	52	BURGHER	1.3	1
DAD	6.5	15	PAEAN	1.3	2

Table 1: Words with high and low familiarity ratings in the MRC Psycholinguistic Database together with their frequency counts in the Brown Corpus (words with a corpus frequency of zero are not included).

2.2 Word associations

The second type of human intuitions to be considered are word associations as obtained from test persons. Such data has been collected from native speakers in large scale experiments, as exemplified in the *Edinburgh Associative Thesaurus* (*EAT*; Kiss et al., 1973) which is the largest classical collection of its kind. The EAT comprises the associative responses as requested from around 100 British students for each of 8400 stimulus words and is available online.[4]

To collect the data, the subjects were given questionnaires with lists of stimulus words, and were asked to write down for each stimulus word the spontaneous association which first came to mind. This leads to collections of associations, the so-called association norms, as exemplified in Table 2.

ABOVE	CONSTELLATİON	FEMİNİNE
below (59)	stars (39)	masculine (26)
high (4)	star (33)	girl (14)
over (4)	sky (5)	woman (8)
sky (4)	andromeda (2)	female (6)
all (3)	aquarius (2)	sex (3)
up (3)	plough (2)	beauty (2)
me (2)	aircraft (1)	bird (2)
under (2)	cancer (1)	girls (2)

Table 2: Top eight associations to three stimulus words as taken from the EAT. The numbers of subjects responding with the respective word are given in brackets.

2.3 Word relatedness

The third type of human intuitions which we consider concerns word relatedness. Landauer & Dumais (1997) introduced a dataset for testing semantic relatedness, namely the synonym portion of the *Test of English as a Foreign Language* (TOEFL). The TOEFL is an often obligatory test for non-native speakers of English who intend to study at a university with English as the teaching language. The data used by Landauer & Dumais had been acquired from the *Educational Testing Service* and comprises 80 test items. As summarized in Rapp (2009), each item consists of a problem word embedded

[3] In the so-called word recognition task test persons are presented strings of characters and their task is to decide whether or not a string matches an English word. It turns out that the average reaction time is inversely related to the familiarity of a word (i.e. the less familiar a word, the longer the reaction time).

[4] http://www.eat.rl.ac.uk/

in a sentence and four alternative words, from which the test taker is asked to choose the one with the most similar meaning to the problem word. For example, given the test sentence *"Both boats and trains are used for transporting the materials*" and the four alternative words *planes*, *ships*, *canoes*, and *railroads*, the subject would be expected to choose the word *ships*, which is supposed to be the one most similar to *boats*.

However, Landauer & Dumais (1997) did not use the test sentences. Instead, only the lists of problem words together with their alternatives were used. A system capable of computing word relatedness should be able to determine for each problem word the alternative word which comes closest in meaning.

Although the TOEFL dataset has been widely used (see e.g. the overview on related work on the ACL Wiki[5]), there are two disadvantages with it: A minor one is that it is not freely available on the web. A more severe one is that it is rather small: This means that statistical variation is strong (which will be illustrated in section 5.3), and that overfitting can easily happen. That is, a system trained on this data may not well perform on other data.

For this reason we decided to come up with a new dataset which avoids these problems. It is based on the index of Fernald's (1896) synonym and antonym dictionary as provided in the Project Gutenberg version.[6] This index lists in alphabetical order English words together with their synonyms. As in the dictionary there is no indication as to the quality of a synonym, in order to avoid arbitrary selections, from this list we removed all words for which several synonyms were listed in the index. In a semi-automatic way, we also removed a number of other items, e.g. those containing multiword units or numbers. As a result, we obtained a list of 4050 words together with their synonyms.

To obtain a dataset analogous to the TOEFL synonym set, we required three alternative words for each item. We could have used random words e.g. taken from the vocabulary of the British National Corpus (BNC). However, as the BNC is from a much later time period, this might have introduced a systematic bias. So we thought we should better use the words from the synonym dictionary itself. Note that the synonyms corresponding to the 4050 words represent a much smaller vocabulary as many of the synonyms are synonyms for several words. For this reason, we used the headwords themselves and applied the following procedure to generate the alternative words from them:

1) We sorted our list of items according to the synonyms in alphabetical order.
2) As the first column of alternative words, we used the given words but shifted them by 1000 positions, i.e. positions 1 to 3050 were matched with 1001 to 4050, and positions 3051 to 4050 where matched with 1 to 1000.
3) Analogous for the second column of alternative words, but here we shifted by 2000 positions.
4) Same for the third column of alternative words, but here we shifted by 3000 positions.

Word	Synonym	Alternative Words		
abandoned	addicted	rescind	bliss	receipts
abdicate	abandon	conflict	indubitable	archaic
aberration	insanity	rational	meliorate	assured
abetter	accessory	carnal	amicable	urbane
abettor	accessory	imbruted	brotherly	policy
abhorrence	abomination	kindliness	supposition	resignation
abiding	permanent	remain	life	stanch
ability	power	chimerical	frontier	diet
abject	pitiful	despotic	blanch	fray
abjure	abandon	contest	overt	disused

Table 3: Ten entries from the synonym dataset derived from Fernald (1896).

[5] http://aclweb.org/aclwiki/index.php?title=TOEFL_Synonym_Questions_(State_of_the_art)
[6] http://www.gutenberg.org/files/28900/28900-h/28900-h.htm

To give an impression of the dataset, its alphabetically first ten entries are shown in Table 3. Let us now quickly discuss some properties of the new dataset: The pros are that it is about 50 times larger than the TOEFL dataset and that it can be freely distributed. The cons are that it is based on somewhat outdated language (the dictionary was published in 1896) and that the alternative words were not carefully selected but generated in a somewhat arbitrary fashion. Also, it is not known how test persons would perform on this dataset, whereas for the TOEFL dataset human performance is known at least for some test takers, i.e. non-native speakers of English. A commonality between both datasets is that the synonyms were produced by experts, i.e. reflect the experts' language intuitions.

3 Corpora

As in previous work (Rapp, 2014a) our corpus representativeness measure is to be applied to a number of well known corpora. These are:

1) Brown Corpus (balanced corpus of 1 million words; Francis & Kuçera, 1989)
2) British National Corpus (BNC; balanced corpus of 100 million words; Burnard & Aston, 1998)
3) English Wikipedia (300 million words of encyclopaedic texts)[7]
4) ukWaC (British English web corpus of 2 billion words)[8]
5) English Gigaword Corpus 4th edition (4 billion words of newswire text)[9]

Both the MRC familiarity norms and the EAT do not distinguish between uppercase and lowercase characters. For this reason, we also did not make such a distinction and, in a pre-processing step, converted all corpora as well as the human data to lowercase only.

For the results presented later we had to measure the size of our corpora and also of partial corpora. We do this by counting the number of running words. Hereby, to avoid language specific sophistications, we count as a word any string which is delimited by either white space (blanks, tabulator, new line) or by transitions between alpha and non-alpha characters.[10]

4 Procedure

4.1 Corpus statistics concerning word familiarities (statistics of order zero)

In the case of word familiarities the statistics extracted from the corpora are the log frequencies of the words. The MRC database contains familiarities for 4920 words. As just two of them are multiword units, we considered this an inconsistency and removed them, so that 4918 words remained.

Word	Word frequency in the BNC	Word familiarity in the MRC database
a	2247100	632
abandon	1316	510
abandonment	500	359
abasement	20	226
abatement	137	294
abbess	57	187
abdication	124	284
abdomen	303	426
abduction	230	413
aberration	149	208

Table 4: BNC frequencies and MRC familiarities for the (alphabetically) first ten words covered in the familiarity norms of the MRC database.

[7] We use the English part of the Wikipedia XML Corpus (Denoyer & Gallinary, 2006). Although this is considerably smaller than current versions, it has the advantage that it is an offline copy so that our results can be replicated.
[8] http://wacky.sslmit.unibo.it/doku.php?id=corpora
[9] http://catalog.ldc.upenn.edu/LDC2009T13
[10] Alternatively, it would also be possible to simply count the number of characters for measuring corpus size (though this seems less customary). But word segmentation is required later on anyway (for computing the representativenesss scores).

The two types of data, namely the word familiarities from the MRC database and the word frequencies as extracted from one of the corpora, were merged as exemplified in Table 4 for the case of the BNC. Note that although the test subjects' familiarity judgements were originally on a scale between 1 (not familiar) and 7 (highly familiar), to avoid decimal numbers when averaging results, all ratings were multiplied by 100. Computing corpus representativeness now simply involves taking the logarithm of the frequencies in column 2, and then computing Pearson's correlation coefficient between the resulting vector and column 3. However, as especially for small corpora many of the word frequencies can be zero, and as the logarithm of zero is not defined, we applied the usual heuristic of adding one to each frequency count before taking the logarithm.

4.2 Corpus statistics concerning word associations (1st order statistics)

As described in Rapp (2014c), we assume that there is a relationship between word associations as collected from human subjects and word co-occurrences as observed in a corpus, and our hypothesis is that the strength of this relationship can be used as a measure of corpus representativeness. A corpus leading to simulated associations akin to the ones collected from humans is likely to be a good surrogate for everyday language, although – similarly to what we said about word frequencies – word co-occurrence counts constitute only one of many properties of a corpus.

For extracting word associations from corpora, in the literature many algorithms were described (e.g. Wettler & Rapp, 1989; Church & Hanks, 1990; Wettler et al., 2005). In analogy, we used the following procedure: For all words with a BNC corpus frequency of 50 or higher we computed the co-occurrence vectors. That is, each vector contains the number of co-occurrences of the stimulus word with all other co-occurring words. It counts as a co-occurrence if two words appear together within a distance of at most ten words, i.e. a text window of ±10 words around the stimulus word is considered. Hereby the exact distance within the window is not taken into account.

In a further step an association measure was applied to the co-occurrence vectors, namely Ted Dunning's (1993) log-likelihood ratio. The resulting vectors we call association vectors. Given these vectors, the strongest association to a given stimulus word can be determined by simply looking for the highest value within the respective association vector. The corresponding word is considered to be the associative response predicted by the system. For the same stimulus words used in Table 2, Table 5 shows some sample associations as computed using the British National Corpus.

ABOVE	CONSTELLATİON	FEMİNİNE
below (59)	stars (39)	masculine (26)
level	star (33)	women (2)
average (1)	southern	gender
high (4)	triangle	woman (8)
feet	bright	female (6)
water	planet (1)	men
head	rather	male (1)
see	south	more
ground	find	hair
left	map	soft

Table 5: Top ten corpus-derived associations for three stimulus words. The numbers of subjects from the EAT responding with the respective word (if larger than zero) are given in brackets.

Concerning evaluation, in principle the idea is to find matches between the human and the corpus-based associations. One possibility is to simply count the number of cases where the primary associative response matches the strongest corpus-based association. However, when it comes to very small corpus sizes of e.g. just 1000 words (see Section 5), the problem of data sparseness becomes so severe that a more tolerant evaluation method leads to more robust results less susceptible to statistical variation. This is why for measuring accuracy we count the number of cases where the respective primary associative response is listed within the top ten corpus-based associations, rather than insisting on a

match with the strongest association. This simple modification leads to improvements in reliability when measuring very low accuracies.

4.3 Corpus statistics concerning word relatedness (2nd order statistics)

Our algorithm for computing word relatedness consists of the following three steps:

1) Counting word co-occurrences.
2) Applying an association measure to the raw co-occurrence counts.
3) Computing vector similarities.

Steps 1 and 2 are in principle analogous to the previous subsection. Only, as mentioned in Rapp (2009), for computing vector similarities it turns out that it is better to consider a smaller window size (such as ±1 or ±2 around the given word). Also, we used a simpler association measure, namely $log(n_{ij}+1)$, whereby n_{ij} is the number of co-occurrences between words i and j, as it slightly outperformed the log-likelihood ratio in this particular setting.

For step 3 (computing vector similarities) we use the standard cosine measure. Table 6 shows some results as obtained using the British National Corpus. For a quantitative evaluation we utilized the TOEFL synonym data as follows: We compared our system's results to the answers as provided in the TOEFL dataset. Remember that in the TOEFL synonym test the subjects had to choose the word most similar to a given stimulus word from a list of four alternatives. Accordingly, in the simulation, we assumed that the system made the right decision if the correct answer was ranked best among the four alternatives. In a further run, we applied exactly the same procedure to the test set derived from Fernald's synonym dictionary.

burden	responsibility (0.62), expense (0.61), expenditure (0.59), problem (0.59), cost (0.59)
arrogant	rude (0.62), naive (0.61), stupid (0.61), impatient (0.61), haughty (0.61)
desperation	panic (0.60), despair (0.60), exasperation (0.59), stillness (0.58), impatience (0.58)
memorandum	appendix (0.59), document (0.59), submission (0.57), constitution (0.57), disclosure (0.57)
trivial	unimportant (0.63), ridiculous (0.60), trifling (0.60), straightforward (0.60), bizarre (0.60)

Table 6: Semantic similarities extracted from the BNC for five English words using only vocabulary from the synonym test set based on Fernald (1896).

5 Results

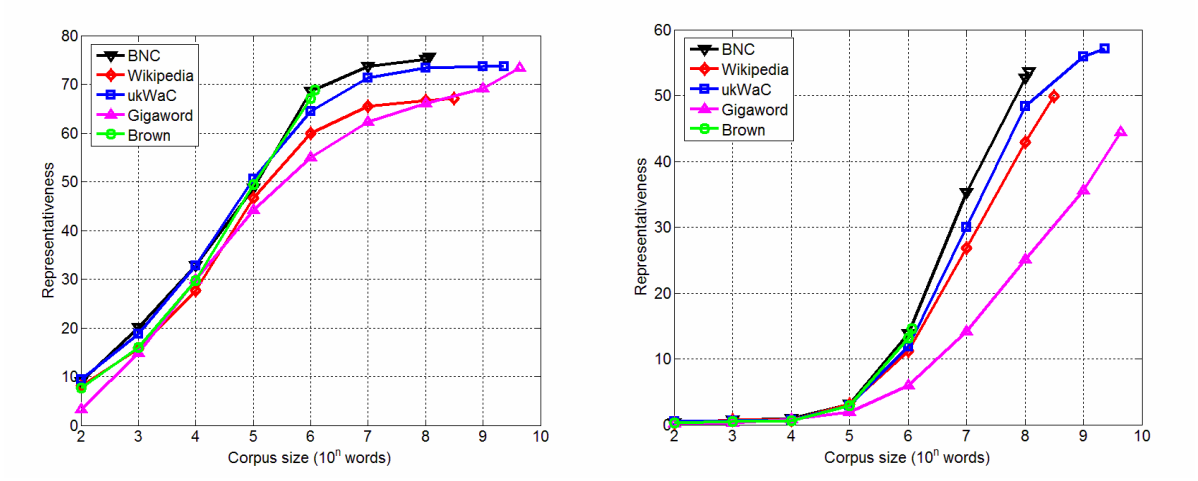
5.1 Results based on word familiarities

These results are given in Figure 1a. There we find in graphical form for each of the five corpora the computed Pearson's correlation coefficients between the words' familiarities and their log corpus frequencies. For easier comparison with the other results (which are percentages) we multiply these correlations by 100 and take the product as the familiarity-based *representativeness of a corpus*. The range of values can thus be between 0 and 100, whereby 0 denotes a complete lack of representativeness, and 100 denotes perfect representativeness. The representativeness scores are also computed for partial corpora, whereby all parts have in common that they start with the beginning of the respective corpus.

We can see in Fig. 1a that, as expected, the representativeness is almost zero if only the first 100 words of a corpus are taken into account, and gradually increases to at least 67 for the full corpora. The horizontal axis has a logarithmic scale, but still the curves flatten with increasing corpus size, especially above 1 million words.

5.2 Results based on word associations

These results are given in Fig. 1b. For each of the five corpora (and their parts) the percentages of primary associative responses are given which ranked among the top ten in the corpus-based associations. These percentages we take as the association-based representativeness of the respective corpus. The range of values is between 0 and 100.



(a) Familiarity

(b) Association

(c) Relatedness

(d) Average

Fig. 1: Results for the three approaches and their average.

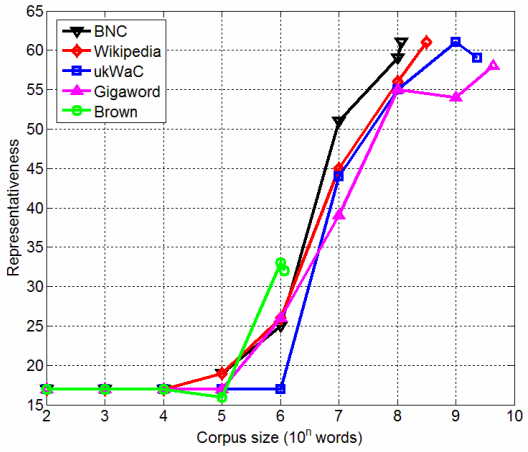


Fig. 2: Results for the TOEFL synonym data.

5.3 Results based on word relatedness

The respective results for the TOEFL synonym data (based on a window size of ±1 words) are given in Fig. 2. There we find for each of the five corpora the percentage of TOEFL questions which were answered correctly. These percentages we take as the relatedness-based representativeness of the respective corpus. Note that the level for very small corpora is higher here as in the TOEFL data with a limited number of candidate words there is a better chance to randomly hit the correct word. As can be seen, the curves are somewhat erratic which is an indication that the test set of 80 items is too small.

For this reason, we did not further use these results but replaced them with those from the synonym test set derived from Fernald (1896). The respective results are shown in Figure 1c.[11] As can be seen, the much higher number of test items leads to smoother curves, but nevertheless the tendencies from the TOEFL data are roughly confirmed.

5.4 Results based on the overall average

The average of the curves in Fig. 1a to 1c is shown in Fig. 1d. The motivation is that this way all three types of statistics are taken into account in a straightforward way. The underlying reasoning is analogous to the BLEU score (Papineni et al., 2002) used in machine translation evaluation: There n-gram matches between a machine translation and a reference translation are counted separately for n-grams of various lengths, and then the individual scores are combined.

6 Discussion

If we compare the curves in Figures 1a, 1b, and 1c it is apparent that the shapes are rather different. This can be explained by the order of the respective statistics: The familiarity-based approach uses statistics of order zero (word frequencies), the association-based approach first order statistics (word co-occurrences), and the relatedness-based approach second order statistics (common context). Although for all three methods a flattening of the curves can be expected for large corpora for the reason that there is an upper limit of corpus representativeness (100) leading to saturation, apparently for the first and second order statistics larger corpora would be required to make this happen.

Concerning very small partial corpora, for the familiarity based approach the curves quickly rise, whereas for the association-based and the relatedness-based approaches the increases in accuracy are small at the beginning. This is also to be expected because in a partial corpus of e.g. 1000 words there is still a chance to find a particular word, but there is almost no chance to find a particular co-occurrence or a common context.

So these discrepancies between the approaches are not a major surprise. Of more interest is a comparison of the results between the different corpora, i.e. their relative performance for each of the methods.

Following Rapp (2014a), concerning the representativeness of our five corpora and their parts, we had tried to come up with some hypotheses before we started to compute the results. These were our predictions:

1) Representativeness should increase with corpus size.
2) The Brown corpus and the BNC should be more representative than unbalanced corpora of the same size.
3) The Brown corpus (1 million words) should be more representative than the first million words of the British National Corpus as the latter is balanced only over its full size (100 million words), but not over its first million words.
4) For same sizes, we would expect ukWaC to be more representative than Wikipedia as we think that corpus heterogeneity is a plus for representativeness. ukWaC is obviously more heterogeneous as, for example, it is multi genre multi topic whereas Wikipedia is single genre multi topic.
5) The Gigaword Corpus should be the least representative for identical sizes. Although, like Wikipedia, it is also single genre multi topic, the distribution of topics is not as wide because in newsticker texts there are strong foci e.g. on politics and sports.

[11] As the Synonym-Dataset involves many very rare words, to reduce data sparseness we used a larger window size of ±2 words to compute these results.

If we compare these hypotheses to the actual results shown in Figures 1a, 1b, and 1c, the findings are as follows:

Hypothesis 1, namely that the representativeness of all corpora steadily increases with corpus size, is clearly confirmed by all three approaches.

Hypothesis 2, saying that the balanced corpora, namely the Brown corpus and the BNC, should be more representative for their sizes than non-balanced corpora, is also confirmed by all approaches. At 1 million words, these two are the top performers. At 100 million words, the BNC performs best. Note, however, that the smaller the corpus sizes, the less predictable the results as the sampling errors increase.

Hypothesis 3 (Brown better than BNC for 1 million words) could sometimes be confirmed but not consistently. Instead, for all approaches the results of these two corpora are fairly close. This indicates that the BNC also seems to have a fairly good balance over the first million words. Concerning the association-based approach, the BNC also has the advantage that its British English should reflect the EAT associations (collected in Edinburgh) better than the American English of the Brown corpus.

Hypothesis 4, namely that ukWaC is better than Wikipedia, is confirmed for the familiarity- and the association-based approach, but not for the relatedness-based approach. Our explanation for the discrepancy is that the relatedness-data contains a larger proportion of outdated and rare words, and that for rare words the coverage of a corpus becomes more important. In this respect, Wikipedia with its wide coverage of topics is likely to have an advantage over the ukWaC corpus.

Hypothesis 5, saying that the Gigaword corpus should be the least representative, is confirmed for almost all corpus sizes.

Overall, several of our hypotheses were consistently confirmed by all approaches. This finding provides some evidence that the computed scores are actually related to what might sensibly be considered as the representativeness of a corpus.

Concerning the average representativeness score (Fig. 1d), we can conclude that overall it seems to make sense to balance a corpus, and that corpus heterogeneity is a plus.

7 Summary and outlook

In this work we defined the term *corpus representativeness* as the ability of a corpus to represent the average language use a native speaker encounters in everyday life. As we cannot easily observe test persons over years, our suggestion was to utilize human intuitions on word familiarities, on word associations, and on word relatedness.

Previous work has provided evidence that human word familiarities are based on word frequencies in perceived language (Rapp, 2005), that human word associations are based on the co-occurrences of words (Wettler et al., 2005), and that human relatedness judgments are based on common context (cf. Harris' (1954) distributional hypothesis). Although all of this may still be controversial, in the current work we took these findings for granted but turned round the perspective. We said that a corpus is representative for the language environment of a group of persons if the word familiarities, the word associations, and the predictions of word relatedness derived from it resemble these persons' intuitions.

For full and partial versions of five well known English corpora we computed the word familiarities, word associations, and word relatedness scores for test sets of several thousand words. We then, for each corpus, compared the extracted information to the human data, and computed similarity scores which we took as measures of corpus representativeness. We also computed a combined score by averaging the results from all three measures.

A shortcoming of our approach is the following: Our measures are limited in so far as they only consider three particular aspects of corpus representativeness, namely word familiarity word association, and word relatedness. They do not explicitly consider higher level features e.g. concerning syntax, semantics, pragmatics, or style.[12] We nevertheless hope that what we described can serve as a starting point for further discussion.

Concerning future work, a possible strait of research would be to modify the relatedness-based approach in a way that the WordSimilarity-353 Test Collection[13] could be used. This test set provides

[12] In section 5.4, when combining our three approaches, we already mentioned an analogy to the BLEU score. A related commonality is that the BLEU score also has these shortcomings.

[13] http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/

direct similarity estimates between words, so a correlation to corpus-derived estimates could be computed in analogy to what we did for the familiarity-based approach.

We would also like to extend the approach to other corpus statistics which seem relevant for human language processing. For example, we might look at associations when given several stimulus words (see Rapp, 2014b), or we could try to predict a word from its WordNet synset. The latter would have the advantage that WordNets are available for many languages, so the corpus representativeness scores could be measured for a number of languages where other human data is scarce.

Related to this would be the use of the Princeton evocation data[14] which provides human similarity estimates between WordNet synsets. The aim would be to replicate these similarities using multiword associations.

Acknowledgement

This research was supported by a Marie Curie Intra European Fellowship within the 7th European Community Framework Programme.

References

Biber, D. (1993): Representativeness in Corpus Design. *Literary and Linguistic Computing*, Vol. 8, Nov. 4, 243–257.

Brisbaert, M.; New, B. (2009). Moving beyond Kučera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for American English. *Behavior Research Methods 41 (4),* 977–990.

Burnard, L.; Aston, G. (1998): *The BNC Handbook: Exploring the British National Corpus with Sara.* Edinburgh: University Press.

Church, K.W.; Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational Linguistics* 16 (1), 22–29.

Coltheart, M. (1981): The MRC psycholinguistic database. *Quarterly Journal of Experimental Psychology*, 33A, 497–505.

Denoyer, L.; Gallinari, P. (2006): The Wikipedia XML Corpus. *SIGIR Forum*, 40(1), 64–69.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistic*s, 19 (1), 61–74.

Fernald, J.C. (1896). *English Synonyms and Antonyms*, 19th edition. New York and London: Funk & Wagnalls Company.

Francis, W.N.; Kuçera, H. (1989): *Manual of Information to Accompany a Standard Corpus of Present-Day Edited American English, for Use with Digital Computers.* Providence, R.I.: Brown University, Department of Linguistics.

Francom, J.; Ussishkin, A. (2011). Converging methodologies: assessing corpus representativeness through psycholinguistic measures. *American Association for Corpus Linguistics.* Georgia State University, Atlanta, GA. http://francojc.files.wordpress.com/2010/01/aacl-2011-converging-methodologies.pdf.

Gries, S.T. (2010). Dispersions and adjusted frequencies in corpora: further explorations. In S.T. Gries, S. Wulff, & M. Davies (eds.): *Corpus Linguistic Applications: Current Studies, New Directions*. Amsterdam: Rodopi, 197–212.

Harris, Z. S. (1954). Distributional structure. *Word*, *10*(23), 146–162.

Kiss, G.R., Armstrong, C., Milroy, R., and Piper, J. (1973). An associative thesaurus of English and its computer analysis. In Aitken, A.J., Bailey, R.W. and Hamilton-Smith, N. (Eds.): *The Computer and Literary Studies*. Edinburgh: University Press, 153–165.

Landauer, T.K.; Dumais, S.T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review* 104 (2), 211–240.

[14] http://wordnet.cs.princeton.edu/downloads.html

Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting of the ACL*, Philadelphia, 311–318.

Rapp, R. (2005): On the relationship between word frequency and word familiarity. In: B. Fisseni; H.-C. Schmitz; B. Schröder; P. Wagner (eds.): *Sprachtechnologie, mobile Kommunikation und linguistische Ressourcen. Beiträge zur GLDV-Tagung 2005 in Bonn.* Frankfurt: Peter Lang. 249–263.

Rapp, R. (2009**).** The automatic generation of thesauri of related words for English, French, German, and Russian**.** *International Journal of Speech Technology* 11 (3), 147–156**.**

Rapp, R. (2011). Language acquisition as the detection, memorization, and reproduction of statistical regularities in perceived language. *Journal of Cognitive Science*, Vol. 12, No. 3, 297–322.

Rapp, R. (2014a). Using word familiarities and word associations to measure corpus representativeness. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*, Reykjavik, Iceland.

Rapp, R. (2014b). Corpus-based computation of reverse associations. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*, Reykjavik, Iceland.

Rapp, R. (2014c). Using word association norms to measure corpus representativeness. In: A. Gelbukh: *Computational Linguistics and Intelligent Text Processing*. 15th International Conference, CICLING 2014, Kathmandu, Nepal. Berlin: Springer. 1–13.

Saldanha, G. (2009): Principles of corpus linguistics and their application to translation studies research. *Tradumática* 7: 1–7.

Temnikova, I.; Baumgartner Jr., W.A.; Hailu, N.D.; Nikolova, I.; McEnery, T.; Kilgarriff, A.; Angelova, G.; Cohen, K.B. (2014). Sublanguage Corpus Analysis Toolkit: a tool for assessing the representativeness and sublanguage characteristics of corpora. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*, Reykjavik, Iceland.

Wettler, M., Rapp, R. (1989). A connectionist system to simulate lexical decisions in information retrieval. In: R. Pfeifer, Z. Schreter, F. Fogelman, L. Steels (eds.): *Connectionism in Perspective*. Amsterdam: Elsevier, 463–469.

Wettler, M.; Rapp, R.; Sedlmeier, P. (2005). Free word associations correspond to contiguities between words in texts. *Journal of Quantitative Linguistics* 12(2), 111–122.

Limited memory incremental coreference resolution

Kellie Webster and **James R. Curran**
ə-lab, School of Information Technologies
University of Sydney
NSW 2006, Australia
{kellie.webster,james.r.curran}@sydney.edu.au

Abstract

We propose an algorithm for coreference resolution based on analogy with shift-reduce parsing. By reconceptualising the task in this way, we unite ranking- and cluster-based approaches to coreference resolution, which have until now been largely orthogonal. Additionally, our framework naturally lends itself to rich discourse modelling, which we use to define a series of psycholinguistically motivated features. We achieve CoNLL scores of 63.33 and 62.91 on the CoNLL-2012 DEV and TEST splits of the OntoNotes 5 corpus, beating the publicly available state of the art systems. These results are also competitive with the best reported research systems despite our system having low memory requirements and a simpler model.

1 Introduction

Coreference resolution is the task of partitioning mentions in a document, usually noun phrases, into clusters which correspond to their real world referents. It is typically approached as a classification task between mentions; given a set of mentions, systems predict the likelihood of their being coreferential with one another and translate these scores into a clustering in a decoding phase.

The task has received considerable research attention due to its importance for downstream inference in tasks such as named entity linking and relation extraction. While simple, local models of coreference have established a reasonable baseline, encoding global consistency requirements remains a challenge since their complete representation is computationally intractable. Two promising but orthogonal directions addressing the need for global consistency measures are ranking-based decoding (Ng and Cardie, 2002; Denis et al., 2007; Fernandes et al., 2012; Durrett and Klein, 2013; Chang et al., 2013) and cluster-based modelling (Rahman and Ng, 2009; Raghunathan et al., 2010; Lee et al., 2011; Klenner and Tuggener, 2011). However, among current systems, decoding strategies are increasingly complex and cluster-based models do not fully leverage psycholinguistic cues such as reading order.

The primary contribution of our work is a reconceptualisation of the coreference task by analogy with the shift-reduce parsing algorithm. This reconceptualisation allows us to capitilise on both ranking- and cluster-based approaches and our system, LIMERIC, outperforms systems using either approach in isolation. We go beyond the shift-reduce algorithm by interpreting our stack of partially formed clusters as a reader's mental status while reading. This allows us to introduce a series of rich discourse features which capture antecedent competition and cognitive accessiblity via a cluster's position in the stack.

Our system is simple and efficient, using maximum-margin averaged perceptron classification and optional beam-search decoding during inference. Despite requiring only a limited amount of memory, our system achieves the competitive CoNLL scores of 63.33 and 62.91 on the CoNLL-2012 DEV and TEST splits of the OntoNotes 5 corpus (Pradhan et al., 2012). We argue that this is due to its more faithful representation of cognitive processing and that extending psycholinguistic insights in modelling is a very promising research direction for even further improvement.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2129–2139, Dublin, Ireland, August 23-29 2014.

2 Related work

Early computational approaches to coreference resolution were built around what is now referred to as mention-pair models. Such models use two stage resolution; the first stage calculates pairwise scores reflecting the likelihood that a mention and its candidate antecedents are coreferential while the second phase decodes these scores into coreference clusters. The simplest way to decode is locationally greedy (Soon et al., 2001), in that the closest candidate with a compatibility score over some threshold is deemed a mention's antecedent. Anaphoricity determination (determining whether a mention constitutes a good first mention of an entity) is mediated by the threshold since a mention without a sufficiently good candidate antecedent starts a new cluster. While these local models achieve a reasonable baseline (Soon et al. (2001) achieves MUC F-scores of 62.6 and 60.4 on MUC 6 and 7), they can make global consistency errors which limit their usefulness downstream. For instance, in the following excerpt from `bn/voa/00/voa_0068` of OntoNotes 5, it is possible that a system uses local evidence such as synonymy to misclassify the ship as the antecedent of a huge Norwegian transport vessel and similarly The battered US Navy destroyer Cole as the antecendent of the ship; unfortunately, these local decisions imply a clustering in which Cole is referred to as a Norwegian transport vessel.

> **The battered US Navy destroyer Cole** has begun **its** journey home from Yemen ... Flanked by other US warships and guarded by aircraft, **the ship** was towed out of Aden Harbor to rendezvous with **a huge Norwegian transport vessel**

While exhaustive comparison would remedy the situation, complete inference has exponential time complexity and so is unrealistic for practical systems. Furthermore, since humans are able to resolve reference on the fly, it seems reasonable that psycholinguistic heuristics would similarly help the task while remaining efficient.

Active research aims to approximately encode global consistency measures, via ranking-based decoding and cluster-level modelling. Ranking-based decoding strategies (Ng and Cardie, 2002; Denis et al., 2007) improve locationally greedy decoding by defining a search window and deeming the best, rather than the closest, candidate within the window to be a mention's antecedent. The publicly available Reconcile system[1] (Stoyanov et al., 2010a; Stoyanov et al., 2010b) uses a simple encoding of this strategy while more recent approaches (Fernandes et al., 2012; Durrett and Klein, 2013; Chang et al., 2013) incorporate the concept within highly sophisticated models. While these systems achieve state of the art performance, they do so at the expense of model complexity.

In cluster-level modelling approaches (Rahman and Ng, 2009; Raghunathan et al., 2010; Lee et al., 2011; Klenner and Tuggener, 2011), instead of basing scoring on the compatibility of pairs of mentions, mentions are compared against incrementally grown partial clusters. This, for instance, may allow a huge Norwegian transport vessel to be compared against a cluster containing both the ship and The battered US Navy destroyer Cole, allowing nationality discord to weigh against the clustering. In this way, global consistency information becomes more important as a mention needs to be compatible with multiple mentions in a cluster, rather than its closest or best antecedent. However, there have been problems with these implementations including their being heavily focussed on surface level features and failing to fully utilise psycholinguistic cues such as reading order. A notable exception is Recasens et al. (2013), which provides a computational model of low salience discourse entities and demonstrates its efficacy in filtering system mentions in the Stanford sieve system (Raghunathan et al., 2010; Lee et al., 2011).

Consistent with Klenner and Tuggener (2011) and others, we argue that psycholinguistic insight is the key to unite cluster- and ranking-based models. This is because theories such as Centering Theory (Grosz et al., 1995) and Accessibility Theory (Ariel, 2001) describe how the human mind keeps track of discourse referents as entities rather than distinct mentions, and resolves anaphora via ranked cognitive accessibility. By reformulating the coreference resolution by analogy with the shift-reduce parsing algorithm, we gain access to the stack of active discourse entities which we rank in order of salience. In this way, the stack in our model becomes an approximation of a reader's mental state when reading a document, allowing us to directly model cognitive models of discourse.

[1] `http://www.cs.utah.edu/nlp/reconcile`

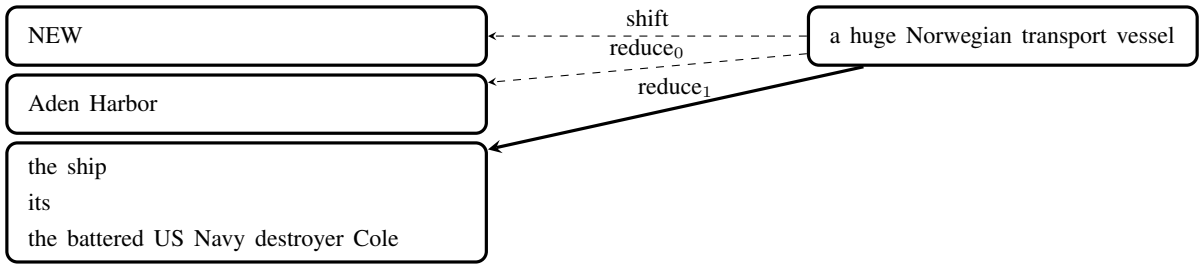


Figure 1: shift-reduce comparisons, `bn/voa/00/voa_0068`

3 Psycholinguistically informed coreference resolution

The shift-reduce algorithm (Aho and Johnson, 1974) is widely used in parsing due to its efficiency, the simplicity of its data structures, and its limited memory usage. For syntactic parsing, a queue is initialised with a series of tokens which is processed in a single reading order pass. Tokens either *shift* onto a stack as a leaf fragment or *reduce* with an existing fragment to former a larger phrasal unit. For the reduce operation, the classifier needs to determine into which fragment the token should merge.

By drawing an analogy between tokens and phrases in syntactic parsing with mentions and clusters in coreference resolution, we derive an algorithm for the latter. In particular, we can initialise a queue of mentions and maintain a stack of clusters which incrementally grow as we read a document. Our classifier is similarly tasked with determining whether a mention should shift onto the stack as the first mention of a new discourse entity or should reduce with an already active one (see Figure 1 for the example of resolving the enqueued mention a huge Norwegian transport vessel). For reduce operations, we additionally need to determine into which entity cluster the mentions should merge. In this way, the shift-reduce algorithm expresses a joint decision of anaphoricity and coreferentiality.

We draw from shift-reduce parsing its simplicity and small memory requirement since we believe these give rise to a more faithful representation of cognitive processing. There are, however, some technical points to consider. Instead of the reduce operation applying to a small window at the top of the stack (top two in the case of binarised grammars), we want to search potentially the whole stack, as described the general formulation of the algorithm. While a full search gives our process worst case $O(n^2)$ time complexity, this is only occurs in the case of an incoherent document which mentions each of its discourse entities exactly once. In the average case, exhaustive stack search still represents a time saving compared to full mention-pair models which compare each mention against all potential antecedent *mentions*. Also, we don't aim to form a single full tree covering all the mentions but rather a collection of clusters. While it is possible to define a document graph of coreference relations (as demonstrated in Fernandes et al. (2012)), it is not necessary to do so.

The algorithms we employ for training and inference our system are represented in Figure 2.

Initialisation

We initialise the stack to be empty and the queue to be the complete set of mentions extracted from the parse structure and named entities in a document. Following the literature, our mention extraction module is designed to be high recall since missed mentions are guaranteed to hurt performance, while it is possible to learn that spurious mentions should not be reported (e.g. Durrett and Klein (2013)). Thus, we train and test on predicted mentions despite the availability of gold mentions for training (to keep system input as similar as possible between training and testing environments) and at test time (since this is not realistic). In this way, we learn a model that is robust to noise in mention extraction.

Learning

On each training pass through a document, we read the enqueued mentions exactly once, in reading order without look ahead. As each mention comes to head the queue, we generate a training instance in which the classifier decides whether it is more likely that the mention *shift* onto the stack as the first mention of a new discourse entity or *reduce* with the cluster of an already active one. In particular, the reduce score

```
initialise queue;
initialise stack;
while queue do
    active = queue.pop();
    prediction = classify(active, stack);
    gold = correct_classification(active, stack);
    if prediction != gold then
        update(prediction, gold);
    end
    cluster = apply_pred(active, stack, gold);
    promote(cluster, stack);
end
```

```
initialise queue;
initialise stacks;
while queue do
    active = queue.pop();
    forall stacks do
        prediction = classify(active, stack);
        cluster = apply_pred(active, stack, prediction);
        promote(cluster, stack);
    end
    prune_stacks(stacks);
end
```

Figure 2: learning (left) and inference (right) algorithms

is the highest of all potential merges. Features are generated on the fly to reduce memory requirements, and because the state of the system is determined by each move made. The margin of classification is widened by augmenting by one the scores corresponding to non-gold decisions.

We then determine whether any difference exists between the classifier's decision and the gold answer key by looking for one of five errors, three taken from Durrett and Klein (2013) (falsely anaphoric, falsely new, wrong link made) and two inspired by the categories used in Kummerfeld and Klein (2013) (extra mention and extra entity). If an error is detected, we perform perceptron updates of the feature weights, increasing those corresponding to the gold decision and decreasing those corresponding to the incorrect prediction. We find that varying the feature value update according to the error made has a performance benefit, particularly when 'falsely new' is given a faster learning rate. This may due to sparsity: across a corpus, the number of first mentions of an entity is smaller than both that of subsequent mentions, and singleton mentions. We note that it should be possible to learn a model using uniform updates by increasing the number of training iterations, though this increases the chance of overfitting. Also, tuning these parameters may affect different balances in error types for different applications.

As noted in Rahman and Ng (2009), since the mention-cluster indicator functions do not apply to the case where a new entity is formed (shift operations), reduce comparisons activate many more features than shift ones do. During development, we noticed that this marked difference in feature set size was negatively impacting performance as reduce operations were unfairly favoured. To grow the shift feature weights faster, we introduced a scaling parameter on the update of these feature weights; we found the ratio of the feature space sizes to work well.

As the final stage, the system applies the decided move. There are two valid 'decided' moves, namely the correct decision, read from the gold standard, or the (potentially incorrect) predicted decision. In this work, we train by following the path of correct decisions, though we plan future research implementing the latter. We hope this will improve the robustness of our system given analogous findings in shift-reduce parsing (Zhang and Nivre, 2012). Novel to our approach, the cluster resulting from application of the decided move is promoted to the top of the stack since recency increases cognitive accessibility. This is a crucial implementation detail given the cognitive interpretation we give to the stack of clusters.

Inference

A benefit of our formulation of the coreference task is that inference is little different to training, without feature weight tuning. In both, documents are processed via a queue of mentions, though a single stack is replaced by possibly multiple in a beam regularly pruned to a fixed width. This has possible cognitive underpinnings since humans need to be able to back track if an interpretation proves incorrect. Analogously, it allows our system to reduce the impact of potentially harmful local decisions. Interestingly, we find in Section 6 that this has little appreciable impact on performance, though this is consistent with Zhang and Nivre (2012), which finds that beam search in inference can hurt the performance of a shift-reduce syntactic parser trained on gold decisions.

4 Rich discourse features

We base our feature space on the pool of features described in the literature (Soon et al., 2001; Ng and Cardie, 2002; Bengtson and Roth, 2008; Stoyanov et al., 2010a; Stoyanov et al., 2010b; Raghunathan et al., 2010; Lee et al., 2011). We introduce *Discourse likelihood* as a novel extension of work in Recasens et al. (2013), designed to mediate system conservativeness in the decision between whether a cluster remains a singleton or grows into a larger cluster. Conjunct discord is introduced as a finer grained extension of traditional number agreement features.

If existing features apply to single mentions or single clusters, they apply in this same way in our system. To map the mention-pair features in the literature into functions which take a mention-cluster pair, we use a range of strategies including the existence of a compatible mention for the active among the cluster pool (Raghunathan et al., 2010), binned proportion of clustered mentions compatible with the active mention (Rahman and Ng, 2009), or the maximum compatibility score between the clustered mentions and the active one (based on Ponzetto and Strube (2006)).
The examples here correspond to the reduce$_1$ move in Figure 1.

Lexical data driven lexicalised features from Durrett and Klein (2013); for the active mention the ship, we would generate the features like **`head_word:ship first_pos:DT last_shape:LOWER`**

String match existence and proportion of clustered mentions with various string matches with the active mention, e.g. **`head_match:none acronym:none`**

Attribute agreement agreement in animacy, gender, number, and NER values pooled across the cluster, and active, e.g.**`number_agree:true`**

Attribute discord where mentions are conjunctions, disagreement between the number of sibling NP children; disagreement between the citation form of any pronouns in cluster and active

Syntax existence of i-within-i (restriction on anaphora due to government and binding requirements on a sentence's parse tree) or subject-object relation between active and any clustered mention e.g.**`iwithini:none`**

Semantics binned value of maximum Lin et al. (2012) similarity score between active and clustered mention heads, e.g. **`lin:high`** since ship and vessel are highly related; disagreement between coarse grained semantic classes of nominals determined from WordNet (Fellbaum, 1998)

Length length of mention in number of tokens **`mlength:3`**; length of cluster in number of mentions

Distance distance between active and closest clustered mention, measured in number of sentences and number of intervening mentions

Discourse patterns whether any subsequent mention is an indefinite nominal

Discourse likelihood an integer value representing the likelihood that cluster has proposed length (singleton or not) given the internal morphosyntactics of the clustered mentions; likelihood of stack given likelihood of contained clusters

4.1 Stack features

Since position in stack in our model represents relative cognitive accessibility, we introduce Depth features as the cognitive analogues of Distance features, designed to more faithfully represent accessibility.

Stack depth depth from top of the stack, binned as top cluster in stack, within five clusters from the top, within ten clusters from the top, outside this[2]; raw depth, depth normalised in turn by ignoring singletons and ignoring clusters not containing a proper name mention **`raw_depth:high`**, **`ne_depth:top`** were all used, with the last two designed to capture the impact of salience

[2]these values were empirically optimised, though we note that they reflect known constraints on human short-term memory

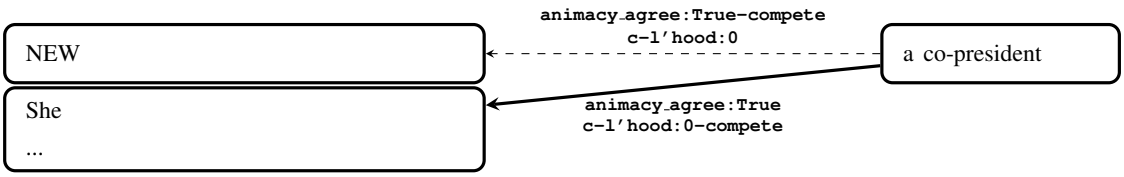


Figure 3: stack features, bn/mnb/00/mnb_0023

Stack competition In an aim to model the competition between clusters in the stack, we introduce stack competition features. In addition to evaluating our mention-cluster features between the active mention and its proposed antecedent, we also evaluate them between it and the other clusters comprising the stack. If a stacked cluster evaluates positively with any of these, we generate a labelled version of the indicated feature. By having these features compete with those of the proposed antecedent, we hope to better learn a more global ranking of candidates than straightforward search window strategies do. Figure 3 shows how stack features can be used to distinguish between the attractiveness of merging an indefinite nominal into a cluster (attractive due to matching linguistic attributes) as compared to starting a new discourse entity (attractive due to discourse likelihood of indefinite nominal in a new cluster).

4.2 Discourse transition prefixing

After Durrett and Klein (2013), we use discourse transition strings formed from the types of the mention and its closest candidate antecedent as feature prefixes, e.g. **m:nominal+a:nominal**. While this inflates the potential size of the feature space[3], the features generated are more meaningful since we would expect many indicator functions to behave differently for pronouns than for subsequent proper names, for example, reintroducing entities. Also, since we use perceptron learning, feature weights are only tuned if the feature is useful in making a decision during training.

5 Results

We evaluate LIMERIC on the OntoNotes 5 corpus (Pradhan et al., 2012) with the included parse and NER annotations. Our experimental setup matches the specifications of the CoNLL-2012 shared task: we use the standard corpus splits, official scorer, and report performance on the CoNLL metric which averages the MUC F-score (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998) and CEAFE (Luo, 2005).

We compare our performance against that of three state of the art systems which reflect the diversity of current approaches. Stanford[4] (Lee et al., 2011) has rule-based decoding with cluster-based modelling. Berkeley[5] (Durrett and Klein, 2013) uses mention-pair features in a factor graph model. Since reported performance for this system is on CoNLL-2011, we compare against the publicly available system using the SURFACE model, which doesn't use features induced from English Gigaword (Graff et al., 2007). Chang et al. (2013)'s L^3M systems comprise both mention-pair and cluster-based variants; we focus on the former here since these perform better on OntoNotes 5. L^3M represents a maximum-margin approach to ranking models, where CL^3M adds some cluster modelling via a constraint term.

5.1 Performance

Table 1 presents our performance on DEV and TEST. Our core LIMERIC system includes all features described in Section 4 including our novel discourse features Discourse patterns, Discourse likelihood, and Stack depth. In development, we experiment with system configurations by deactivating semantic features (-*s*) and activating stack competition features (+*c*) in turn. Despite a good CEAFE score, we opt not to include stack competition features in our final system.

Given the simplicity of our learning and decoding, our system compares favourably with existing systems. In all configurations, we beat both publicly available systems and the mention-pair variant L^3M: by uniting aspects of ranking- and cluster-based approaches, we achieve benefits beyond either in

[3] since distinct feature strings correspond to completely distinct features

[4] http://nlp.stanford.edu/software/corenlp.shtml

[5] http://nlp.cs.berkeley.edu/berkeleycoref.shtml

System	MUC	B^3	CEAFE	CoNLL
Stanford	64.30	70.46	46.35	60.37
Berkeley	66.10	68.56	50.09	61.58
L^3M	67.88	71.88	47.16	62.30
CL^3M	69.20	**72.89**	48.67	**63.59**
LIMERIC	**71.02**	68.66	50.31	63.33
LIMERIC +*c*	70.67	68.33	**50.55**	63.18
LIMERIC -*s*	70.53	68.21	50.34	63.03

System	MUC	B^3	CEAFE	CoNLL
Stanford	63.83	68.52	45.36	59.23
Berkeley	69.09	65.89	48.26	61.08
L^3M	68.31	70.81	46.73	61.95
CL^3M	69.64	**71.93**	48.32	**63.30**
LIMERIC	**71.52**	67.47	**49.75**	62.91

Table 1: CoNLL-2012 DEV (left) and TEST (right)

isolation. Also, we consistently outperform CL^3M on two of the three performance metrics; our method for uniting existing approaches is more direct and psycholinguistically faithful than that in CL^3M and our competitive system results are promising for future work.

Our system's MUC and CEAFE scores are the highest across all systems on both datasets. Our high CEAFE score in particular suggests that our system produces an accurate *number* of clusters. We explore this further in Figure 4 using the tool described in Kummerfeld and Klein (2013)[6]. Between Berkeley and LIMERIC, the notable difference is that we make considerably fewer Divided Entity and Missed Entity errors for a small increase in Conflated Entity errors. By introducing features which model when a new discourse entity should form and how the relative accessibility of already active ones impacts coreference decisions, we more accurately predict the bounds of entity clusters. This modelling is independent of surface features: 85% and 96% of Berkeley's Divided Entity errors occur where there is no head match and string match between mentions, respectively, compared to our values of 87% and 96%.

We note also that, given Kummerfeld and Klein's finding that MUC recall is highly sensitive to Divided Entity errors and B^3 precision to Conflated Entity errors, we can understand our performance on these metrics, particularly if our errors occur in larger clusters.

Between LIMERIC and LIMERIC+*c*, the notable difference is that LIMERIC+*c* makes fewer Missed Mention errors, but at a high cost to Extra Entity errors. A principled solution for future work might be to enrich our model of what makes a discourse transition unfavourable, in contrast to the predominate tradition of modelling what makes a discourse transition favourable.

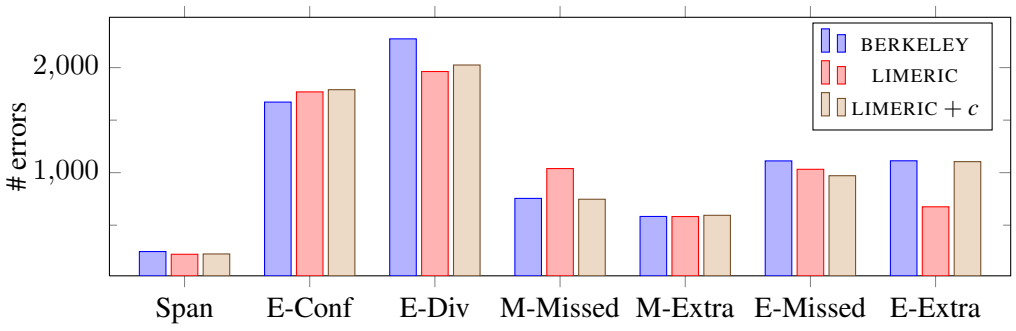


Figure 4: error counts in fine grained categories of Kummerfeld and Klein (2013)

6 System analysis

Features

Since we use simple, linear learning, it is possible to analyse feature weights to introspect system performance. In particular, we would like to understand why our stack competition features, which are well principled, did not give a substantial performance gain. We do this by analysing the number of non-zero

[6] https://code.google.com/p/berkeley-coreference-analyser/

Features	% non-zero	avg. mag.
Surface	17.4	0.23
POS	61.9	4.55
Shape	57.3	3.77
Str match	93.7	20.60
Length	48.7	2.37
Distance	92.5	19.74

Features	% non-zero	avg. mag.
Attr agree	88.3	24.88
Attr discord	74.6	20.00
WN similarity	94.5	21.67
Competition	**87.9**	**15.28**
Likelihood	**61.4**	**7.47**
Depth	**93.9**	**11.73**

Table 2: proportion of features within a set with non-zero weight in LIMERIC+*c* model (left) and average magnitude of this weight across the set (right); novel features are indicated in bold

features in our feature sets as an indication of how often they were useful in distinguishing predictions, and average feature weight magnitude as an indication of how trusted they were in inference.

Given the performance decrease of LIMERIC+*c* against our base system, it is surprising that it is the competition features which appear to be the best performing of our novel feature space. We are cautious that their very high feature weight could represent overfitting and future work could use regularisation, as well as explore any discourse level differences between TRAIN and the TEST datasets.

Depth in stack performs well, particularly given that it captures similar information to Distance and feature weight needs to be shared between the two feature sets. The least useful feature set is Surface, probably due to our large feature space size and its sparseness. Since this comprises the greatest number of features, we anticipate its deactivation will improve efficiency for a minimal impact on performance.

Stack

Our reported performance is based on a search of the full stack, but this gives rise to a large time cost which is not practical given the role of coreference resolution to inform downstream inference. While recency is important cue for coreference, it is not clear what bounds we can place on candidate generation while maintaining good performance. Figure 5 plots the depth from the top of the stack of the correct reduce operation in DEV.

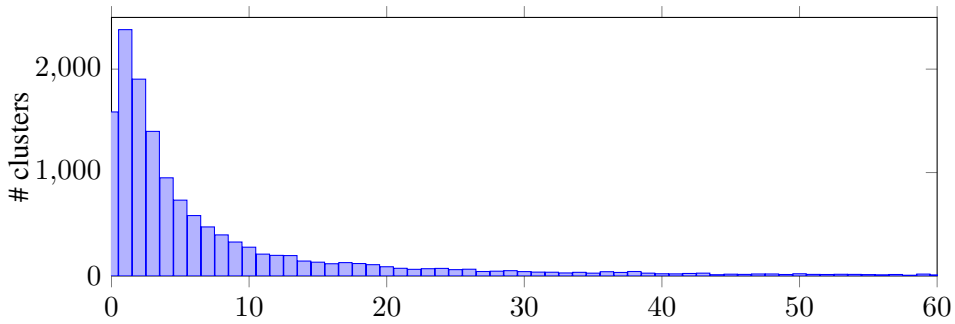


Figure 5: distribution of correct merge targets in the stack, DEV

We note a very long tail to this distribution and cut it off at depth 60, which cumulatively represents 97% of the data. The vast majority of correct merge targets are near the top of the stack, with 78% up to depth 10 and 88% up to depth 20. Setting maximum search depth to 60 yields a model which scores 61.31 on DEV. While this outperforms Stanford and is competitive with Berkeley, the magnitude of loss is surprising given the distribution in Figure 5. Error analysis shows an increased number of Conflated Entity and Extra Mention errors, which were shown in Kummerfeld and Klein (2013) to have a substantial precision cost. We note that this is consistent with our system having good accuracy in predicting whether or not a new entity cluster should form, but being restricted to choose an incorrect merge target when the correct one is outside its search window.

Configuration	MUC	B^3	CEAFE	CoNLL
LIMERIC+*c*	**70.67**	**68.33**	50.55	**63.18**
classifier scoring	70.36	68.12	50.65	63.04
# beams=1	70.52	68.21	**50.66**	63.13
no beam threshold	69.60	67.42	50.12	62.38

Table 3: impact of various parameters for beam search, DEV

Beam search

Beam search affects both time and space complexity since each classification step proposes new stacks which need to be compared for pruning. Our final system uses a maximum beam size of 10 with a conservative threshold of 5 for new stack formation. We find little difference between using classification score or stack discourse likelihood as our pruning metric. The results in Table 3 indicate that beam search isn't essential for state of the art performance in our system, our rich feature set is adequate alone. If we limit the beam to a single stack, we still have competitive performance with CL^3M. Indeed, if we do not set a strict threshold on the score at which a new stack is formed, we are forced to maintain the maximum 10 stacks and this actually hurts performance. These findings are consistent with those in Zhang and Nivre (2012), which demonstrates that performance gains are only seen from beam search at run time when their shift-reduce parser was trained similarly, maintaining a beam of potentially incorrect predictions and learning to recover as well as possible from unfavourable states.

7 Conclusion

The primary contribution of our work is a reconceptualisation of coreference by analogy with the shift-reduce parsing algorithm. We present LIMERIC, a simple, low memory coreference resolution system which achieves the competitive CoNLL scores of 63.33 and 62.91 on the CoNLL-2012 DEV and TEST splits of OntoNotes 5. Our framework unites ranking- and cluster-based approximations to global consistency encoding, and we outperform systems using either in isolation. By interpreting the stack of incrementally growing entity clusters in our system as a reader's mental status while reading, we naturally extend the shift-reduce algorithm to express a series of rich discourse features which perform well in feature analysis. Our results demonstrate the promise of psycholinguistic insights for coreference resolution and future directions include further extension of our discourse, as well as semantic, model. We plan future work in enriching our training process with beam search, and incorporating more insights from Centering and Accessibility Theories.

8 Acknowledgements

The authors thank the anonymous reviewers for their helpful feedback, and Will Radford, Joel Nothman, and Matthew Honnibal for their contribution to this work. The first author was supported by an Australian Postgraduate Award scholarship. This work was supported by ARC Discovery grant DP1097291 and the Capital Markets CRC Computable News project.

References

A. V. Aho and S. C. Johnson. 1974. LR parsing. *ACM Computing Surveys*, 6(2):99–124.

Mira Ariel. 2001. Accessibility theory: An overview. *Text representation: Linguistic and psycholinguistic aspects*, pages 29–87.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566, Grenada, Spain.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Edinburgh, Scotland.

Kai-Wei Chang, Rajhans Samdani, and Dan Roth. 2013. A constrained latent variable model for coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 601–612, Seattle, Washington.

Pascal Denis, Jason Baldridge, et al. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *Proceedings of NAACL HLT*, pages 236–243, Rochester, New York.

Greg Durrett and Dan Klein. 2013. Easy victories and uphill battles in coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1971–1982, Seattle, Washington.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts.

Eraldo Fernandes, Cícero dos Santos, and Ruy Milidiú. 2012. Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 41–48, Jeju Island, Korea.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English gigaword (third edition), LDC catalog number LDC2003T05.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21:203–225.

Manfred Klenner and Don Tuggener. 2011. An incremental entity-mention model for coreference resolution with restrictive antecedent accessibility. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 178–185, Hissar, Bulgaria.

Jonathan K Kummerfeld and Dan Klein. 2013. Error-driven analysis of challenges in coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 265–277, Seattle, Washington.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34, Portland, Oregon.

Thomas Lin, Oren Etzioni, et al. 2012. No noun phrase left behind: detecting and typing unlinkable entities. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 893–903, Seattle, Washington.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 104–111, Philadelphia, Pennsylvania.

Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 192–199, New York, New York.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nate Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501, Cambridge, Massachusetts.

Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 968–977, Edinburgh, Scotland.

Marta Recasens, Marie-Catherine de Marneffe, and Christopher Potts. 2013. The life and death of discourse entities: Identifying singleton mentions. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 627–633, Atlanta, Georgia.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544.

Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010a. Coreference resolution with reconcile. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 156–161, Uppsala, Sweden.

Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010b. Reconcile: A coreference resolution research platform.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th conference on Message understanding*, pages 45–52, Columbia, Maryland.

Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 1391–1400, Bombay, India.

Left-corner Transitions on Dependency Parsing

Hiroshi Noji and **Yusuke Miyao**
Department of Informatics
The Graduate University for Advanced Studies
National Institute of Informatics, Tokyo, Japan
{noji,yusuke}@nii.ac.jp

Abstract

We propose a transition system for dependency parsing with a left-corner parsing strategy. Unlike parsers with conventional transition systems, such as arc-standard or arc-eager, a parser with our system correctly predicts the processing difficulties people have, such as of center-embedding. We characterize our transition system by comparing its oracle behaviors with those of other transition systems on treebanks of 18 typologically diverse languages. A crosslinguistical analysis confirms the universality of the claim that a parser with our system requires less memory for parsing naturally occurring sentences.

1 Introduction

It is sometimes argued that transition-based dependency parsing is appealing not only from an engineering perspective due to its efficiency, but also from a scientific perspective: These parsers process a sentence incrementally similar to a human parser, which have motivated several studies concerning their cognitive plausibility (Nivre, 2004; Boston and Hale, 2007; Boston et al., 2008). A cognitively plausible dependency parser is attractive for many reasons, one of the most important being that dependency treebanks are available in *many languages*, so it is suitable for crosslinguistical studies of human language processing (Keller, 2010). However, current transition systems based on shift-reduce actions fully or partially employ a bottom-up strategy[1], which is problematic from a psycholinguistical point of view: Bottom-up or top-down strategies are known to fail in predicting the difficulty for certain sentences, such as center-embedding, which people have troubles in comprehending (Abney and Johnson, 1991).

We propose a transition system for dependency parsing with a left-corner strategy. For constituency parsing, unlike other strategies, the arc-eager left-corner strategy is known to correctly predict processing difficulties people have (Abney and Johnson, 1991). To the best of our knowledge, however, the idea of left-corner strategy has not been introduced in the dependency parsing literature. We define the memory cost for a transition system as the number of unconnected subtrees on a stack. Under this condition, the proposed system incurs non-constant memory cost only when encountering center-embedded structures.

After developing the transition system, we characterize it by looking into the following question: Is it true that naturally occurring sentences can be parsed on this system with a lower memory overhead? This should be true under the assumptions that 1) people avoid generating a sentence that causes difficulty for them, and 2) center-embedding is a kind of such structure. Specifically, we focus on analyzing the oracle transitions of the system, i.e., parser actions to recover the gold dependency tree for a sentence. In English, it is known that left-corner transformed treebank sentences can be parsed with less memory (Schuler et al., 2010), but our focus in this paper is on the language universality of the claim in a crosslingual setting. Two different but relevant motivations exist for this analysis. The first is to answer the following scientific question: Is the claim that people tend to avoid generating center-embedded sentences language universal? This is unclear since the observation that a center-embedded sentence is

[1]The top-down parser of Hayashi et al. (2012) is an exception, but its processing is not incremental.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2140–2150, Dublin, Ireland, August 23-29 2014.

difficult to comprehend is from psycholinguistic studies mainly on English. The second motivation is to verify whether a parser with the developed system can be viable for crosslinguistical study of human language processing. There is evidence that a human parser cannot store elements of a small constant number, such as three or four (Cowan, 2001). If our system confirms to such a severe constraint, we may claim its cognitive plausibility across languages. We will pursue these questions using the multilingual dependency treebanks from the CoNLL-X and 2007 shared tasks (Buchholz and Marsi, 2006; Nivre et al., 2007).

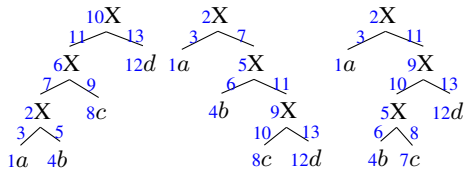
In short, our contributions of this paper can be sketched as follows:

1. We formulate a transition system for dependency parsing with a left-corner strategy.
2. We characterize our transition system with its memory cost by simulating oracle transitions along with other transition systems on the CoNLL multilingual treebanks. This is the first empirical study of required memory for left-corner parsing in a *crosslinguistical* setting.

2 Memory cost of Parsing Algorithms

In this work, we focus on the memory cost for dependency parsing transition systems. While there have been many studies concerning the memory cost for an algorithm in constituency parsing (Abney and Johnson, 1991; Resnik, 1992), the same kind of study is rare in dependency parsing. This section discusses the memory cost for the current dependency parsing transition systems. Before that, we first review the known results in constituency parsing regarding memory cost.

2.1 Center-Embedding and the Left-Corner Strategy



The structures on the right side are called left-branching, right-branching, and center-embedding, respectively. People have difficulty when parsing center-embedded structures, while no difficulty with right-branching or left-branching structures. An example of a center-embedded sentence is *the rat [the cat [the dog chased] bit] ate the cheese*, which is difficult, but if we rewrite it as *the cheese was eaten [by the rat [that bit the cat [that chased the dog]]]*, which is a kind of right-branching structure, the parse becomes easier.

Abney and Johnson (1991) showed that top-down or bottom-up strategies[2] fail to predict this result. For example, for the right-branching structure, a bottom-up strategy requires $O(n)$ memory, since it must first construct a subtree of c and d, but the center-embedded structure requires less memory. The arc-eager left-corner strategy correctly predicts the difficulty of a center-embedded structure, which is characterized by the following order of recognitions of nodes and arcs:

1. A node is enumerated when the subtree of its first child has been enumerated.
2. An arc is enumerated when two nodes it connects have been enumerated.

The numbers on the trees above indicate the order of recognition for this strategy. We can see that it requires a constant memory for both right-branching and left-branching structures. For example, for the right-branching structure, it reaches 7 after reading b, which means that a and b are connected by a subtree. On the other hand, for the center-embedded structure, it reaches 6 after reading b, but a and b cannot be connected at this point, requiring extra memory.

2.2 Transition-based Dependency Parsing

Next, we summarize the issues with current transition systems for dependency parsing with regards to their memory cost. A transition-based dependency parser processes a sentence on a transition system, which is defined as a set of configurations and a set of transitions between configurations (Nivre, 2008). Each configuration has a stack preserving constructed subtrees on which we define the memory cost as a function for each system.

[2]We should distinguish between two types of characterizations of parsing: *strategy* and *algorithm*. A parsing strategy is an abstract notion that defines "a way of enumerating the nodes and arcs of parse trees" (Abney and Johnson, 1991), while a parsing algorithm defines the implementation of that strategy, typically with push-down automata (Johnson-Laird, 1983; Resnik, 1992). A parsing strategy is useful for characterizing the properties of each parser, and we concentrate on the *strategy* for exposition of constituency parsing. For dependency parsing, we mainly discuss the algorithm, i.e., the transition system.

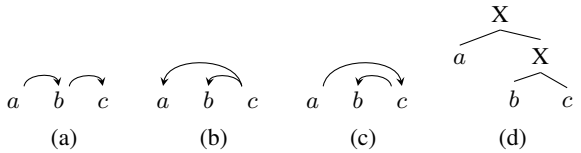


Figure 1: (a)–(c): Right-branching dependency trees for three words; (d): the corresponding CNF.

	Arc-standard	Arc-eager	Left-corner
left-branching	$O(1)$	$O(1)$	$O(1)$
right-branching	$O(n)$	$O(1 \sim n)$	$O(1)$
center-embedding	$O(n)$	$O(1 \sim n)$	$O(n)$

Table 1: Order of the memory cost for each structure for each transition system. $O(1 \sim n)$ means that it processes some structures with a constant cost but some with a non-constant cost.

Arc-Standard We define the memory cost as the number of elements on the stack since all stack elements are disjoint. In this setting, we can see that the arc-standard system has a problem for a right-branching structure, such as $a \curvearrowright b \curvearrowright c \curvearrowright \cdots$, in which the system first pushes all words on the stack before connecting each pair of words, requiring $O(n)$ memory. Nivre (2004) discussed the problem with this system in greater detail, observing that its stack size grows when processing structures that become right-branching when converted to the Chomsky normal form (CNF) of a context-free grammar (CFG). Figure 1 lists those dependency structures for three words, for which the system must construct a subtree of b and c before connecting a to either, requiring extra memory. This is because the system builds a tree bottom-up: each token collects all dependents before being attached to its head. In fact, the arc-standard system is essentially equivalent to the push-down automaton of a CFG in the CNF with a bottom-up strategy (Nivre, 2004), so it has the same property as the bottom-up parser for a CFG.

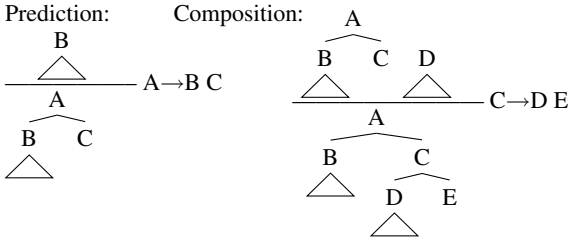
Arc-Eager In the arc-eager system, the stack contains sequences of tokens comprising connected components, so we can define the memory cost as the number of connected components on the stack. With this definition, we can partially resolve the problem with the arc-standard system. The arc-eager system does not incur any cost for processing the structure in Figure1(a) and $a \curvearrowright b \curvearrowright c \curvearrowright \cdots$ since it can connect all tokens on the stack (Nivre, 2004). Because its construction is no longer pure bottom-up, it is difficult to formally characterize the cost based on the type of tree structure. However, this transition system cannot correctly predict difficulties with center-embedding because the cost never increases as long as all dependency arcs are left-to-right, e.g., a sentence $a \curvearrowright b \overset{\frown}{c\ d}$ becomes center-embedding when converted to a CNF, but it does not incur any cost for it. Note that this system still incurs cost for some right-branching structures, such as in Figures 1(b–c), and some center-embedded structures. Therefore, for the arc-eager system, it is complicated to discuss the required order of memory cost. We summarize these results in Table 1. Our goal is to develop an algorithm with the properties of the last column, requiring non-constant memory for only center-embedded structures.

Other systems All systems where stack elements cannot be connected have the same problem as the arc-standard system because of their bottom-up constructions, including the hybrid system of Kuhlmann et al. (2011). Kitagawa and Tanaka-Ishii (2010) and Sartorio et al. (2013) present an interesting variant, which attaches a node to another node that may not be the head of a subtree on the stack. We can use the same reasoning for the arc-eager system for these systems: they sometimes do not incur costs for center-embedded structures, while they incur a non-constant cost for some right-branching structures.

3 Left-corner Dependency Parsing

We now discuss the construction of our transition system with the left-corner *strategy*. Resnik (1992) proposed a push-down recognizer for a CFG. In the following, we instead characterize his algorithm by inference rules, which are more intuitive and helpful to adapt the idea for dependency parsing.

Prediction and Composition There are two characteristic operations in the push-down recognizer of Resnik (1992): **prediction** and **composition**. We show inference rules of these operations on the right side:



Prediction is used to predict the parent node and the sibling of a recognized subtree when the sub-

SHIFT	$(\sigma, j\|\beta, A) \mapsto (\sigma\|\langle j\rangle, \beta, A)$
INSERT	$(\sigma\|\langle\sigma_1'\|i\|x(\lambda)\rangle), j\|\beta, A) \mapsto (\sigma\|\langle\sigma_1'\|i\|j\rangle, \beta, A \cup \{(i,j)\} \cup \{\cup_{k\in\lambda}(j,k)\})$
LEFT-PRED	$(\sigma\|\langle\sigma_{11}, \cdots\rangle, \beta, A) \mapsto (\sigma\|\langle x(\sigma_{11})\rangle, \beta, A)$
RIGHT-PRED	$(\sigma\|\langle\sigma_{11}, \cdots\rangle, \beta, A) \mapsto (\sigma\|\langle\sigma_{11}, x(\emptyset)\rangle, \beta, A)$
LEFT-COMP	$(\sigma\|\langle\sigma_2'\|x(\lambda)\rangle\|\langle\sigma_{11}, \cdots\rangle, \beta, A) \mapsto (\sigma\|\langle\sigma_2'\|x(\lambda \cup \{\sigma_{11}\})\rangle, \beta, A)$
RIGHT-COMP	$(\sigma\|\langle\sigma_2'\|x(\lambda)\rangle\|\langle\sigma_{11}, \cdots\rangle, \beta, A) \mapsto (\sigma\|\langle\sigma_2'\|\sigma_{11}\|x(\emptyset)\rangle, \beta, A \cup \{\cup_{k\in\lambda}(\sigma_{11}, k)\})$

Figure 2: Actions of the left-corner transition system.

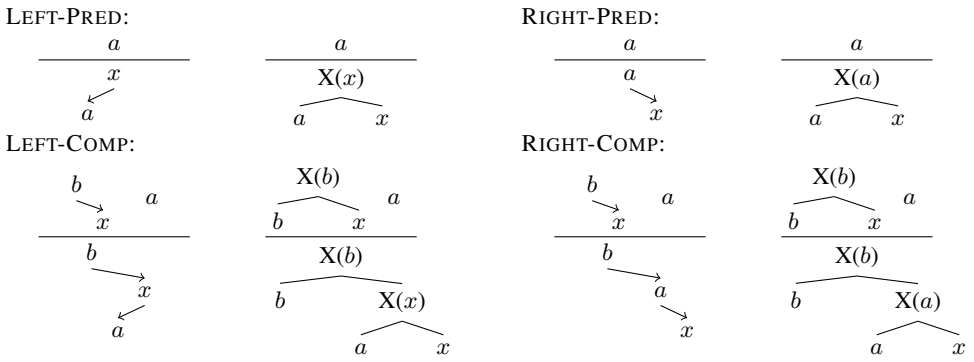
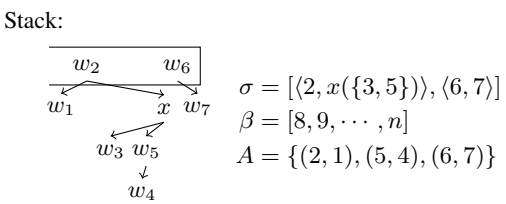


Figure 3: Correspondences of reduce actions between dependency and CFG. Nonterminal X(t) means that its lexical head is t. We only show minimal example subtrees for simplicity. However, a can have an arbitrary number of children, so can b or x, as long as x is on a right spine and has no right children.

tree is complete and its parent node is not yet recognized. **Composition** composes two subtrees by first predicting the parent and the sibling of a recognized subtree then immediately connecting trees by identifying the same node on two trees (C, in this case). This is used when the parent node of a completed subtree has already been predicted as a part of another tree in a top-down fashion.

Dummy Node We now turn to the discussion of dependency parsing. The key characteristic of our transition system is the introduction of a dummy node on a subtree, which is needed to represent a subtree containing some predicted structures as in constituency subtrees for Resnik's recognizer. To get an intuition of the parser actions, we present a simulation of transitions for the sentence in Figure 1(b), of which current systems fail to predict its difficulty. Our system first shifts a then conducts a kind of **prediction** operation, resulting in a subtree $a \swarrow x$, where x is a dummy node. This means that we predict that a will become a left dependent of an incoming word. Next, it shifts b to the stack then conducts a **composition** operation to obtain a tree $a \swarrow b \swarrow x$. It finally inserts c to the position of x, recovering the tree.

Transition system As in many other transition systems, a configuration for our system is a tuple $c = (\sigma, \beta, A)$, where σ is a stack, and we use a vertical bar to signify an append operation, e.g., $\sigma = \sigma'|\sigma_1$ denoting σ_1 is the top most element of the stack σ, and β is an input buffer consisting of token indexes not processed yet. $\beta = j|\beta'$ means j is a first element of β, and $A \subseteq V_w \times V_w$ is a set of arcs given V_w, a set of token indexes for a sentence w.



Each element of a stack is a list representing a **right spine** of a subtree, as in Kitagawa and Tanaka-Ishii (2010) and Sartorio et al. (2013). A right spine $\sigma_i = \langle\sigma_{i1}, \sigma_{i2}, \cdots, \sigma_{ik}\rangle$ consists of all nodes in a descending path from the head of σ_i, i.e., σ_{i1}, taking the rightmost child at each step. We also write $\sigma_i = \sigma_i'|\sigma_{ik}$ meaning that σ_{ik} is the right most node of spine σ_i. Each element of σ_i is an index of a token in a sentence, or a dummy node $x(\lambda)$, where λ is a set of the left dependents of x. The figure above depicts an example of the configuration, where the i-th word in a sentence is written as w_i on the stack.

In the following, we say a right spine σ_i is *complete* if it does not contain any dummy nodes, while σ_i containing a dummy node is referred to as *incomplete*. Our transition system uses six actions, two of

which are **shift** actions and four are **reduce** actions. All actions are defined in Figure 2.

Shift Actions There are two kinds of shift actions: SHIFT and INSERT[3]. SHIFT moves a token from the top of the buffer to the stack. INSERT replaces a dummy node on the top of the stack with a token from the top of the buffer. This adds arcs from/to tokens connected to the dummy node. Note that this action can be performed for a configuration where $x(\lambda)$ is the top of σ_1 or λ is empty, in which case arcs (i, j) or $\cup_{k\in\lambda}(j, k)$ are not added. Resnik (1992) does not define this action, but instead uses a verification operation (Rule 9). One can view our INSERT action as a composition of two actions: SHIFT and a verification. We note that after these shift actions, the top element of the stack must be complete.

Reduce Actions Reduce actions create new arcs for subtrees on the stack. LEFT-PRED and RIGHT-PRED correspond to the predictions of the CFG counterpart. Figure 3 describes these transitions for minimal subtrees. LEFT-PRED assigns a dummy node x as the head of a (this corresponds to σ_{11}), while RIGHT-PRED creates x as a new right dependent. When we convert the resulting tree into a CNF, we can see that the difference between these two operations lies in the predicted parent node of a: LEFT-PRED predicts a nonterminal X(x), i.e., it predicts that the head of this subtree is the head of the predicting sibling node, while RIGHT-PRED predicts that the head is a. Note that different from CFG rules, we do not have to predict the actual sibling node; rather, we can abstract this predicted node as a dummy node x. A similar correspondence holds between the composition actions: RIGHT-COMP and LEFT-COMP.

We note that to obtain a valid tree, shift and reduce actions must be performed alternatively. We can prove this as follows: Let $c = (\sigma|\sigma_2|\sigma_1, \beta, A)$. Since reduce actions turn an incomplete σ_1 into a complete subtree, we cannot perform two consecutive reduce actions. Shift actions make σ_1 complete. After a shift action, we cannot perform INSERT since it requires σ_i to be incomplete; if we perform SHIFT, the top two elements on the stack become complete, but we cannot connect these two trees since the only way to connect two trees on the stack is composition, but this requires σ_2 to be incomplete.

Defining Oracle The oracle for a transition system is a function that returns a correct action given the current configuration and a set of gold arcs. It is typically used for training a parser (Nivre, 2008), but we define it to analyze the behavior of our system on treebank sentences.

First, we show that our system has the **spurious ambiguity**, and discuss its implications. Consider a sentence $a^\frown b^\frown c$, which can be parsed with two different action sequences as follows:

1. SHIFT → LEFT-PRED → INSERT → RIGHT-PRED → INSERT
2. SHIFT → LEFT-PRED → SHIFT → RIGHT-COMP → INSERT

The former INSERTs b at step 3, then RIGHT-PREDs to wait for a right dependent (c). The latter, on the other hand, SHIFTs b at step 3, then RIGHT-COMPs to combine two subtrees ($a^\frown x$ and b) to obtain a tree $a^\frown b^\frown x$. These ambiguities between action sequences and the resulting tree are referred to as the spurious ambiguity. Next, we analyze the underlying differences between these two operations. We argue that the difference lies in the form of the recognized constituency tree: The former RIGHT-PREDs at step 4, which means that it recognizes a constituency of the form $((a\ b)\ c)$, while the latter recognizes $(a\ (b\ c))$ due to its RIGHT-COMP operation. Therefore, the spurious ambiguity of our system is caused by the ambiguity of converting a dependency tree to a constituency tree. Recently, some transition systems have exploited similar ambiguities using *dynamic oracles* (Goldberg and Nivre, 2013; Sartorio et al., 2013; Honnibal et al., 2013). The same type of analysis might be possible for our system, but we leave it for future work; here we only present a *static oracle* and discuss its properties.

Since our system performs shift and reduce actions interchangeably, we need two functions to define the oracle. Let $c = (\sigma|\sigma_2|\sigma_1, \beta, A)$. The next shift action is determined as follows:

- INSERT: if $\sigma_1 = \langle\sigma_1'|i|x(\lambda)\rangle$ and $(i, j) \in A_g$ and j has no dependents in β (if i exists) or $\exists k \in \lambda; (j, k) \in A_g$ (otherwise).
- SHIFT: otherwise.

The next reduce action is determined as follows:

[3]We use small caps to refer to a specific action, e.g., SHIFT, while "shift" refers to an action type.

- LEFT-COMP: if $\sigma_2 = \langle \sigma_2' | i | x(\lambda) \rangle, \sigma_1 = \langle \sigma_{11}, \cdots \rangle$, σ_{11} has no dependents in β, and σ_{11} can be a left dependent of x: i's next dependent is the head of σ_{11} (if i exists) or $k \in \lambda$ and σ_{11} share the same head (otherwise).
- RIGHT-COMP: if $\sigma_2 = \langle \sigma_2' | i | x(\lambda) \rangle, \sigma_1 = \langle \sigma_{11}, \cdots \rangle$, σ_{11} has one more dependent in β, and σ_{11} can be insertable at the position of x: $(i, \sigma_{11}) \in A_g$ or $\exists k \in \lambda; (\sigma_{11}, k) \in A_g$.
- RIGHT-PRED: if $\sigma_1 = \langle \sigma_{11}, \cdots \rangle$ and σ_{11} has one more dependent in β.
- LEFT-PRED: otherwise.

Each condition checks whether we can obtain the gold dependency arcs after the transition. This oracle follows the strategy of "compose or insert when possible". As we saw in the example, sometimes INSERT and SHIFT both can be valid to recover the gold arcs; however, we always select INSERT. Sometimes the same ambiguity occurs between LEFT-COMP and LEFT-PRED or RIGHT-COMP and RIGHT-PRED, but we always prefer composition.

As we saw above, the spurious ambiguity of our system occurs when the conversion from a dependency tree to a constituency tree is not deterministic. This oracle has the property that its recognized constituency tree corresponds to the one that can be obtained by constructing all left-arcs first given a dependency tree. For example, in the above example for $a \overset{\curvearrowleft}{b} \overset{\curvearrowright}{c}$, we select action sequences 1 to recognize a constituency of $((a\ b)\ c)$. We can prove this property by showing that the algorithm always collects all left-arcs for a head before any right-arcs; Not doing INSERT or composition when possible means that we create a right-arc for a head when the left-arcs are not yet completed. We can also verify that this algorithm can parse all no-center-embedded sentences in a CNF converted in this manner with the stack depth never exceeding three, requiring non-constant memory for only center-embedded structures.

4 Memory Cost Analysis

To characterize our transition system, we compare it to other systems by observing the incurred memory cost during running oracle transitions for sentences on a set of typologically diverse languages. For this analysis, we aim to verify the language universality of the claim: naturally occurring sentences should be parsed with a left-corner parser with less required memory.

Settings We collect 18 treebanks from the CoNLL-X and 2007 shared tasks (Buchholz and Marsi, 2006; Nivre et al., 2007). Some languages were covered by both shared tasks; we use only 2007 data. We remove sentences with non-projective arcs (Nivre, 2008) or without any root nodes. We follow the common practice adding a dummy root token to each sentence. This token is placed at the end of each sentence, as in Ballesteros and Nivre (2013), since it does not change the cost on sentences with one root token on all systems.

We compare three transition systems: arc-standard, arc-eager, and left-corner. For each system, we perform oracle transitions for all sentences and languages, measuring the memory cost for each configuration defined as follows. For the arc-standard and left-corner systems, we use the number of elements on the stack. This arc-standard system uses the original formulation of Nivre (2003), connecting two items on the stack at the reduce action. For the arc-eager system, we use the number of connected components. The system can create a subtree at the beginning of a buffer, in which case we add 1 to the cost.

We run a static oracle for each system. For the left-corner system, we implemented the algorithm presented in Section 3. For the arc-standard and arc-eager systems, we implemented an oracle preferring reduce actions over shift, which can minimizes the memory cost.

Memory costs for general sentences For each language, we count the number of configurations for each memory cost during performing oracles on all sentences. In Figure 4, we show the cumulative frequencies of configurations having each memory cost (see solid lines in the figure). These lines can answer the question: What memory cost is required to cover X% of configurations when recovering all gold trees? Note that comparing absolute values are not meaningful since the minimal cost to construct an arc is different for each system, e.g., the arc-standard system requires at least two items on the stack,

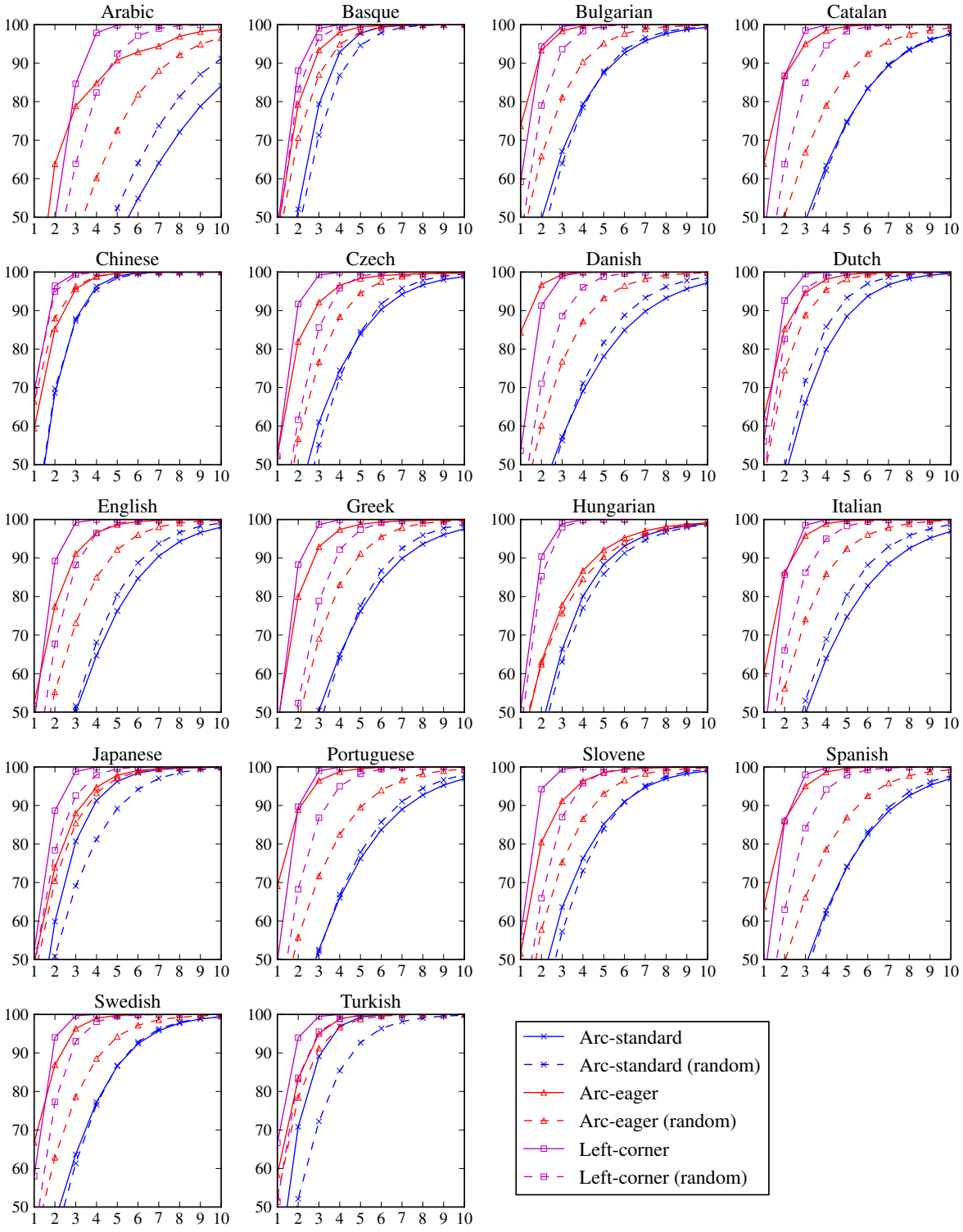


Figure 4: Crosslinguistic comparison of cumulative frequency of memory cost when processing all sentences for each system. For example, in Arabic, for the arc-eager system, around 90% of all configurations incurred memory cost of ≤ 5. Dotted lines (random) are results on the sentences, which are randomly reordered while preserving the graph structure and projectivity.

while the arc-eager system can create a right arc if the stack contains one element. Instead, we focus on the universality of each system's behavior for different languages.

As we discussed in section 2.2, the arc-standard system can process only left-branching structures with a constant memory, which are typical in head-final languages such as Japanese or Turkish, and we can

see this tendency. The system behaves poorly in many other languages.

The arc-eager and left-corner systems behave similarly for many languages, but we can see that there are some languages for which the left-corner system behaves similarly to other languages, while the arc-eager system requires larger cost; Arabic, Hungarian, or Japanese, for example. In fact, except Arabic, the left-corner system reaches 98% of configurations with a memory cost of ≤ 3, which indicates that the property of the left-corner system requiring less memory is more universal than that of other systems.

Comparison to randomized sentences One might wonder that the results above come from the nature of left-corner parsing reducing the stack size, not from the bias in language avoiding center-embedded structures. To partially answer this question, we conduct another experiment comparing oracle transitions on original treebank sentences and on wrong sentences. We create these wrong sentences by using the method from Gildea and Temperley (2007). We reorder words in each sentence by first extracting a directed graph then randomly reordering the children of each node while preserving projectivity. The dotted lines in Figure 4 denotes the results of randomized sentences for each system.

There are notable differences in required memory between original and random sentences for many languages. This result indicates that our system can parse with less memory for only naturally occurring sentences. For Chinese and Hungarian, the differences are subtle. However, the differences are also small for the other systems, which implies that these corpora have some biases on graphs reducing the differences.

5 Related Work and Discussion

To the best of our knowledge, parsing with a left-corner strategy has only been studied for constituency. Roark (2001) proposed a top-down parser for a CFG with a left-corner grammar transform (Johnson, 1998), which is essentially the same as left-corner parsing but enables several extensions in a unified framework. Roark et al. (2009) studied the psychological plausibility of Roark's parser, observing that it fits well to human reading time data. Another model with a left-corner strategy is Schuler et al. (2010): they observed that the transformed grammar of English requires only limited memory, proposing a finite state approximation with a hierarchical hidden Markov model. This parser was later extended by van Schijndel and Schuler (2013), which defined a left-corner parser for constituency with shift and reduce actions. In fact, they used the same kind of actions as our transition system: shift, insert, predict, and composition. Though they did not mentioned explicitly, we showed how to construct a left-corner parsing algorithm with these actions by decomposing the push-down recognizer of Resnik (1992). These are examples of broad-coverage parsing models with cognitively plausibility, which has recently received considerable attention in interdisciplinary research on psycholinguistics and computational linguistics (Schuler et al., 2010; Keller, 2010; Demberg et al., 2013).

Differently from previous models, our target is dependency. A dependency-based cognitively plausible model is attractive, especially from a crosslinguistical viewpoint. Keller (2010) argued that current models only work for English, or German in few exceptions, and the importance of crosslinguistically valid models of human language processing. There has been some attempts to use a transition system for studying human language processing (Boston and Hale, 2007; Boston et al., 2008), so it is interesting to compare automatic parsing behaviors with various transition systems to human processing.

We introduced a dummy node for representing a subtree with an unknown head or dependent. Recently, Menzel and colleagues (Beuck and Menzel, 2013; Kohn and Menzel, 2014) have also studied dependency parsing with a dummy node. While conceptually similar, the aim of introducing a dummy node is different between our approach and theirs: We need a dummy node to represent a subtree corresponding to that in Resnik's algorithm, while they introduced it to confirm that every dependency tree on a sentence prefix is fully connected. This difference leads to a technical difference; a subtree of their parser can contain more than one dummy node, while we restrict each subtree to containing only one dummy node on a right spine.

Our experiments in section 4 can be considered as a study on functional biases existing in language or language evolution (Jaeger and Tily, 2011). In computational linguistics, Gildea and Temperley (2007; 2010) examined the bias on general sentences called *dependency length minimization* (DLM), which

argues that grammar should favor the dependency structures that reduce the sum of dependency arc lengths. They reordered English and German treebank sentences with various criteria: original, random with projectivity, and optimal that minimizes the sum of dependency lengths. They observed that the word order of English fits very well to the optimal ordering, while German does not. We examined the universality of the bias to reduce memory cost for left-corner parsing. Although we cannot compare the incurred cost with the *optimal* reordered sentences, our results on original sentences, in which there are few configurations requiring the stack depth ≥ 4, suggest the bias to avoid center-embedded structures is language universal. It will be interesting to analyze in more detail the relationships between DLM and the bias of our system since the two biases are not independent, e.g., center-embed structures typically appear with longer dependencies. Are there languages that do not hold DLM while requiring less memory, or vice versa? For these analyses, we might have to take care of the grammar construction, e.g., there are several definitions for coordination structures for dependency grammars (Popel et al., 2013). The functional views discussed above might shed some light on the desired construction for these cases.

6 Conclusion

We have pointed out that the memory cost on current transition systems for dependency parsing do not coincide with observations in people, proposing a system with a left-corner strategy. Our crosslinguistical analysis confirms the universality of the claim that people avoid generating center-embedded sentences, which also suggests that it is worthy for crosslinguistical studies of human language processing.

As a next stage, we are seeking to train a parser model as in other transition systems with a discriminative framework such as a structured perceptron (Zhang and Nivre, 2011; Huang and Sagae, 2010). A parser with our transition system might also be attractive for the problem of grammar induction, where recovering dependency trees are a central problem (Klein and Manning, 2004), and where some linguistic biases have been exploited, such as reducibility (Mareček and Žabokrtský, 2012) or acoustic cues (Pate and Goldwater, 2013). Recently, Cohen et al. (2011) showed how to interpret shift-reduce actions as a generative model; combining their idea and our transition system might enable the model to exploit memory biases that exist in natural sentences.

Finally, dependency grammars are suitable for treating non-projective structures. Extensions for transition systems have been proposed to handle non-projective structures with additional actions (Attardi, 2006; Nivre, 2009). Although our system cannot handle non-projective structures, a similar extension might be possible, which would enable a left-corner analysis for non-projective structures.

Acknowledgements

We thank Pontus Stenetorp and anonymous reviewers for their valuable feedbacks on a preliminary version of this paper.

References

Steven Abney and Mark Johnson. 1991. Memory requirements and local ambiguities of parsing strategies. *Journal of Psycholinguistic Research*, 20(3):233–250.

Giuseppe Attardi. 2006. Experiments with a multilanguage non-projective dependency parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 166–170, New York City, June. Association for Computational Linguistics.

Miguel Ballesteros and Joakim Nivre. 2013. Going to the roots of dependency parsing. *Computational Linguistics*, 39(1):5–13.

Niels Beuck and Wolfgang Menzel. 2013. Structural prediction in incremental dependency parsing. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 7816 of *Lecture Notes in Computer Science*, pages 245–257. Springer Berlin Heidelberg.

Marisa Ferrara Boston and John T. Hale. 2007. Garden-pathing in a statistical dependency parser. In *Proceedings of the Midwest Computational Linguistics Colloquium*, West Lafayette, IN. Midwest Computational Linguistics Colloquium.

Marisa Ferrara Boston, John T. Hale, Umesh Patil, Reinhold Kliegl, and Shravan Vasishth. 2008. Parsing costs as predictors of reading difficulty: An evaluation using the Potsdam Sentence Corpus. *Journal of Eye Movement Research*, 2(1):1–12.

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

Shay B. Cohen, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Exact inference for generative probabilistic non-projective dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1245, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Nelson Cowan. 2001. The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences*, 24(1):87–114.

Vera Demberg, Frank Keller, and Alexander Koller. 2013. Incremental, predictive parsing with psycholinguistically motivated tree-adjoining grammar. *Computational Linguistics*, 39(4):1025–1066.

Daniel Gildea and David Temperley. 2007. Optimizing grammars for minimum dependency length. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 184–191, Prague, Czech Republic, June. Association for Computational Linguistics.

Daniel Gildea and David Temperley. 2010. Do grammars minimize dependency length? *Cognitive Science*, 34(2):286–310.

Yoav Goldberg and Joakim Nivre. 2013. Training deterministic parsers with non-deterministic oracles. *TACL*, 1:403–414.

Katsuhiko Hayashi, Taro Watanabe, Masayuki Asahara, and Yuji Matsumoto. 2012. Head-driven transition-based parsing with top-down prediction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 657–665, Jeju Island, Korea, July. Association for Computational Linguistics.

Matthew Honnibal, Yoav Goldberg, and Mark Johnson. 2013. A non-monotonic arc-eager transition system for dependency parsing. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 163–172, Sofia, Bulgaria, August. Association for Computational Linguistics.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1077–1086, Uppsala, Sweden, July. Association for Computational Linguistics.

T.Florian Jaeger and Harry Tily. 2011. On language utility: Processing complexity and communicative efficiency. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(3):323–335.

P. N. Johnson-Laird. 1983. *Mental models: towards a cognitive science of language, inference, and consciousness*. Harvard University Press, Cambridge, MA, USA.

Mark Johnson. 1998. Finite-state approximation of constraint-based grammars using left-corner grammar transforms. In Christian Boitet and Pete Whitelock, editors, *COLING-ACL*, pages 619–623. Morgan Kaufmann Publishers / ACL.

Frank Keller. 2010. Cognitively plausible models of human language processing. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 60–67, Uppsala, Sweden, July. Association for Computational Linguistics.

Kotaro Kitagawa and Kumiko Tanaka-Ishii. 2010. Tree-based deterministic dependency parsing — an application to nivre's method —. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 189–193, Uppsala, Sweden, July. Association for Computational Linguistics.

Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 478–485, Barcelona, Spain, July.

Arne Kohn and Wolfgang Menzel. 2014. Incremental predictive parsing with turboparser. In *Proceedings of the ACL 2014 Conference Short Papers*, Baltimore, USA, June. Association for Computational Linguistics.

Marco Kuhlmann, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Dynamic programming algorithms for transition-based dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 673–682, Portland, Oregon, USA, June. Association for Computational Linguistics.

David Mareček and Zdeněk Žabokrtský. 2012. Exploiting reducibility in unsupervised dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 297–307, Jeju Island, Korea, July. Association for Computational Linguistics.

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

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In Frank Keller, Stephen Clark, Matthew Crocker, and Mark Steedman, editors, *Proceedings of the ACL Workshop Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain, July. Association for Computational Linguistics.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

Joakim Nivre. 2009. Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 351–359, Suntec, Singapore, August. Association for Computational Linguistics.

John K. Pate and Sharon Goldwater. 2013. Unsupervised dependency parsing with acoustic cues. *TACL*, 1:63–74.

Martin Popel, David Mareček, Jan Štěpánek, Daniel Zeman, and Zdeněk Žabokrtský. 2013. Coordination structures in dependency treebanks. In *ACL*, pages 517–527.

Philip Resnik. 1992. Left-corner parsing and psychological plausibility. In *COLING*, pages 191–197.

Brian Roark, Asaf Bachrach, Carlos Cardenas, and Christophe Pallier. 2009. Deriving lexical and syntactic expectation-based measures for psycholinguistic modeling via incremental top-down parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 324–333, Singapore, August. Association for Computational Linguistics.

Brian Edward Roark. 2001. *Robust Probabilistic Predictive Syntactic Processing: Motivations, Models, and Applications*. Ph.D. thesis, Providence, RI, USA. AAI3006783.

Francesco Sartorio, Giorgio Satta, and Joakim Nivre. 2013. A transition-based dependency parser using a dynamic parsing strategy. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 135–144, Sofia, Bulgaria, August. Association for Computational Linguistics.

William Schuler, Samir AbdelRahman, Tim Miller, and Lane Schwartz. 2010. Broad-coverage parsing using human-like memory constraints. *Computational Linguistics*, 36(1):1–30.

Marten van Schijndel and William Schuler. 2013. An analysis of frequency- and memory-based processing costs. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 95–105, Atlanta, Georgia, June. Association for Computational Linguistics.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA, June. Association for Computational Linguistics.

Data-driven Measurement of Child Language Development with Simple Syntactic Templates

Shannon Lubetich
Pomona College
Claremont, CA 91711
shannon.lubetich@pomona.edu

Kenji Sagae
Institute for Creative Technologies
University of Southern California
Los Angeles, CA 90089
sagae@ict.usc.edu

Abstract

When assessing child language development, researchers have traditionally had to choose between easily computable metrics focused on superficial aspects of language, and more expressive metrics that are carefully designed to cover specific syntactic structures and require substantial and tedious labor. Recent work has shown that existing expressive metrics for child language development can be automated and produce accurate results. We go a step further and propose that measurement of syntactic development can be performed automatically in a completely data-driven way without the need for definition of language-specific inventories of grammatical structures. As a crucial step in that direction, we show that four simple feature templates are as expressive of language development as a carefully crafted standard inventory of grammatical structures that is commonly used and has been validated empirically.

1 Introduction

Although child language has been the focus of much study, our understanding of first language acquisition is still limited. In attempts to measure child language development over time, several metrics have been proposed. The most commonly used metric is Mean Length of Utterance, or MLU (Brown, 1973), which is based on the number of morphemes per utterance. The main appeal of MLU is that it can be easily computed automatically, given machine-readable transcripts. Although MLU values may not be meaningful across languages, the general approach is suitable for analysis within different languages. However, MLU's ability to track language development from age four has been questioned (Klee and Fitzgerald, 1985; Scarborough, 1990), and its usefulness is still the subject of debate (Rice et al., 2010).

Several metrics based on the usage of grammatical structure have been proposed as more sensitive to changes in language over a wider range of ages (Scarborough, 1990; Lee and Canter, 1971; Fletcher and Garman, 1988). These metrics continue to show score increases where MLU plateaus, but their increased expressivity is typically associated with two severe drawbacks. The first is that their use for computation of language development scores involves identification of several specific grammatical structures in child language transcripts, a process that requires linguistic expertise and is both time-consuming and error-prone. This issue has been addressed by recent work that shows that current natural language processing techniques can be applied to automate the computation of these metrics, removing the bottleneck of manual labor (Sagae et al., 2005; Roark et al., 2007; Sahakian and Snyder, 2012). The second drawback is that these measures are language-specific, and development of a measure for a specific language requires deep expertise and careful design of an inventory of grammatical structures that researchers believe to be indicative of language development. Going beyond previous work, which addressed the first drawback of traditional metrics for child language development, we address the second, paving the way for a language-independent methodology for tracking child language development that is as expressive as current language-specific alternatives, but without the need for carefully constructed inventories of grammatical structures.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2151–2160, Dublin, Ireland, August 23-29 2014.

The specific hypothesis we address in this paper is whether a fully-data driven approach that uses only a few simple feature templates applied to syntactic dependency trees can capture the same information as the well-known Index of Productive Syntax, or IPSyn (Scarborough, 1990). In contrast to previous work that showed that the computation of IPSyn scores can be performed automatically by encoding each of the 60 language structures in a language-specific inventory (e.g. wh-questions with auxiliary inversion, propositional complements, conjoined sentences) as complex patterns over parse trees, we propose that child language development can instead be measured automatically in a way that is fully data-driven and can be applied to many languages for which accurate dependency parsers are available, without relying on carefully constructed lists of grammatical structures or complex syntactic patterns in each language. Specifically, we examine two hypotheses: (1) counts of features extracted from syntactic parse trees using only simple templates are at least as expressive of changes in language development as the Index of Productive Syntax (Scarborough, 1990), an empirically validated metric based on an inventory of grammatical structures derived from the child language literature; and (2) these parse tree features can be used to model language development without the use of an inventory of specific structures, assuming only the knowledge that in typically developing children the level of language development is correlated with age. We emphasize that the goal of this work is not to develop yet one more way to compute IPSyn scores automatically, but to show empirically that lists of grammatical structures such as those used to compute IPSyn are not essential to measure syntactic development in children.

In this paper, we start by reviewing IPSyn and previous work on automatic IPSyn scoring based on manually crafted syntactic patterns in section 2. Using a similar approach, we validate the language development curves observed by Scarborough (1990) in the original IPSyn study. In section 3 we show how IPSyn scores can be computed in an entirely different, fully data-driven way, using a support vector regression. In section 4 we examine how this data-driven framework can be used to track language development in the absence of a metric such as IPSyn, which allows for application of this approach to languages other than English. We discuss related work in section 5, and conclude in section 6.

2 Index of Productive Syntax (IPSyn)

The Index of Productive Syntax (Scarborough, 1990) evaluates a child's linguistic development by analyzing a transcript of utterances and awarding points when certain syntactic and morphological structures are encountered. The end result is a number score ranging from 0 to 120, with a higher score corresponding to the presence of more complex grammatical structures, and thus further linguistic development. IPSyn was designed to be more sensitive to language changes after age 3 than the more common Mean Length of Utterance (MLU) (Brown, 1973), which fails to account for the fact that children's speech increases in complexity even after utterances stop increasing in length.

IPSyn scores are calculated by analyzing a transcript of 100 utterances of a child's speech, and awarding points to specific language structures encountered. There are 60 forms in total from four categories of noun phrases, verb phrases, questions and negations, and sentence structures. Each form is awarded 0 points if not encountered, 1 point if found once in a transcript, and 2 points if found at least twice. This sums to a total ranging between 0 and 120 points. Scarborough (1990) motivates the use of this specific inventory of 60 forms by stating that they "have been shown to occur in preschool language production in innumerable studies of language acquisition during the past 25 years," highlighting that the task of generating such an inventory and performing empirical validation for additional languages requires considerable expertise and is far from trivial.

2.1 Automating IPSyn

In support of empirical testing of our first hypothesis–that features extracted from parse trees using only simple feature templates are as expressive of child language development as the carefully constructed inventory of grammatical structures in IPSyn–we first implemented an automated version of IPSyn following Sagae et al. (2005), who showed that this task can be performed nearly at the level of trained human experts. This allows us to generate IPSyn scores for a large set of child language transcripts. Our implementation differs from previous work mainly in that it uses only the tools provided in the CLAN

software suite (MacWhinney, 2000), which were designed specifically for analysis of child language transcripts, instead of the Charniak (2000) parser, which was used by Sagae et al. and later by Hassanali et al. (2014) in a more recent implementation of the same general approach.

We evaluated our implementation using the set of 20 manually scored transcripts described by Sagae et al. as Set A, and subsequently used to evaluate the implementation of Hassanali et al. Three transcripts were used as development data, following Sagae et al. The mean absolute difference between manually generated and automatically generated scores was 3.6, which is very similar to what has been reported by Hassanali et al. and by Sagae et al. (3.05 and 3.7, respectively) for the same set of transcripts. Given the possible score differences in manual scoring reported by Scarborough (1990) and the small number of transcripts used for testing, the differences observed among the automatic systems are not meaningful. In fact, in examining our development data, we found multiple errors in the manual coding, causing point discrepancies when our system produced correct results. This highlights the difficulty of performing this scoring task manually, and raises the question of whether automatic scoring has in fact surpassed the reliability of manual scoring. That three different implementations of IPSyn appear to perform comparably suggests this might be the case. We leave an empirical investigation of this question to future work.

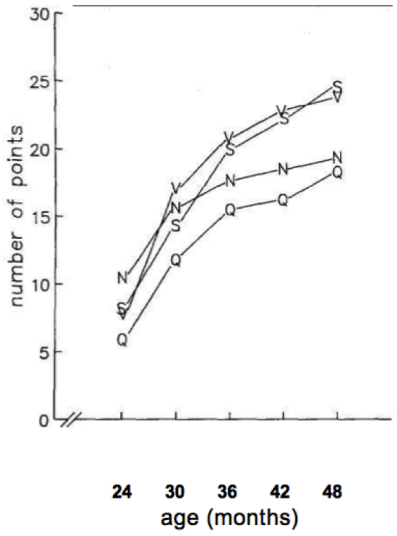
3 From automatic IPSyn to data-driven IPSyn

The fully automatic way of computing IPSyn scores described above in section 2.1, paired with a sufficiently large amount of child language transcript data, gives us a way to test the hypothesis mentioned in the beginning of section 2.1, that simple features of parse trees are as expressive as the hand-crafted IPSyn language structure inventory. We did this by first creating several 100-utterance transcripts from existing child language transcripts, then automatically assigning them IPSyn scores, and using these scores as targets to be learned from features extracted from the corresponding 100-utterance transcripts. Details of the data and learning approach used for this experiment, as well as empirical results, are described in the remainder of this section.

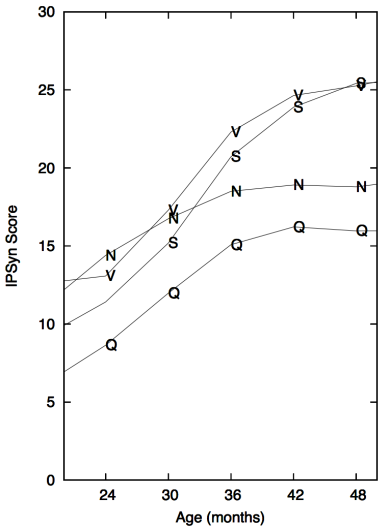
3.1 Generating IPSyn data

To obtain enough child language transcripts in a wide range of ages to test our hypothesis, we turned to the CHILDES database. To generate training and development sets for our experiments, we used transcripts from CHILDES of 14 different children with ages ranging from 1 year 1 month to 8 years. Because each application of IPSyn requires only 100 child utterances, transcripts were split, producing a total of 593 transcripts, each containing 100 utterances. The 14 children in our dataset came from the following CHILDES corpora: Brown, MacWhinney, Sachs and Warren. The reason for choosing these corpora is that they were quickly identified as containing spontaneous natural interactions, as opposed to reading or specific games and activities designed to elicit a certain kind of language production. It is likely that other corpora in CHILDES would also suit our purposes, but the data in these four corpora was sufficient for our experiments. Each of the 593 transcripts was assigned an IPSyn score automatically. From the Brown, MacWhinney and Sachs corpora, we used transcripts from a total of four children (Adam from Brown, Mark and Ross from MacWhinney, and Naomi from Sachs), from whom language data was collected over several years. Transcripts from these three corpora, 572 in total, served as our training set. The Warren corpus includes data from ten children with ages ranging from 1;6 to 6;2 (that is, 1 year and 6 months to 6 years and 2 months, using the commonly accepted age notation for this type of data), from which we created 21 transcripts that served as our development set.

The complete set of 593 transcripts with IPSyn scores gives us the opportunity to verify whether the language development curves observed by Scarborough (1990) averaged over 75 transcripts in the original IPSyn study matches curves produced from averaging results from 593 transcripts from entirely different subjects. Figure 1 shows a side-by-side comparison between the original figure from (Scarborough, 1990) and a corresponding figure generated with our automatically scored transcripts. Although not identical, the two figures are remarkably similar, reflecting that aspects of the emergence of grammar in child language development are shared across children, and that IPSyn captures some of these aspects.



(a) Original IPSyn study.



(b) Automatically generated.

Figure 1: Comparison between the IPSyn development curves for the four subscales in (a) the 75 transcripts in the original IPSyn study (reproduced from (Scarborough, 1990)), and (b) our set of 593 transcripts scored automatically.

Finally, we used the Garvey corpus to generate a test set. This corpus includes data from 48 different children with ages ranging from 2;10 to 5;7, from which we extracted 60 transcripts covering all 48 children and the full range of ages in the corpus. No data from the 48 children in the Garvey corpus, which we used as a test set, were used for training or development of the models used in our experiments.

3.2 A regression model for IPSyn

Given 593 pairs of transcript and IPSyn score, we approached the task of learning a data-driven model for IPSyn scoring as one of regression. For each transcript, a set of features is extracted, and the IPSyn score is associated with that feature vector. The features extracted from the transcripts followed four templates, described in the next subsection. If an accurate function for predicting IPSyn scores from these feature vectors can be learned, our hypothesis that these features are at least expressive enough to track child language development as well as the inventory of IPSyn structures is confirmed. To learn our model, we used the SVM Light[1] implementation of support vector regression (Drucker et al., 1997).

3.3 Features

An important step in learning a regression model for IPSyn is choosing what features to use. To support our goal of language independence, we decided not to consider language specific features that have been shown to be useful in this task but are language dependent[2], and opted instead to see whether the use of only simple parse tree features would be sufficient. The only prerequisite for extraction of our feature set is that each transcript must be parsed to produce a syntactic dependency tree. We used the CLAN tools for morphology analysis (MOR), part-of-speech tagging (POST) and parsing (MEGRASP)[3], since it is straightforward to process CHILDES transcripts using those, and they provide high-accuracy analyses for child language transcripts. The accuracy of the MEGRASP dependency parser for child utterances in English is estimated to be close to 93% (Sagae et al., 2010).

All of the features used in our model are extracted from parse trees according to four simple classes that target the following information:

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

[2] This is in contrast to, for example, the related work of Sahakian and Snyder (2012), which we discuss in section 5.

[3] Models for MOR and POST are available for a wide variety of languages. Models for MEGRASP are available only for English and Japanese, but our data-driven approach is not tied to any specific tagger or parser.

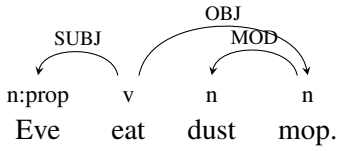


Figure 2: A dependency tree generated with part-of-speech and grammatical relation information.

Part-of-speech tags: The first type of feature we used is simply the part-of-speech tag of each word. This can be thought of as a bag of part-of-speech tags. We intentionally avoided the commonly used bag-of-words, because our goal is to obtain a model that tracks changes in syntax structure, not content. Although it is highly likely that lexical features would be very informative in this learning task, they would be useful for the wrong reason: our model is intended to target the emergence of *syntax*, and not what children talk *about* at different ages. We note, however, that as with the Penn Treebank tagset, the tags used by MOR and POST also reflect morphology, so that information is accounted for. The full tag set is listed in (MacWhinney, 2000).

Grammatical relations: The second feature class in our model is a bag of dependency labels, where each label correspond to a grammatical relation that holds between two words in the dependency tree (the head word and the dependent word). The full set of grammatical relations is listed in (Sagae et al., 2010).

Head-dependent part-of-speech pairs: Our third feature class is based on pairs of part-of-speech tags, where each pair corresponds to a bilexical dependency relation in the parse tree, and one of the tags comes from the head in the dependency, and the other tag comes from the dependent.

Head-relation-dependent triples: The last feature class is similar to the head-dependent pairs described above, but also including the dependency label that indicates the grammatical relation that holds between the head and dependent words. Features in this class are then triples composed of a head part-of-speech tag, a dependent part-of-speech tag, and a dependency label.

As an example, given the parse tree shown in Figure 2, the following features would be extracted:

```
n:prop v n n
SUBJ OBJ MOD
v_n:prop v_n n_n
v_n:prop_SUBJ v_n_OBJ n_n_MOD
```

Features are extracted for every tree in each transcript. Because our goal is to measure grammatical development in child language, these four feature templates were designed to capture the grammatical relations represented in dependency trees, while leaving out the content reflected in specific lexical items. While the content of what is said may be related to language development, our features are intended to focus on syntactic information, covering exactly each of the labeled arcs and the part-of-speech tags in a dependency tree (Figure 2) with the words removed. We also experimented with part-of-speech tag bigrams (pairs of adjacent part-of-speech tags), and dependency chains formed by two dependency arcs. The final choice of the four templates described above was based on results obtained on development data.

3.4 Data-driven IPSyn evaluation

We trained a support vector regression model using our training set of 572 transcripts, using a polynomial kernel and tuning the degree d and the regularization metaparameter C on the development set. While the default C and d values resulted in a mean absolute error of 6.6 points in the score predictions in the development set, setting $C = 1$ and $d = 3$ resulted in a mean absolute error of 4.1 on the development set. We used these values for the rest of our experiments. The mean absolute error obtained on our

test set of 48 children (60 transcripts) not used in training or tuning of the system was 3.9. When applying our regression model to the manually scored set of 20 transcripts used by Sagae et al. (2005), the mean absolute difference was 4.2 from the scores computed automatically using the approach in section 2.1, and 5.4 from the manually computed scores, which we consider our gold standard target. Compared to these manually computed scores, the absolute difference of 5.4 is higher than what we obtained using carefully designed templates based on the IPSyn inventory, but still within the range of variability expected for trained human scorers (Scarborough, 1990). It is important to keep in mind that the goal of this experiment was not to improve on the accuracy of previous automatic scoring programs, which work quite differently by listing manually crafted patterns over parse trees, but to show that a scoring function can be learned in a data-driven way, without manually crafted patterns. The results obtained with our regression model do confirm our hypothesis that simple features extracted from parse trees are enough for tracking child language development in the same way as the much more complex patterns included in IPSyn.

4 Age prediction

Given the ability of our data-driven approach to approximate IPSyn scores, confirming that a regression approach with parse tree features is capable of capturing the progression of language development, we now turn to the question of whether the same type of data-driven framework can be used to track child language development without the need for a metric such as IPSyn.

Assuming only that language acquisition progresses monotonically over time, we can apply the same data-driven regression approach to predict a child's age given a language sample. This task was approached recently by Sahakian and Snyder (2012), who used an ensemble of existing metrics with a few additional features. Unlike in our approach, Sahakian and Snyder do include lexical features and hand-selected patterns in the form of an existing metric (D-level). They make the reasonable argument that the task of age prediction is child-dependent, and that prediction across children would not make sense due to individual variation in the rate of language development. Following Sahakian and Snyder, we first approach age prediction as a child-specific task, but then discuss the application of our regression models for other children than those used for training.

4.1 Child-specific age prediction

To determine whether our data-driven regression approach can model the development of individual children at the level where accurate age predictions can be made, we used the same feature templates described in section 3.3, but trained a regression model to predict age in months, rather than IPSyn scores. Because this is a child-specific prediction task, we train separate regression models for each child. We tested our age predictions using 10-fold cross-validation for three children from three different CHILDES corpora (Adam from Brown, Ross from MacWhinney and Naomi from Sachs) for whom enough data was available over a wide enough range of ages. In each case the regression approach performed well. Table 1 shows the mean absolute error in months for each child, and the Pearson r for the correlation between predicted age and actual age.

Child (corpus)	Mean Abs Err	Pearson (r)
Adam (Brown)	2.5	0.93
Ross (MacWhinney)	3.7	0.84
Naomi (Sachs)	3.1	0.91

Table 1: Regression results for single corpus age prediction ($p < 0.0001$ for all r values.)

Perhaps more interesting than the strong correlations between actual age and predicted age for each of the individual corpora is a comparison of these correlations to correlations between age and MLU, and age and IPSyn score. One main general criticism of MLU is that it fails to correlate well with age for older children (around three to four years old). More detailed metrics such as IPSyn are believed to have better correlation with age after that point. We do observe this situation in our data. Interestingly, our

predicted age scores have much stronger correlations to actual age for older children, which suggests that our regression approach with simple syntactic features is more expressive in tracking syntactic development in older children than either MLU or IPSyn. This is shown in Table 2, which contains Pearson r correlation coefficients for age and MLU, age and IPSyn, and age and predicted age using our regression approach.

Child (corpus)	MLU r	IPSyn r	Regression r
Adam (Brown)	$0.37^{\dagger}$	$0.53^{\dagger}$	$0.85^{\dagger}$
Ross (MacW)	0.19	0.34^{*}	$0.79^{\dagger}$
Naomi (Sachs)	0.27	0.52	$0.82^{\dagger}$

Table 2: Pearson correlation coefficients between actual age and MLU, actual age and IPSyn score, and actual age and predicted age, for children at least three years and four months old. $^{\dagger}p < 0.0001$. $^{*}p < 0.05$.

The results shown in Table 2 confirm that features extracted from parse trees alone can offer substantially better prediction of age for individual children than MLU or even IPSyn scores. This is not surprising, given that weights for these features are optimized to predict age using data from the specific child and discriminative learning, but it does show that these features offer enough resolution to track syntactic development in child language, confirming our second hypothesis.

4.2 Pilot experiment with Japanese language data

A great advantage of using a data-driven framework based on simple feature templates rather than a traditional approach for measuring syntactic development with manually crafted lists of grammatical structures is that the data-driven approach is, in principle, language-independent. The same features described in section 2.1 could be extracted from dependency parse trees in any language, assuming only that these dependency trees can be produced automatically. Syntactic dependency parsers and treebanks are in fact available for a variety of languages (Buchholz and Marsi, 2006; Nivre et al., 2007). Although the availability of treebanks that include child language samples is certainly desirable, it is not clear whether it is strictly required in order to generate the syntactic structures used in our approach. While Sagae et al. (2005) and Hassanali et al. (2014) obtained high levels of accuracy in IPSyn scoring using the Charniak (2000) parser with a model trained on the Wall Street Journal portion of the Penn Treebank (Marcus et al., 1993), we have not verified the effects of parser errors in our data-driven approach. Of course, the language independence claim applies only to the ability to measure syntactic development within different languages, and direct numerical comparisons across languages are not meaningful, since the available syntactic annotations for different languages follow different conventions and syntactic theories.

Although a full empirical validation of our regression approach in other languages is left as future work, we performed a pilot experiment with a single Japanese child that suggests our findings may be robust across languages. We used transcripts from the child Ryo, from the Miyata corpus of the Japanese section of the CHILDES database[4]. We extracted 80 transcripts of 100 utterances each, covering ages 1;10 (22 months) to 3;0 (36 months). These transcripts were analyzed with the Japanese version of the MEGRASP parser for CHILDES transcripts at an estimated accuracy of 93% (Miyata et al., 2013). Using the exact same experimental settings and feature templates as for English, we performed a 10-fold cross-validation for age prediction using the Japanese data. We obtained a strong correlation between predicted age and actual age, with $r = 0.82$ ($p < 0.0001$). Although this value is slightly lower than the values in Table 1 for English, the range of target values (age in months) is more compressed. Although this experiment included only one child, it does suggest that our approach may work well for Japanese.

5 Related work

Within the literature on assessment of child language development, the metric most closely related to our work is the Index of Productive Syntax (Scarborough, 1990), which we discussed in more detail in

[4]http://childes.psy.cmu.edu/data/EastAsian/Japanese/Miyata/

section 2, and used as a target for data-driven learning. Other traditional metrics include Developmental Sentence Scoring (Lee and Canter, 1971), Language Assessment Remediation and Screening Procedure (Fletcher and Garman, 1988), and D-level (Parisse and Le Normand, 1987) all of which share with IPSyn the reliance on a hand-crafted inventory of grammatical structures meant to be identified manually in transcribed child language samples.

Each of these metrics for child language development, along with the Mean Length of Utterance (Brown, 1973), can be computed semi-automatically using the Computerized Profiling system (Long et al., 2004). Although fully automatic computation with Computerized Profiling produces levels of reliability lower than that of manual scoring, the system can be used with human intervention to produce results of higher quality. More closely related is the work of Sagae et al. (2005) on automating IPSyn using patterns extracted from automatic parse trees. The work we describe in section 2.1 is closely based on that of Sagae et al., which we use as a way to validate our data-driven approach.

Roark et al. (2007) examined the ability of several automatically computed syntactic complexity metrics to discriminate between healthy and language impaired subjects. Among other metrics, Roark et al. used Frazier scoring (Frazier, 1985) and Yngve scoring (Yngve, 1960), which are more commonly associated with processing difficulty than with emergence of syntax in child language development, but are related to our approach in that they are based on straightforward generic features of parse trees (depth, node count), like our counts of grammatical relation labels. Finally, Sahakian and Snyder (Sahakian and Snyder, 2012) have also approached the problem of learning automatic metrics for child language development using a regression approach. Their focus, however, was on the combination of the existing metrics MLU, mean depth of tree (similar to Yngve scoring mentioned above) and D-level, along with a few hand-picked features (counts of certain closed-class words, ratio of function words to content words, and average word frequency), to achieve better discrimination than any of these metrics or features alone. A key difference between our approach and that of Sahakian and Snyder is that their approach builds on and assumes the existence of a metric such as D-level, which, like IPSyn, includes a carefully designed language-dependent inventory of language structures, while we use only simple feature templates applied to parse trees. In addition, they include vocabulary-centric features, while we explicitly avoid vocabulary features, focusing on structural features. It is possible that Sahakian and Snyder's approach would benefit from the parse tree features of our approach, either by using the features directly, or by taking a score obtained by our approach as an additional feature in theirs.

6 Conclusion and future work

We presented a framework for assessment of syntactic development in child language that is completely data-driven, and unlike traditional metrics such as IPSyn, LARSP and D-level, does not rely on a language-dependent inventory of language structures chosen specifically for the task. Instead, our approach is based on the application of support vector regression with simple features extracted from syntactic parse trees. In our experiments we used dependency parses produced by the MEGRASP parser for CHILDES transcripts, but it is likely that other modern dependency and constituent parsers would provide similar results. We showed that our framework is capable of learning IPSyn scores, and that for individual children it can model syntactic development well after MLU and IPSyn scores fail to correlate with age.

Having shown that the feature templates described in section 2.1 are as expressive as the inventory of grammatical structures in IPSyn at tracking language development, and that syntactic development of individual children can be modeled using our data-driven framework in complete absence of an existing metric such as IPSyn, it is interesting to consider the applicability of this framework to different languages for which child language development metrics have not been developed or are not widely used. One possible way to do this is to train several age regression models representing different development profiles. In most practical scenarios, the child's age is known and would not need to be predicted by a model. By predicting age with several different models and selecting the one that most closely predicts the child's actual age, a language development profile matching the child can be found. This could be used, for example, in diagnosis of language impairment. In this paper we established only the expressive

power of regression using simple syntactic features, and the application of this approach to practical tasks is left as an interesting direction for future work.

A related direction for future work is the application of this method for assessment of syntactic development in languages other than English. Given the availability of child language data in various languages (MacWhinney, 2000) and recent progress in syntactic analysis for many of these languages (Buchholz and Marsi, 2006; Nivre et al., 2007), we are optimistic about the applicability of our approach to other languages. Preliminary results using data from one Japanese child suggest that the same set of simple feature templates can be used to track language development in Japanese.

Acknowledgments

We thank the anonymous reviewers for insightful suggestions. This work was partly supported by the National Science Foundation under grants 1263386 and 1 219253 and by the U.S. Army. Any opinion, content or information presented does not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

References

Roger Brown. 1973. *A first language: The early stages*. George Allen & Unwin.

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

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, pages 132–139, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Harris Drucker, Chris, Burges L. Kaufman, Alex Smola, and Vladimir Vapnik. 1997. Support vector regression machines. In *Advances in Neural Information Processing Systems 9*, volume 9, pages 155–161.

Paul Fletcher and Michael Garman. 1988. LARSPing by numbers. *British Journal of Disorders of Communication*, 23(3):309–321.

L. Frazier. 1985. Syntactic complexity. In D. R. Dowty, L. Karttunen, and A. M. Zwicky, editors, *Natural Language Parsing: Psychological, Computational, and Theoretical Perspectives*, pages 129–189. Cambridge University Press, Cambridge.

Khairun-Nisa Hassanali, Yang Liu, Aquiles Iglesias, Thamar Solorio, and Christine Dollaghan. 2014. Automatic generation of the index of productive syntax for child language transcripts. *Behavior Research Methods*, 46:254–262.

Thomas Klee and Martha Deitz Fitzgerald. 1985. The relation between grammatical development and mean length of utterance in morphemes. *Journal of Child Language*, 12:251–269, 6.

Laura L. Lee and Susan M. Canter. 1971. Developmental sentence scoring: A clinical procedure for estimating syntactic development in childrenÕs spontaneous speech. *Journal of Speech and Hearing Disorders*, 36(3):315–340.

Steven H. Long, Marc E. Fey, and Ron W. Channell. 2004. Computerized profiling (version 9.6.0).

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*. Lawrence Erlbaum Associates, 3rd edition edition.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, June.

Susanne Miyata, Kenji Sagae, and Brian MacWhinney. 2013. The syntax parser GRASP for CHILDES (in Japanese). *Journal of Health and Medical Science*, 3:45–62.

Joakim Nivre, Johan Hall, Sandra Kubler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June. Association for Computational Linguistics.

Christophe Parisse and Marie-Thrse Le Normand. 1987. Indicators of linguistic competence in the peer group conversational behavior of mildly retarded adults. *Applied Psycholinguistics*, 8:19–32.

Mabel L. Rice, Filip Smolik, Denise Perpich, Travis Thompson, Nathan Rytting, and Megan Blossom. 2010. Mean length of utterance levels in 6-month intervals for children 3 to 9 years with and without language impairments. *Journal of Speech, Language, and Hearing Research*, 53:1–17.

Brian Roark, Margaret Mitchell, and Kristy Hollingshead. 2007. Syntactic complexity measures for detecting mild cognitive impairment. In *Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing*, BioNLP '07, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kenji Sagae, Alon Lavie, and Brian MacWhinney. 2005. Automatic measurement of syntactic development in child language. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 197–204, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kenji Sagae, Eric Davis, Alon Lavie, Brian MacWhinney, and Shuly Wintner. 2010. Morphosyntactic annotation of CHILDES transcripts. *Journal of Child Language*, 37:705–729.

Sam Sahakian and Benjamin Snyder. 2012. Automatically learning measures of child language development. In *ACL (2)*, pages 95–99. The Association for Computer Linguistics.

Hollis S. Scarborough. 1990. Index of productive syntax. *Applied Psycholinguistics*, 11:1–22, 3.

Victor H. Yngve. 1960. A model and an hypothesis for language structure. *Proceedings of the American Philosophical Society*, 104:444–466.

Employing Event Inference to Improve Semi-Supervised Chinese Event Extraction

Peifeng Li, Qiaoming Zhu, Guodong Zhou
School of Computer Science & Technology
Soochow University, Suzhou, 215006, China
{pfli, qmzhu, gdzhou}@suda.edu.cn

Abstract

Although semi-supervised model can extract the event mentions matching frequent event patterns, it suffers much from those event mentions, which match infrequent patterns or have no matching pattern. To solve this issue, this paper introduces various kinds of linguistic knowledge-driven event inference mechanisms to semi-supervised Chinese event extraction. These event inference mechanisms can capture linguistic knowledge from four aspects, i.e. semantics of argument role, compositional semantics of trigger, consistency on coreference events and relevant events, to further recover missing event mentions from unlabeled texts. Evaluation on the ACE 2005 Chinese corpus shows that our event inference mechanisms significantly outperform the refined state-of-the-art semi-supervised Chinese event extraction system in F1-score by 8.5%.

1 Introduction

An event is a specific occurrence involving arguments (participants and attributes) of the specific roles. In an event, trigger is the main word which most clearly expresses its occurrence, so recognizing an event can be recast as identifying a corresponding trigger. An event may have several arguments, which are entity mentions (e.g., person name, time, location, etc.) and must fulfill the corresponding roles. Take the following sentence as an example:

S1: On the 25th Dec. (A1: *Artifact*), peacekeepers (A2: *Artifact*) **returned** (E1: *Transport*) to Amman (A3: *Place*) by flight (A4: *Vehicle*).

For this example, an event extraction system should identify one event mention E1, which is triggered by verb "returned" whose event type is *Transport*, with four arguments, "peacekeepers", "25th Dec.", "flight", and "Amman", fulfilling the roles of *Artifact*, *Time*, *Vehicle*, and *Place*, respectively.

Automatically extracting events from free texts is a higher-level Information Extraction (IE) task, which is still a challenge due to the complexity of natural language and the domain-specific nature, especially in Chinese for its specific characteristics. In particular, most of previous studies have focused on English event extraction, while only a few concern Chinese.

Currently, supervised learning models have dominated event extraction. To reduce the labeled data required, a few semi-supervised models have been applied to English event extraction (e.g., Riloff 1996; Yangarber et al., 2000; Stevenson and Greenwood, 2005; Huang and Riloff, 2012). Since classifier-based model needs dozens of annotated documents to train model, most of previous semi-supervised models focused on pattern-based approach, which only needed a few seed (event) patterns. In those pattern-based approaches, frequent event patterns, which occur in many documents, were chosen as relevant patterns to match event mentions in unlabeled texts. However, the order of words in a Chinese sentence is rather agile for its open and flexible structure, and different orders might express the same meaning due to the semantics-driven nature of the Chinese language. This results in the diversity of Chinese event patterns and numerous infrequent patterns, even some event mentions having no matching patterns. Hence, it is an issue to extract the event mentions with infrequent patterns.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2161–2171, Dublin, Ireland, August 23-29 2014.

In this paper, we first implement a pattern-based semi-supervised model for Chinese event extraction as a baseline, following the state-of-the-art system as described in (Liao and Grishman, 2010a) and then refine this model to suit Chinese event extraction. Moreover, we propose various kinds of novel linguistic knowledge-driven event inference mechanisms to address the above issue and recover missing event mentions. These event inference mechanisms can capture the linguistic knowledge from semantics of argument role, compositional semantics of trigger, consistency on coreference events and relevant events. Evaluation on the ACE 2005 Chinese corpus shows that our event inference mechanisms dramatically outperform the baseline.

The rest of this paper is organized as follows. Section 2 overviews related work. Section 3 presents the refined semi-supervised model for Chinese event extraction. Section 4 proposes several linguistic knowledge-driven event inference mechanisms. Section 5 reports and analyzes the experimental results. Finally, we conclude our work in Section 6.

2 Related Work

Almost all previous semi-supervised models focus on English event extraction, which can be subdivided into pattern-based models (e.g., Riloff, 1996; Yangrber et al., 2000; Liao and Grishman, 2010a; Chambers and Jurafsky, 2011; Balasubramanian et al., 2013) and classifier-based models (e.g., Chieu et al., 2003; Maslennikov and Chua, 2007; Patwardhan and Riloff, 2009; Liu and Strzalkowski, 2012; Wang et al., 2013). Classifier-based models normally require a small set of annotated data (e.g., 100 annotated documents), while pattern-based models need dozens of high quality seed patterns.

Riloff (1996) first divided unlabeled documents into irrelevant and relevant documents, and the latter was much likely to contain further relevant patterns. Then event patterns from relevant documents were generated by using an annotated data and a set of heuristic rules. Yangarber et al. (2000) proposed a document-centric view to boost a semi-supervised event extraction system, which assumes relevant documents always contain some shared patterns. Yangarber (2003) further introduced multiple learners into the bootstrapping procedure to make the final decision on the combination of multiple learners on distinct event types. Huang and Riloff (2012) employed role-identifying nouns, which proposed by Phillips and Riloff (2007), as seed terms to extract patterns from relevant documents and then generated the labeled instances to train three classifiers in their event extraction system.

As an alternative, Stevenson and Greenwood (2005) proposed a pattern similarity-centric view and selected relevant patterns on similarity scores. Normally, bootstrapping on the document-centric view tends to accept the irrelevant patterns with a high occurrence frequency in relevant documents. To address this problem, Liao and Grishman (2010a) introduced a pattern similarity metric into the document-centric view as a filter to eliminate those irrelevant patterns. Liao and Grishman (2011) further applied an information retrieval mechanism to detect relevant documents and proposed a self-training strategy for bootstrapping.

In addition, several studies focused on the event pattern representation, such as pairwise (e.g., Subject-Verb, Verb-Object) (Chambers and Jurafsky, 2008, 2009), SVO (Subject-Verb-Object) (Yangarber, 2000; Balasubramanian et al., 2013), chain (Sudo et al., 2001), subtree (Sudo et al., 2003) and complex pattern (Liu and Strzalkowski, 2012).

In the literature, only one paper concerns semi-supervised Chinese event extraction. Chen and Ji (2009a) applied various kinds of cross-lingual features in the bootstrapping procedure to extract Chinese event. With the help of over 500 annotated seed event mentions in 100 documents, they only achieved 35% in F1-score. This indicates the critical challenge in semi-supervised Chinese event extraction.

Only a few studies concern event inference mechanisms. Ji and Grishman (2008) employed a rule-based approach to propagate consistent triggers and arguments across topic-related documents. Liao and Grishman (2010b) employed cross-event consistent information to improve sentence-level event extraction. Hong et al. (2011) regarded entity type consistency as a key feature to predict event mentions and adopted an information retrieval mechanism to promote event extraction. Li et al. (2013) proposed a global argument inference model on Chinese argument extraction to explore specific relationships among relevant event mentions to recover those inter-sentence arguments in the sentence, discourse and document layers. Li et al. (2014) also introduced Markov Logic Network (MLN) to capture the discourse-level consistency between Chinese trigger mentions to further recover those poor-

context event mentions. In a word, all of above mechanisms focus on supervised event extraction and no literature involves in the event inference of semi-supervised event extraction.

3 Semi-supervised Model for Chinese Event Extraction

In this section, we refine a semi-supervised model for Chinese event extraction as a baseline, which includes two views, the document-centric view and pattern similarity-centric view.

3.1 Semi-supervised Model

Liao and Grishman (2010a) proposed a state-of-the-art semi-supervised event extraction system, which was a pattern-based approach and adopted bootstrapping mechanism to extract relevant patterns. Besides, two distinct views, the document-centric view and the pattern similarity-centric view as described in Subsection 3.2 and 3.3, are incorporated in the bootstrapping procedure to rank event patterns on different metrics. In each iteration, the candidate patterns, which extracted from unlabeled texts as the candidates of relevant patterns, are ranked following the document-centric view, then the candidate patterns with pattern similarity scores below a similarity threshold (0.9 in (Liao and Grishman, 2010a)) will be removed; only top 3 candidate patterns in the ranking scores of the document-centric view will be accepted as relevant patterns. In addition, if no pattern is found in the current iteration, the threshold will be reduced by 0.1 until new relevant patterns are extracted.

As we mentioned earlier, the open and flexible structure of Chinese sentences results in the diversity of Chinese event patterns. Moreover, the syntax or semantic path is often used to represent event patterns, but the performance in Chinese syntactic parsers and Semantic Role Labeling (SRL) tools is lower than that in English. Therefore, we refine this semi-supervised model to suit Chinese event extraction in three aspects as follows, due to the above characteristics of Chinese language.

Firstly, we construct a refined event pattern representation of Chinese events. Liao and Grishman (2010a) used semantic roles to represent the relationship between the trigger and its arguments. Due to the wide spread of ellipsis (especially entities) and the relatively low performance of Chinese SRL, pairwise (trigger-entity) representation and dependency path are introduced to represent Chinese event pattern in our refined model. Hence, the event pattern in this paper is a triple-style template as follows.

<trigger, entity type, their dependency path >

A pattern is formed by a trigger, the entity type of its argument[1] and the dependency path from the trigger to the argument. For example, trigger "returned" and its argument "peacekeepers" (entity type: PER) in sentence S1 can be described as a pattern *<returned, PER, nsubj>*.

Secondly, we introduce a novel mechanism to extract candidate patterns. Since verb and noun dominate in triggering an event in Chinese and they are chosen as candidate triggers to create candidate patterns. Besides, since different event types may have different roles and different roles are fulfilled by entities with different types, the entities whose types can fulfil the core roles of a specific event are chosen as candidate entities. For example, *Attacker* and *Target* are the core roles of event *Attack* and entity types *PER/ORG/GPE*[2] can fulfil above two roles, so we only accept those entities, whose types belong to *PER/ORG/GPE*, to form candidate patterns. For each sentence in the unlabeled data, all candidate trigger-entity pairs and their dependency path are enumerated as candidate patterns.

Finally, we present a new mechanism to generate seed patterns based on seed triggers. Considering the relatively large number of Chinese triggers and the flexibility of Chinese sentences, an instance-based approach is adopted by enumerating a few high-quality seed triggers with explicit meaning and high probability to trigger a specific event. Instead of dozens of predefined patterns required in previous studies, only one seed trigger is given to each event type or subtype without any predefined patterns. Hence, all patterns consisting of a seed trigger in the candidate patterns are accepted as seed patterns for their high probability to trigger a specific event.

3.2 Document-centric View

The document-centric view regards those documents containing the patterns always identified as relevant to a specific event as relevant documents and concludes that they are likely to contain additional

[1] All event arguments must be entity mentions following the ACE 2005 annotation guidelines of events.

[2] PER/ORG/GPE refers to person, organization and geo-political entity respectively, which are annotated in the ACE 2005 corpus. These helpful information can be seen as ontological classes.

relevant patterns. Hence, those candidate patterns occurring in the relevant documents frequently will be extracted as relevant ones. Following Yangarber et al. (2000) and Liao and Grishman (2010a), we also employ the disjunctive voting scheme to calculate the ranking scores $R_{score}(p)$ of pattern p as follows.

$$R_{Score}(p) = \frac{\sum_{d \in L(p)} Rel(d)}{|L(p)|} * log \sum_{d \in L(p)} Rel(d) \quad (1)$$

where $L(p)$ is the set of documents, which contain candidate pattern p, and $Rel(d)$ is the relevance score of document d as follows.

$$Rel(d) = 1 - \prod_{p \in P} (1 - \frac{\sum_{d \in L(p)} Rel'(d)}{|L(p)|}) \quad (2)$$

where $Rel'(d)$ is the relevance score of document d in the previous iteration. Initially the relevance score of document d is set to n if document d has n relevant patterns in the set of extracted patterns P.

3.3 Pattern Similarity-centric View

The similarity-centric view tries to find the candidate patterns who are similar to those seed patterns. The similarity scores derive from two aspects, lexical similarity and syntactic similarity, while the former is based on the trigger and entity type in a pattern and the latter is based on the relation between the trigger and the entity. Especially, we realize the pattern similarity view following the lexical and syntactic similarity, and refine the similarity ranking score $I_{score}(p)$ of candidate pattern p as follows:

$$I_{score}(p) = \max_{s \in P}(WSim(t_p, t_s) \times ESim(e_p, e_s) \times DSim(d_p, d_s)) \quad (3)$$

where t, e and d represent the trigger, entity type and dependency path in candidate pattern $p(t_p, e_p, d_p)$ or seed pattern $s(t_s, e_s, d_s)$ in the set of extracted patterns P, respectively; *ESim* identifies whether two entities have the same type, and assigned 1 if two entities have the same entity type and otherwise a small number 0.1; *DSim* calculates the similarity between two dependency paths in edit distance. Finally, *WSim* is to obtain the trigger similarity in lexical semantics, using Hownet (Dong and Dong, 2006) following Liu and Li (2002):

$$WSim(t_p, t_s) = \frac{\phi}{Dis(t_p, t_s) + \phi} \quad (4)$$

where $Dis(t_p,t_s)$ is the distance between the sememes of triggers t_p and t_s, in HowNet's sememe hierarchical architecture, with parameter ϕ assigned 0.75 following Liu and Li (2002).

4 Event Inference

The pattern-based semi-supervised model cannot extract those event mentions matching infrequent patterns or without matching patterns. The knowledge from linguistic aspect (e.g., definition of events, compositional semantics of Chinese words, coreference events and relevant events, etc.) is helpful to further recover missing event mentions or filter pseudo event mentions. In this section, various kinds of event inference mechanisms based on linguistic knowledge are proposed to improve the performance of semi-supervised Chinese event extraction.

We unify the semi-supervised model and the event inference mechanisms into one model as follows: In each iteration, after the top 3 patterns have been chosen following the document-centric view and event mentions in the unlabeled data have been extracted by pattern matching, all event inference

mechanisms are applied to recover missing event mentions,. Due to our inference mechanisms are trigger-based and each inferred event mention may have more than one pattern while most of them are noisy, we do not add those patterns in the set of relevant patterns for bootstrapping.

4.1 Event Inference on Role Semantics

The core of an event can be expressed as "*Who do What to Whom*" in which "Who" and "Whom" are the core roles[3] to participate in an event, while "What" often refers to event trigger. The relationship between the verbal trigger and its core roles are the key clues to express event semantics. Since the subject or object always play the core roles in an event mention, SVO (Sbject-Verb-Object) is a better representation of event pattern. However, ellipsis is a widespread phenomenon in Chinese language and many sentences do not have an overt subject or object, so lots of event mentions cannot be represented as SVO pattern. In this paper, we only use the trigger-entity pair to represent event pattern and one of the disadvantages of this representation is its loose constraint on events, which will extract lots of pseudo event mentions.

In most cases in Chinese, the object is often the most important core role to identify a specific event and it is more helpful than the subject to distinguish true event mentions from pseudo ones. Take following two sentences as examples:

S2: 老师(PER) **打(hit)**了 这个学生(PER)。(The teacher **hit** this student.)

S3: 老师(PER) **打(call)**了 电话 给 这个学生(PER)。(The teacher made a phone **call** to this student.)

The relation between verb 打 (hit) and object 这个学生 (this student) is clear to indicate sentence S2 is an *Attack* event mention since the object is a person, while object 电话 (phone) in sentence S3 is not a person and it indicates this sentence is not an *Attack* event mention following the sense of verb 打 (call). Therefore, the object is an effective evidence to indicate event mentions and it is incorporated in our model to remove pseudo event mentions as follows.

Role Semantics: If the object of a candidate verbal trigger mention is not an entity or its entity type cannot fulfil the object roles (e.g., *Victim* in events *Injure* and *Die*) in a specific event, this candidate trigger mention[4] will be inferred as pseudo one.

For example, core role *Target* of event *Attack* often acts as the object of a verbal trigger and entity types PER, ORG and GPE can fulfill this role according to be definition of event *Attack* in the ACE 2005 corpus. Hence, a candidate trigger mention of event *Attack* will be regarded as pseudo one when this mention has an object which is not an entity or whose entity type is not PER, ORG or GPE.

4.2 Event Inference on Compositional Semantics

In Chinese language, a word is composed of one or more characters. Almost all Chinese characters have their own meanings and are morpheme (or single-morpheme word), the minimal meaningful unit. If a Chinese word contains more than one character, its meaning can often be derived from its composite morphemes. This more fine-grained semantics is compositional semantics of Chinese words. Actually, it is also a normal way for a native Chinese speaker to understand a new Chinese word.

Two-morpheme words are used widely in Chinese language and almost all Chinese triggers contain one or two morphemes. The compositional semantics of a two-morpheme word comes from both its morphemes and morphological structure. Besides morphological structure *Coordination*, all other morphological structures (e.g., *Modifier-Head, Predicate-Object, Predicate-Complement* (Li and zhou, 2012)) always have one head morpheme, the morpheme as the governing semantic element, to express the meaning of a word. Commonly, there are two head morphemes in a two-morpheme word of *Coordination* structure. In particular, a two-morpheme word triggers an event if its two head morphemes are homogeneous (e.g., 攻(attack)击(attack), 死(die)亡(die)). Otherwise, it may refer to more than one event and this means that two triggers are within a word whose morphological structure is *Coordination*. Take the following sentence as an example:

[3] We select core roles following the ACE Chinese annotation guidelines of events. *Agent/Victim* are the core roles of events *Die/Injure* while *Attacker/Target* are the core roles of event *Attack*.

[4] Recognizing a trigger mention can be recast as identifying a corresponding event mention, since trigger is the main word which most clearly expresses the occurrence of an event.

S4: 一名少年**刺**(E2: *Attack*)**死**(E3: *Die*)一名妇女。(A younger **stabbed** (E2: *Attack*) a woman to **death** (E3: *Die*).)

In S4, two-morpheme word 刺死 (stab a person to death) is a trigger with the *Coordination* structure. There are two event mentions in sentence S4, one *Attack* (E2) and one *Die* (E3), while morpheme 刺 (stab) triggers an *Attack* event and 死 (die) refers to a *Die* one.

Almost all event extraction systems assigned only one event type to a trigger and this will lead to that the other event type does not have any patterns to match and then cannot be identified. To address this issue, we first identify those triggers who refers to two distinct events as follows: for each two-morpheme candidate trigger in the candidate patterns whose morphemes are m_1 and m_2, it will be identified as candidate trigger with two event types and split into two single-morpheme word to generate two candidate trigger mentions when the following three conditions are satisfied:

1) $verb \in POS(m_1) \wedge verb \in POS(m_2)$
2) $\underset{s_1 \in seeds}{Max}(Wsim(m_1, s_1)) = 1 \wedge \underset{s_2 \in seeds}{Max}(Wsim(m_2, s_2)) = 1 \wedge Etype(s_1) \neq Etype(s_2)$
3) $Morph(m_1\ m_2) = Coordination$

where *POS(m)* returns all possible parts of speech of morpheme *m* in Hownet and *Etype(s)* is to obtain the event type of seed trigger *s*; *WSim(m,s)* is defined in Subsection 3.3 and returns 1 when one word *m* is the synonym of the other word *s*; *Morph(w)* is to obtain the morphological structure of word *w* following Li and Zhou (2012).

Since there is a strong trigger consistency in those two-morpheme words of *Coordination* structure which refers to two distinct events, we propose an event inference mechanism as follows.

Compositional semantics: For each two-morpheme word identified by the above three conditions, if one of its morphemes has been extracted as an trigger mention of a specific event type, the other morpheme in the same word will refer to an a relevant event type.

4.3 Event Inference on Coreference Events

To mine more event mentions, we use the simple trigger-entity pair to represent event pattern in this paper. However, lots of event mentions still cannot be extracted due to the ellipsis of arguments. Take following sentences as examples:

S5: 美国与北韩在吉隆坡结束**会谈**(E4: *Meeting*)。(The US and DPRK finished **talking** (E4: *Meeting*) in Kuala Lumpur.)

S6: **会谈**(E5: *Meeting*)的气氛严肃。(The **talks** (E5: *Meeting*) are serious.)

Obviously, more than one pattern of event mention E4 can be generated from sentence S5, since it contains more than one entity. On the contrary, no pattern can be extracted from S6 and this leads to event mention E5 cannot be extracted in our pattern-based semi-supervised model.

Within a document, almost all event mentions are around a topic and there is a strong trigger consistency: if one mention of a word triggers a specific event, its other mentions in the same document will refer to the same event type. Besides, similar words (e.g., 炸 (bomb), 爆炸 (bomb), 轰炸 (bomb)), which contains the same head morpheme, always express the same or similar meaning following the principle of compositional semantics. Similarly, there is a strong trigger consistency on those similar words: If one mention of a word refers to a specific event, the mentions of its similar words in the same document will trigger events of the same type.

Since the mentions of the same word or similar words are often coreference ones and always refer to the same event type, we propose an event inference mechanism on coreference events to recover missing event mentions based on head morpheme as follows. In particular, head morphemes are also identified following Li and Zhou (2012).

Coreference events: 1) if a mention of a candidate trigger refers to a specific event, all its other mentions in the same document will trigger the same type event; 2) if one mention of a candidate trigger refers to a specific event, all the mentions of its similar words in the same document will trigger the same type event too.

4.4 Event Inference on Relevant Events

The bootstrapping procedure of the document-centric view selects frequent patterns in relevant docu-

ments and ignores those infrequent patterns both in relevant or irrelevant documents. However, the number of infrequent patterns in Chinese is larger than that in English, due to its open and flexible sentence structure, as mentioned in Subsection 3.1.

Besides the pattern-based semi-supervised model, we propose a trigger-based mechanism as a supplement to recover those missing event mentions concerning infrequent patterns following this assumption: if a trigger mention refers a specific event in a document, there is a high probability that its relevant events occur in the same document. Take the following sentence as an example:

S7: 在**冲突**(E6: *Attack*)中，有 *1 名阿拉伯人***死亡**(E7: *Die*)。(An Arabian was **dying** (E7: *Die*) in this **conflict** (E6: *Attack*).)

In sentence S7, there is an extracted *Die* event mention E7 triggered by 死亡 (die) and 冲突 (conflict) is a candidate trigger mention. If there is an evidence that 冲突 (conflict) triggers an *Attack* event in the other documents, it is possible to identify 冲突 (conflict) as a trigger mention of *Attack* event in S7 for the high probability that events *Die* and *Attack* occur in the same document. We propose an inference mechanism on relevant events as follows.

Relevant Events: If a trigger mention is identified in a document, each candidate trigger mention in the same document will be recognized as true ones when it satisfies the following condition: this candidate trigger occurs in the other documents as an event trigger and refers to the relevant events of this identified trigger mentions.

Since the seed triggers have a high probability to trigger a specific event, to further explore those missing event mentions, we expand this inference mechanism following compositional semantics in Chinese and expand the condition as follows: This candidate trigger occurs in the other documents as an event trigger or contains one of the seed triggers, which refers to the relevant events of this identified trigger mentions.

5 Experimentation

In this section, we systematically evaluate our event inference mechanisms on the ACE 2005 Chinese corpus and provide the analysis.

5.1 Experimental Setting

The ACE 2005 Chinese corpus is the only available corpus in Chinese event extraction and it is used in all our experiments. This corpus contains 633 documents annotated with 33 predefined types. Due to evaluation on all 33 types is a hard work for the time-consuming bootstrapping procedure and the diversity of distinct event types, most of previous works selected part of event types for evaluation. In this paper, 3 event types (i.e. *Die*, *Injure* and *Attack*) are selected for evaluation, because they reflect the relevance of different event types and occur at different frequencies in the corpus. While events *Die* and *Injure* are easy to define, event *Attack* is rather complicated and can be divided into several subtypes. In the ACE 2005 Chinese corpus, almost one third of the annotated event mentions belong to the above three event types. Moreover, we report the experimental results on all 33 event types to further verify the effectiveness of our inference mechanisms in Subsection 5.2.

Unlike MUC shared task, which only distinguishes whether a sentence contains a specific event mention or not, we follow previous studies on the ACE 2005 corpus and report the performance of trigger-based event extraction: a trigger is correctly identified if its position and event type match a reference trigger. As for evaluation, we use the ground truth entities, time and values annotated in the ACE 2005 Chinese corpus, and report the micro-average Precision (P), Recall (R) and F1-score (F1).

Table 1 shows the seed triggers for the three event types. For example, only one seed trigger is provided for either the *Die* or *Injure* event, while three seed triggers are given for event *Attack.* Since the *Attack* event contains several distinct event subtypes, we assign one seed trigger to each of its major subtypes. Thus, all patterns whose triggers belong to the set of seed triggers are accepted as seed patterns automatically.

Type	Die	Injure	Attack
Seed triggers	死(die)	伤(injure)	攻击(attack), 冲突(conflict), 打(hit)

Table 1. Seed triggers of *Die*, *Injure* and *Attack* event types

Besides, all the sentences in the corpus are divided into words using a Chinese word segmentation tool (*ICTCLAS*) with all entities annotated in the corpus kept. We use *Berkeley Parser* and *Stanford Parser* to create the constituent and dependency parse trees.

5.2 Experimental Results

To verify the performance of our event inference mechanisms, it is compared with the refined baseline, a supervised model for Chinese event extraction. Table 2 shows the results of our event inference mechanisms with peak recall, precision and F1-score, following Liao and Grishman (2010a). Compared with the baseline, Table 2 shows that our event inference mechanisms improve the F1-score of Chinese event extraction by 8.5%, largely due to the improvement of 11.8% in recall. These results confirm the effectiveness of our event inference mechanisms in recovering missing event mentions. The disadvantage of our event inference mechanisms is the fact that it will also introduce some pseudo event mentions into our model and harm the precision. Additionally, there is still a big performance gap between our model and the supervised model and this leaves much room for future research.

Approach	Attack			Injure			Die			All (micro-average)		
	P(%)	R(%)	F1	P(%)	R(%)	F1	P(%)	R(%)	F1	P(%)	R(%)	F1
Baseline	**71.4**	36.6	48.4	**93.2**	41.7	57.6	**90.1**	44.0	59.3	**79.7**	39.4	52.7
+Event inference	70.9	**47.5**	**56.9**	83.2	**54.6**	**65.9**	80.8	**57.2**	**67.0**	75.5	**51.2**	**61.2**
Supervised model	*70.4*	*72.5*	*71.4*	*85.3*	*78.4*	*81.7*	*83.9*	*92.9*	*88.1*	*77.2*	*78.4*	*77.8*

Table 2. Performance of event inference mechanisms in Chinese event extraction *(Attack/Injure/Die*).

Table 2 also indicates the performance difference of our inference mechanisms for distinct event types. Among all event types, event *Attack* achieves the highest improvement (8.5%) in F1-score, with a dramatic improvement of 10.9% in recall and a less loss of 0.5% in precision. Event *Die* and *Injure* also gain a significant improvement of 7.7% and 8.3% in F1-score respectively, largely due to the increase in recall, while their precisions reduce rapidly due to those pseudo event mentions inferred by our inference mechanisms. However, the loss of precision of event *Attack* is much less than these of events *Die* and *Injure*. The reason is that the inference on role semantics mainly impacts on *Attack* events to remove pseudo event mentions.

To well evaluate different approaches, it is better to compare them on different corpora. Since the ACE 2005 Chinese corpus is the only available corpus in Chinese event extraction, we divide it into three sub-corpora according to data sources, i.e. Broadcast News, Newswire and WebLog, which are much different in various aspects, such as quality, length and style. Figure 1 compares the performance of different models on different sub-corpora. It indicates that our event inference mechanisms perfect better than the baseline in all three sub-corpora and that results confirm the huge influence of the event inference mechanisms. It also shows that the WebLog sub-corpus reports the worst F1-score due to the low document quality and the low percentage of relevant documents, and that the Newswire sub-corpus reports significantly better performance than the Broadcast News sub-corpus due to its spoken nature.

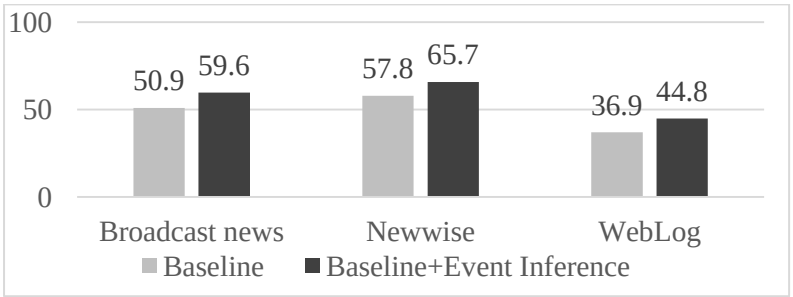


Figure 1. Performance comparison (F1-score) on different data sources.

To further verify the effectiveness of our event inference mechanisms, we evaluate them on all 33 event types. Due to event extraction is a domain-specific task, distinct event types have the different seed triggers and different pro-process procedures. In this paper, we just report the final results for the

sake of brevity. Table 3 shows the experimental results on all 33 event types and it ensures that our mechanisms are effective on extracting all event types. Compared with the baseline, our approach improves the F1-score by 7.6%, which is less than that reported in Table 2. Among all 33 event types, the performances of almost all event types associated with justice are higher than other event types for their unambiguous definitions and high coverage of seed triggers while event *Transport* achieves the lowest performance for its complexity and low coverage of seed triggers. Besides, the performance on all event types is lower than that on 3 event types and this result comes from the low performance of the *Transport* event which occupies almost 20% of all annotated event mentions in the ACE 2005 Chinese corpus.

Approach	P(%)	R(%)	F1
Baseline	**70.7**	34.2	46.1
+Event inference	65.2	**45.7**	**53.7**

Table 3. Performance of event inference mechanisms in Chinese event extraction (All 33 event types).

5.3 Analysis on Event Inference Mechanisms

Table 4 shows the contributions of the different event inference mechanisms. It is worthy to mention that an event mentions may be identified by both the semi-supervised model and the event inference mechanisms. In this paper, we attribute those extracted event mentions to the former and the contribution of our inference mechanisms is greater than those in Table 4.

Inference	P(%)	R(%)	F1
Baseline	79.7	39.4	52.7
+Inference on role semantics (**RS**)	87.5(+7.8)	39.1(-0.3)	54.1(+1.4)
+Inference on compositional semantic (**CS**)	85.7(+6.0)	43.7(+4.3)	57.8(+3.7)
+Inference on coreference events (**CE**)	83.0(+3.3)	45.8(+6.4)	59.0(+1.2)
+Inference on relevant events (**RE**)	75.7(-4.0)	51.3(+11.9)	61.2(+2.2)

Table 4. The contribution of event inference on Chinese event extraction.

Actually, inference mechanism **RS** is a filter to remove those pseudo event mentions and it can improve the precision (+7.8%), with a less lost (-0.3%) in recall. Moreover, it can also help the seed pattern generation to generate high quality seed patterns. Table 5 shows the contribution of **RS** on seed pattern generation and we report the result of Chinese event extraction which only uses the seed patterns[5]. It improves the accuracy from 75.8% to 82.5%, largely due to the decline (-30) in the set of pseudo event mentions. These results indicate that the object is a key clue to identify event mentions.

Method	#True event mentions	#Pseudo event mentions
w/o **RS**	273	87
w/ **RS**	269	57

Table 5. The contribution of **RS** on seed pattern generation.

Chen and Ji (2009b) have reported that almost 13% of Chinese triggers are in-word or cross-words and this figure ensures it is an important issue. Inference mechanism **CS** gains the highest improvement (+3.7%) in F1-score and this result indicates that compositional semantics is an effective way to solve such issue. The accuracy of this inference mechanism is very high (~92%) and most of the exceptions need the help of deep semantics since these instances are also hard to be distinguished by humans without the context.

Inference mechanisms **CE** and **RE** improve the F1-scores by 1.2% and 2.2% respectively. **CE** assumes all mentions of a word in a document only have one sense and it will introduce lots of pseudo event mentions to reduce precision. The experimental results also show that **RE** is an effective supplement of the document-centric view to mine event mentions. Although they derive from the similar

[5] Since sometimes a pattern can infer both true event mentions and pseudo event mentions, it is hard to identify whether a pattern is relevant or irrelevant without the test data. Hence, we compare their extracted event mentions in this paper.

principle of occurrence of relevant events, they focus on different perspectives where **RE** is trigger-based and the document-centric view is pattern-based. **RE** ignores the difference on patterns and identifies event mentions on the occurrence of their relevant event mentions. In addition, sense shifting of Chinese words in different contexts is the main factor to extract lots of pseudo event mentions and then reduce the precision rapidly.

It's obvious that these inference mechanisms interact with others. In particular, almost 20% event mentions can be inferred by both **CE** and **RE** for the transitivity of event inference on coreference and relevant events. Besides, **RS** is not only beneficial to the semi-supervised model, but also helpful to the other inference mechanisms to further remove pseudo event mentions.

6 Conclusion

This paper proposes various kinds of novel linguistic knowledge-driven event inference mechanisms as a supplement of the semi-supervised Chinese event extraction to recover missing event mentions. The experimental results verify their effectiveness to extract the event mentions with infrequent patterns or without matching pattern. Although this paper focuses on Chinese language, most of the event inference mechanisms are language-independent and can be applied to other languages. Our future work will focus on how to apply our event inference mechanisms to other languages and introduce more effective inference mechanisms to further improve the performance of semi-supervised event extraction.

Acknowledgments

The authors would like to thank three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China under Grant No. 61331011 and No. 61272260, the National 863 Project of China under Grant No. 2012AA011102.

Reference

Niranjan Balasubramanian, Stephen Soderland, Mausam and Oren Etzioni. 2013. *Generating Coherent Event Schemas at Scale*. In Proc. EMNLP 2013, pages 1721-1731, Seatle, WA.

Nathanael Chambers and Dan Jurafsky. 2008. *Unsupervised Learning of Narrative Event Chains*. In Proc. ACL-HLT 2008, pages 787-797, Hawaii.

Nathanael Chambers and Dan Jurafsky. 2009. *Unsupervised Learning of Narrative Schemas and Their Participants*. In Proc. ACL 2009, pages 602-610, Columbus, OH.

Nathanael Chambers and Dan Jurafsky. 2011. *Template-Based Information Extraction without the Templates*. In Proc. ACL 2011, pages 976-986, Portland, OR.

Hai Leong Chieu, Hwee Tou Ng and Yoong Keok Lee. 2003. *Closing the Gap: Learning-based Information Extraction Rivaling Knowledge-Engineering Methods*. In Proc. ACL 2003, pages 216-230, Sapporo, Japan.

Zheng Chen and Heng Ji. 2009a. *Can One Language Bootstrap the Other: A Case Study on Event Extraction*. In Proc. NAACL-HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing, pages 66-74, Boulder, CO.

Zheng Chen and Heng Ji. 2009b. *Language Specific Issue and Feature Exploration in Chinese Event Extraction*. In Proc. NAACL-HLT 2009, pages 209-212, Boulder, CO.

Zhengdong Dong and Qiang Dong. 2006. *HowNet and the Computation of Meaning*. World Scientific Pub Co. Inc.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou and Qiaoming Zhu. 2011. *Using Cross-Entity Inference to Improve Event Extraction*. In Proc. ACL 2011, pages 1127-1136, Portland, OR.

Ruihong Huang and Ellen Riloff. 2012. *Bootstrpped Training of Event Extraction Classifiers*. In Proc. EACL 2012, pages 286-295, Avignon, France.

Heng Ji and Ralph Grishman. 2008. *Refining Event Extraction through Cross-Document Inference*. In Proc. ACL-HLT 2008, pages 254-262, Columbus, OH.

Peifeng Li and Guodong Zhou. 2012. *Employing Morphological Structures and Sememes for Chinese Event Extraction*. In Proc. COLING 2012, pages 1619-1634, Mumbai, India.

Peifeng Li, Qiaoming Zhu, and Guodong Zhou. 2013. *Argument Inference from Relevant Event Mentions in Chinese Argument Extraction*. In Proc. ACL 2013, pages 1477-1487, Sofia, Bugaria.

Peifeng Li, Qiaoming Zhu, Guodong Zhou. 2014. *Using Compositional Semantics and Discourse Consistency to Improve Chinese Trigger Identification*. Information Processing and Management, 50: 399–415.

Shasha Liao and Ralph Grishman. 2010a. *Filtered Ranking for Bootstrapping in Event Extraction*. In Proc. COLING 2010, pages 680-688, Beijing, China.

Shasha Liao and Ralph Grishman. 2010b. *Using Document Level Cross-Event Inference to Improve Event Extraction*. In Proc. ACL 2010, pages 789-797, Uppsala, Sweden.

Shasha Liao and Ralph Grishman. 2011. *Can Document Selection Help Semi-supervised Learning? A Case Study On Event Extraction*. In Proc. ACL 2011, pages 260-265, Portland, OR.

Qun Liu and Sujian Li. 2002. *Word Similarity Computing Based on How-net*. In Proc. 3th Chinese Lexical Semantic Workshop, Taibei, Taiwan.

Ting Liu and Tomek Strzalkowski. 2012. *Bootstrapping Events and Relations from Text*. In Proc. EACL 2012, pages 296-305, Avignon, France.

Mstislav Maslennikov and Tat-Seng Chua. 2007. *A Multi-resolution Framework for Information Extraction from Free Text*. In Proc. ACL 2007, pages 592-599, Prague, Czech Republic.

Siddharth Patwardhan and Ellen Riloff. 2009. *A Unified Model of Phrasal and Sentential Evidence for Information Extraction*. In Proc. EMNLP 2009, pages 151-160, Singapore.

William Phillips and Ellen Riloff. 2007. *Exploiting Role-Identifying Nouns and Expressions for Information Extraction*. In Proc. RANLP 2007, pages 468-473, Borovets, Bulgaria.

Ellen Riloff. 1996. *Automatically Generating Extraction Patterns from Untagged Text.* In Proc. AAAI 1996, pages 1044-1049, Portland, OR.

Mark Stevenson and Mark Greenwood. 2005. *A Semantic Approach to IE Pattern Induction.* In Proc. ACL 2005, pages 379-386, Ann Arbor, MI.

Kiyoshi Sudo, Satoshi Sekine, Ralph Grishman. 2001. *Automatic Pattern Acquisition for Japanese Information Extraction*. In Proc. HLT 2001, pages 1-7, San Diego, CA.

Kiyoshi Sudo, Satoshi Sekine, Ralph Grishman. 2003. *An Improved Extraction Pattern Representation Model for Automatic IE Pattern Acquisition*. In Proc. ACL 2003, pages 224-231, Tokyo, Japan.

Roman Yangarber, Ralph Grishman, Pasi Tapanainen and Silja Huttunen. 2000. *Automatic Acquisition of Domain Knowledge for Information Extraction.* In Proc. COLING 2000, pages 940-946, Hong Kong.

Roman Yangarber. 2003. *Counter-Training in Discovery of Semantic Patterns.* In Proc. ACL 2003, pages 343-350, Sapporo, Japan.

Jian Wang, Qian Xu, Hongfei Lin, Zhihao Yang, Yanpeng Li. 2013. *Semi-supervised Method for Biomedical Event Extraction*. Proteome Science, 11(Suppl 1): S17.

Supervised Ranking of Co-occurrence Profiles for Acquisition of Continuous Lexical Attributes

Julian Brooke
Department of Computer Science
University of Toronto
jbrooke@cs.toronto.edu

Graeme Hirst
Department of Computer Science
University of Toronto
gh@cs.toronto.edu

Abstract

Certain common lexical attributes such as polarity and formality are continuous, creating challenges for accurate lexicon creation. Here we present a general method for automatically placing words on these spectra, using *co-occurrence profiles*, counts of co-occurring words within a large corpus, as a feature vector to a supervised ranking algorithm. With regards to both polarity and formality, we show this method consistently outperforms commonly-used alternatives, both with respect to the intrinsic quality of the lexicon and also when these newly-built lexicons are used in downstream tasks.

1 Introduction

Lexicon acquisition represents one key way that the information in large corpora and other resources can be leveraged in various NLP tasks, particularly when the range of lexical items involved in a particular phenomenon is much more diverse than can typically be captured in manually-built resources. Another property of the lexicon which might limit a manual approach is the fact that certain attributes are not discrete, instead falling on a continuous spectrum; although there are manually-built dictionaries which contain fine-grained judgments of spectra—an example is the MRC psychological database (Coltheart, 1980)—these tend to be very low in coverage, reflecting the difficulty in collecting this information.

Within computational linguistics, the continuous lexical attribute that has received the most attention is undoubtedly the positive-negative spectrum, otherwise known as *semantic orientation* (SO) or *polarity*. Much of the work focused on acquisition of this attribute at the lexical level has involved simplification to a binary (positive-negative) or ternary (positive-neutral-negative) distinction (Hatzivassiloglou and McKeown, 1997; Takamura et al., 2005; Kaji and Kitsuregawa, 2007; Rao and Ravichandra, 2009; Mohammad et al., 2009; Volkova et al., 2013) but other work explicitly offers a continuous quantification (Turney, 2002; Turney and Littman, 2003; Baccianella et al., 2010). Another spectrum with a prominent role in the lexicon is *formality* (Brooke et al., 2010; Lahiri et al., 2011), which includes colloquial words at one end, socially-distancing words at the other, and common vocabulary in the middle. In this paper, we will focus on these two spectra; the method presented, however, is intended to be general, and as such could be easily applied to other spectra such as those in the MRC database, e.g. abstractness (Turney et al., 2011), and other kinds of variation captured in, for instance, Osgood's semantic differential (Osgood et al., 1957).

The typical approach to this problem involves semi-supervised methods using vector space and/or graph representations and a set of seed terms. Our method is novel in that it uses fully supervised SVM ranking of co-occurrence profiles, i.e. normalized counts of instances of binary text co-occurrence between the target word and a large set of profiling words, selected on the basis of their frequency, in a publicly-available blog corpus. The seed terms from earlier methods are now viewed as training examples for building a supervised model that can connect the distributions of co-occurring words in this wider vocabulary to relative locations on a continuous spectrum. This approach depends somewhat upon improved manual lexical resources available for these tasks, such as the SO-CAL dictionary (Taboada

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2172–2183, Dublin, Ireland, August 23-29 2014.

et al., 2011) in the case of polarity, but we limit our (word) training set size in order to show it will work in resource-scarce situations, such as languages other than English. Our method is straightforward, practical, and offers essentially full coverage, including words and lexicogrammtical patterns that are simply not accessible by the many popular methods that are primarily based on WordNet.

To evaluate, we compare our method with popular alternatives in both polarity and formality, with particular emphasis on other methods based on corpus co-occurrence that have also been shown to be generalizable across various spectra, i.e. LSA and PMI. For both spectra of interest here, we evaluate both intrinsically using pairwise comparisons from manually-built lexical resources, and also extrinsically in downstream tasks such as text-level polarity classification and sentence-level formality judgments. We show our method is consistently superior across our various evaluations. We also show that not only are co-occurrence profiles a good source of information for supervised ranking, but that a focus on ranking rather than regression in this space appears to be fundamental to the success of a supervised approach to lexical spectra.

2 Related Work

Viewed primarily as a categorical task, the creation or expansion of lexical resources for sentiment analysis is a commonly-addressed problem. In addition to SentiWordNet (Baccianella et al., 2010), which we will compare to directly to here, there are numerous mostly semi-supervised approaches based on exploiting the glosses and/or the graph structure of WordNet to determine whether a word is positive or negative (Kamps et al., 2004; Hu and Liu, 2004; Kim and Hovy, 2004; Takamura et al., 2005; Andreevskaia and Bergler, 2006; Rao and Ravichandra, 2009; Hassan and Radev, 2010), or taking advantage of some other lexicographic resources (Mohammad et al., 2009; Klebanov et al., 2013). The earliest corpus-based approach was that of Hatzivassiloglou and McKeown (1997) who used local syntactic information, i.e. conjunctions, to make connections between adjectives; other work that makes use of local patterns in a corpus includes that of Kaji and Kitsuregawa (2007) and Kanayama and Nasukawa (2006). Turney (2002) built a continuous polarity lexicon using PMI based on Internet hit counts as a useful measure of relatedness between seeds, and Turney and Littman (2003) compared this approach with LSA, which uses general patterns of co-occurrence based on dimensionality reduction. Velikovich et al. (2010) combined web-scale corpora with a graph-based approach, assigning polarity scores to n-grams on the basis of the maximum weighed path from an n-gram to the seed terms, using a small (6-word) context around the word. Like us, Volkova et al. (2013) use social media, iteratively labeling tweets and words for subjectivity and polarity. Fully-supervised approaches to polarity lexicon acquisition are rare, but one example is the work of Chetviorkin and Loukachevitch (2012), who classify words as being sentiment-relevant in Russian using a small set of statistical features, including ratios across disparate corpora.

Our interest in the continuous aspect of polarity overlaps with work on deriving the semantic intensity of lexical items from corpora (Sheinman and Tokunaga, 2009); in this task, small sets of synonyms are ranked according to their intensity, including (but not limited to) polarity. De Melo and Bansal (2013) use a Mixed Integer Linear Programing algorithm to combine information from multiple pairs into a single coherent ranking. As with some of the work in polarity, the focus is on adjectives and local patterns which explicitly distinguish degrees of intensity e.g. *not only x but also y*, which limits its range of application; it would not, for instance, be useful for formality or other more pragmatic variations.

Beyond the our work in LSA-based formality lexicon creation (Brooke et al., 2010) and the sentence-level formality annotation of Lahiri et al. (2011), which we discuss later in more detail, there is a relatively small amount of computational research that directly addresses formality. At lexical level, Li and Yarowsky (2008) identify formal and informal synonyms in Chinese. Heylighen and Dewaele (2002) and Li et al. (2013) both offer text-level quantifications of formality; the former is based on POS frequency, while the latter is based on the Coh-Metrix textual metrics. Using these kinds of metrics, formality has been evaluated in social media (Mosquera and Moreda, 2012). A supervised text classification approach to formality is offered by Sheika and Inkpen (2012). Lexical formality is obviously related to lexicon-based readability (Kidwell et al., 2009) and lexical simplification (Carroll et al., 1999), and is

also relevant to the recent interest in identifying social relationships (Peterson et al., 2011) and shows of politeness (Danescu-Niculescu-Mizil et al., 2013).

3 Method

Our approach to lexicon acquisition falls into the general category of corpus-based techniques. For both attributes addressed in this paper, we use the same corpus, the 2009 ICWSM Spinn3r dataset (Burton et al., 2009), a publicly-available blog corpus which we also used in our earlier work on lexical formality (Brooke et al., 2010). Blogs are a good resource for broad lexical acquisition because they are very broad in style and content, and are available in essentially unlimited amounts. We use the English Tier 1 (high-quality) blogs that have at least 100 word types, excluding duplicate texts; after this filtering, our dataset contains a total of about 2.4 million blogs.

To build a lexicon for any continuous attribute of interest, we begin by creating *co-occurrence profiles* as follows: First, we select a document frequency range *min-df*, *max-df* that determines a set of profile words P in a corpus S, where for each $p \in P$, the document frequency df_p^S of p in S is limited to be *min-df* $< df_p^S <$ *max-df*; that is, each profile word appears in more than *min-df* documents, but fewer than *max-df* documents in our corpus. Then, given a sample size n and a target word w that we wish to profile, we sample a set of n texts from S which contain the target word (or all the documents where the word appears, if it appears fewer than n times), and count the document frequency of each profile word p in this subcorpus, T_w. We ignore the term frequencies within individual documents because a binary representation is known to be preferred for stylistic dimensions like formality (Brooke et al., 2010), and this seems to be also somewhat true in the domain of polarity, where better results can be obtained when multiple instances of a polar word are discounted (Taboada et al., 2011). To avoid overfitting our statistical model, we do not count a word as appearing with itself. Once we have sub-corpus document frequences $df_p^{T_w}$ for each p, for each profile word p we define the element of our co-occurrence profile vector $\mathbf{v}_p$ as

$$\mathbf{v}_p = \frac{df_p^{T_w}}{\sum_{q \in P} df_q^{T_w}}$$

That is, we normalize each count by the sum across all counts, such that the L1 norm of $\mathbf{v}$ is 1. For our applications here, the dimension of the co-occurrence profile vector is typically in the tens of thousands, but to illustrate the creation of this vector, suppose we choose an extremely narrow document frequency band *min-df*, *max-df*, such there were only three co-occurrence profile words: p_1,p_2,p_3. For some word w, we sample n instances of texts from our corpus which contain w, and find that p_1 appears in 10 of these texts, p_2 in 40 of them, and p_3 in 50 of them. The resulting co-occurrence profile vector is $\mathbf{v} = \langle 0.1, 0.4, 0.5 \rangle$. This profile could be viewed as a distributional vector space representation of the word (Turney and Pantel, 2010), or as an estimate of the probability of each p occurring with w; without any further manipulation, however, we will use it directly as a feature vector for our supervised ranking.

In order to proceed with a supervised approach, we need a ranking of a set of words relevant to the lexical attribute that we wish to acquire; this ranking is specific to the attribute in question, so we discuss this in later sections. Given such a ranking (which, we note, may be partial), we apply SVM^{rank} (Joachims, 2002), which is part of the SVM^{light} set of SVM-based machine learning tools. SVM^{rank} was developed for ranking web page results, and, to our knowledge, has not been applied in this space. SVM^{rank} uses an algorithm which optimizes the Kendall's τ (Kendall, 1955) between a correct ranking r_a and the automatically-generated ranking r_b. The simplest version of τ is based on the number of pairwise rankings which are in concord (C), i.e. both rankings rank the pair relative to each other and the pairwise rankings are the same, or in discord (D), i.e. both rankings rank the pair relative to each other but the rankings offered are contradictory. τ is defined as:

$$\tau(r_a, r_b) = \frac{C - D}{C + D}$$

In practice, this is accomplished in SVM^{rank} by modifying the original SVM algorithm to use as feature vectors the difference between ranked input vectors, rather than the input vectors directly. In the context of this feature space, this means that the model is trained on vectors which represent the differences in the co-occurrence profiles of ranked words; if the word with co-occurrence profile $\mathbf{u}$ is ranked higher than a word with co-occurrence profile $\mathbf{v}$ in our annotation, then SVM^{rank} will try to find a weight vector $\mathbf{w}$ such that $(\mathbf{u}-\mathbf{v})\cdot\mathbf{w}>0$, where $\mathbf{w}$ is constrained to be a sum of co-occurrence profile differences (i.e. the support vectors). Like standard SVM, ranking SVM uses a C parameter which represents the trade-off between margin size and classification errors, though the interpretation of the margin in ranking SVM is less clear. The output of the classification step of SVM^{rank} is a number for each word which can be used directly to rank words, or which can be normalized across words into a scale. If the input rankings also have a continuous numerical representation (which is true in our case for polarity), then this ranking approach can be compared directly to a standard regression which is not directly sensitive to rankings; to maximize comparability, we use the regression function included in SVM^{light} for this purpose. For both, we used a linear kernel.

There is a small number of parameters that need to be set: the sample size n, the frequency range *min-df*, *max-df*, and the SVM C parameter. For each of the two lexical attributes of interest, we carried out independent tuning of these parameters using 5-fold crossvalidation in the training set, carrying out a grid search at powers of 10. We will discuss the values of parameters with respect to specific experiments later, but we mention here that a higher-than-default C, which corresponds to more emphasis on avoiding error rather than maximizing the margin, gave better results for both ranking and correlation, though with diminishing returns. The role of n is primarily to make the method (much) more tractable, but we suspect it might be beneficial to the training of the model for the profiles to be based on a uniform number of examples across word types.

Before we move on to the experimental evaluation, we highlight some intrinsic advantages of this model, independent of performance. As a technique based on large corpus co-occurrence, it has the important property that it can go beyond the limited vocabulary offered by, for instance, WordNet. Since we rely only on co-occurrence, we are not at all limited to individual words (or specific types of words): we could just as easily derive attribute values for n-grams, collocations, or full lexico-grammatical constructions (for instance, distinguishing *high* as related to *price* from *high* as related to *quality*); though our interest here is in general lexical properties, there is no reason this approach could not be used for domain-specific applications, for any lexical units that appear often enough to obtain a reliable co-occurrence profile. Unlike many graph-based techniques, new vocabulary can be classified directly without perturbing the model, potentially in an online fashion if the corpus is properly indexed (which, we note, is by far the most time-consuming step of our method). Though some lemmatization may be required for highly inflectional languages, the method extends easily to any language for which blog data is likely to be available in sufficient quantities. Our approach is more straightforward than most other methods based on co-occurrence, which means fewer arbitrary choices and nuisance variables (such as the dimensionality k or feature weighting typically used in dimensionality-reduction approaches such as LSA); the parameters that we have are fairly well-behaved. Unlike methods which rely only on examples from the extremes of a spectrum to derive a quantification of it, our method naturally integrates examples from the middle of the spectrum (e.g. neutral examples in the case of polarity), but does not inherently require fine-grained quantification of the entire spectrum; in fact, pairwise examples alone could be used for training.

4 Polarity experiments

4.1 Word-level Evaluation

We first consider whether our model can be used to build a lexicon which reflects the polarity spectrum. Our training set of words is taken from the SO-CAL dictionary (Taboada et al., 2011), which has manually assigned SO (polarity) values for words at integer intervals in the range $+5$ to -5. The entire dictionary contains about 5000 words, but we do not use the entire set: first, we restrict our investigation here to adjectives, which allows us to sidestep inflection issues (we do not consider comparative adjec-

tives), and we randomly select only 50 words from each of the 11 possible SO ratings in the dictionary (for a total of 550 words), so as to mimic a (relatively) low-resource situation as we might find working in other languages, and to make it possible to keep the counts equal across SO ratings. Note that the SO-CAL dictionary does not contain neutral words (words not in the dictionary are assumed to be neutral), but we used a set of about 200 hand-marked neutral adjectives that had been excluded from the lexicon during its creation from the words in a set of Epinions product reviews, and which were used for the original dictionary evaluation by Taboada et al. (2011).

After training our model, we evaluate in two test sets. The first test set is the rest of the SO-CAL dictionary, excluding words in the training set as well as those not given a rating by SentiWordNet (see below). Note that this set is not balanced across SO values, since there are many more weakly positive (SO 1 to 3) or weakly negative (SO -1 to -3) words than more-extreme or neutral words; we would argue, though, that this reflects the actual situation in subjective corpora such as product reviews. To test whether we might be overfitting to the product reviews domain, we also test using annotations from the MPQA (Subjectivity) lexicon (Wilson et al., 2005), which was built primarily from news texts.[1] For this, we again include only words that are in SentiWordNet. The MPQA lexicon uses a very different tagging schema than the SO-CAL dictionary, with 3 polarity categories (positive, negative, and neutral) as well as two degrees of subjectivity, weak or strong. Strong or weak subjectivity is defined as how reliable an indicator of subjectivity the word is, which does not directly correspond to the rationale used for the SO-CAL dictionary (which is closer to the notion of force or intensity); the results in Taboada et al. (2011) and our own examination of the lexicon suggest, however, that there is some correlation.[2] Despite this uncertainty, we combined the MPQA tags to form a polarity spectrum: strongly subjective negative, weakly subjective negative, neutral, weakly subjective positive, strongly subjective positive. Given a ranking by our SVM ranker, we evaluate overall pairwise accuracy by considering all possible pairings of words across different ratings within the SO-CAL or MPQA test sets, and count the percentage of those where the ordering of the pair with respect to the polarity spectrum is correctly predicted by the ranking. For a more detailed breakdown, we divide these pairwise comparisons into 3 categories: polarity (pairs which involve one positive and one negative word), neutrality (pairs which have one neutral word), and intensity (pairs which have two words with the same polarity). Note that much work in bootstrapping lexicons for sentiment analysis uses precision and recall, but this is not the most appropriate evaluation metric in this case because our method can assign a rank (and, eventually, an SO value) to any word in the 2 million word vocabulary of our corpus.[3] Here, we are interested only in reliability of these rankings.

During parameter tuning in the development phase, we found that *min-df* $= 10^3$, *max-df* $= 10^5$ was a good choice: in other words, our profile words are words that appear less than once in 24 texts, but more than once in 2400 texts. In the ICWSM, there are 30,852 words that fall into this category, so that is the length of our feature vector. Based on results in the development set, we take $n = 1000$ as our default; larger samples provided no appreciable benefit and were even slightly worse in some cases. For the SVM C parameter, we used 100. We also test using SVM correlation, using the same parameters.

In addition to these variations on our co-occurrence profile technique, we also compare with three independent alternatives. The first is SentiWordNet 3.0 (Baccianella et al., 2010), which uses a random walk method in WordNet to derive positive, negative, and neutral values (which sum to 1) for each synset in WordNet. We follow Taboada et al. (2011) in converting this to a single spectrum for each word by subtracting the negative score from the positive score, and averaging the result across senses for each

[1] There are of course other popular manually-built lexicons, for instance the General Inquirer (Stone et al., 1966), but they tend to have only binary annotations.

[2] One example of where these two dictionaries differ is the word *nervous*, which is tagged as a strongly subjective negative word in the MPQA, but has only a -1 score in the SO-CAL dictionary, since it does not describe a particularly intense negative emotion. An example of a -5 word is *horrendous*, which is also a strongly subjective negative word in the MPQA. Instances of discord where the SO-CAL dictionary is clearly stronger are rarer, but an example is *comprehensive*, which has an SO of 3 in SO-CAL, but is weakly subjective in the MPQA, probably because of its common descriptive uses, such as in the context of insurance and (in the UK) education.

[3] We have not yet build such a lexicon, but, to facilitate comparison, but we are making available raw scores for all the adjectives already contained in at least one of the SO-CAL, MPQA, and SentiWordNet lists (excluding the 550 training words), as well as lists of specific words used for training and testing. These resources can be found at http://www.cs.toronto.edu/~jbrooke/rankingpolarity.zip .

Table 1: Results of polarity experiments. Left side of table shows pairwise accuracy (%) for various sentiment lexicon ranking methods in SO-CAL and MPQA test sets. Pol. = Pairs with different polarity; Neu. = Pairs with at least one neutral word; Int. = Pairs with the same polarity, but different intensities. Right side of table shows text polarity classification accuracy (%) in Epinions Corpus for various adjective lexicons. Bold is best in column.

	SO-CAL words				**MPQA words**				**Epinions texts**
Method	**Pol.**	**Neu.**	**Int.**	**All**	**Pol.**	**Neu.**	**Int.**	**All**	**Acc.**
SentiWordNet	82.3	72.3	57.4	72.1	82.8	72.0	49.9	72.0	65.8
LSA	83.5	70.6	63.0	74.5	82.5	70.2	64.0	75.8	66.3
PMI	86.3	73.6	65.8	77.3	84.5	73.4	61.6	76.9	68.0
Profile regression	80.2	67.6	59.6	71.2	77.6	69.0	**74.7**	75.8	60.3
Profile ranking	**88.6**	**75.7**	**67.5**	**79.4**	**87.5**	**74.0**	56.5	**77.0**	**71.8**

word (in that paper, they also considered using only the most common sense but found the results to be indistinguishable). The second alternative uses the semi-supervised LSA-based method of Turney and Littman (2003). For the first step, singular value decomposition, we use a binary term-document matrix with the same ICWSM texts as our supervised model, with $k = 500$ (a fairly standard choice). In the second step, which involves calculating the cosine similarity with a set of seed terms using the LSA vectors and then taking the difference, the positive and negative seeds are just the training instances for our supervised model (neutral terms are discarded). Our third comparison is the PMI approach of Turney (2002), which is still popular: for instance, PMI was used to built a Twitter sentiment lexicon in the winning entry in a recent shared task (Mohammad et al., 2013). Because they have access to the same corpus and even the same example words as our method, the LSA and PMI alternatives are most directly comparable to ours.

The results for the word-level polarity experiments are shown in the left side of Table 1. In the SO-CAL test set, the results are clear: our SVM ranking method is preferred over alternatives, across all the different categories of pairwise comparison. The relative difficulty of each pair type reflects the average distance between relevant pairs on the spectrum, as expected. Surprisingly, the correlation method, despite using the same feature input as the ranking method, is the worst performing method here, though SentiWordNet is only marginally better, while LSA falls roughly in the middle of the range, and PMI is the strongest competitor. One potential criticism is that a ranking method is likely to have an advantage when evaluating by rank. This is true, but we think that relative rank among words is fundamental to the notion of a spectrum, whereas the bucketing of words into evenly spaced integer ratings is just an annotation convenience. That said, our output ranking is perhaps too fine-grained in comparison to our input (offering a full ranking for all words), and it would be desirable if our ranking algorithm allowed us to encourage some words to be ranked the same.

Although SVM ranking is also the best method on the MPQA test set, the results are marginal as compared to the SO-CAL test. Part of this could be a moderate amount of domain overfitting, or perhaps the ranking method is better at fine-grained scales relative to the other methods. However, the most obvious difference between the test sets appears relative to the intensity comparison, where the profile ranking performance is relatively poor. This is likely attributable to the differences between the two kinds of annotations: the SVM ranking method learns the SO-CAL intensity scale fairly well, but this actually becomes a handicap when degree of subjectivity and not force is the deciding factor; on the other hand, corpus-based models which did relatively poorly in all the other evaluations (profile regression, LSA) actually do somewhat better in MPQA intensity than their most comparable alternatives (profile ranking, PMI) to a degree that is in fact proportional to their relative inferiority elsewhere, suggesting that sensitivity to degree of subjectivity might be interfering with acquisition of the SO-CAL polarity spectrum. Interestingly, the value provided by SentiWordNet does not seem to correspond well to either of these interpretations of intensity, since it does rather poorly with respect to both.

4.2 Text-level Evaluation

The most common use of polarity lexicons is the task of text polarity classification (Turney, 2002), identifying whether an opinionated text is positive or negative. In this section, we convert the initial output of the models to polarity lexicons with an appropriate scale so that we can use the SO-CAL software (Taboada et al., 2011) to carry out text sentiment analysis with our alternative adjective lexicons rather than its original, manually-built one. SO-CAL is an unsupervised lexical sentiment analysis system with a number of built-in features, e.g. handling of negation and intensification, that improve the accuracy of the model, particularly when using a fine-grained, high-precision lexicon. Taboada et al. evaluate across 4 corpora of balanced product reviews. For our evaluation, we use one of those corpora, a set of 400 product reviews from Epinions, with 50 balanced texts from each of 8 product categories (movies, books, cars, computers, cookware, hotels, phones, and music). Taboada et al. call this corpus Epinions 2, to distinguish it from the Epinions corpus that the SO-CAL dictionary was built from. They report an accuracy of 80% using SO-CAL with all word types and features enabled.

Our interest here is to test the influence of lexicon quality on polarity detection. Coverage, though of course important, is actually a potential source of noise: Low coverage can naturally result in low performance, but Taboada et al. point out that high coverage can also cause problems, when many of the rarer words added to the lexicon, even when human-tagged, are not relevant to the primary sentiment of the text, but rather irrelevant aspects like (in the movie or book domain) character descriptions. Steps can be taken to mitigate this by identifying relevant sentiment (Scheible and Schütze, 2013), but here we sidestep this problem by forcing our lexicons to have exactly the same coverage by limiting them to words that appear in the static SentiWordNet lexicon. Again, we also consider only adjectives here.

To build the lexicon for this evaluation we used a different training set: it is not possible to take 50 samples from each SO rating in the SO-CAL adjective lexicon and not have training words that also appear in the corpus, which we explicitly wanted to avoid.[4] Instead, we train using all the adjectives in the SO-CAL dictionary that either don't appear in the Epinions corpus or don't appear in SentiWordNet (since we are limiting our output lexicon to SentiWordNet words). This results in a much larger training set than in the word-level evaluation (about 1500 words), but they are distributed unevenly across SO ratings. Relative to the word-level evaluation, this is closer to the situation if we were using the entire SO-CAL dictionary to expand the lexicon. We use the same set of training words as seeds for LSA.

To convert SentiWordNet to a SO-CAL-compatible dictionary, we simply multiply the raw score, guaranteed to be between -1 and $+1$, by 5, creating a range of -5 to $+5$. For the raw scores for our other three options (LSA, profile regression, and profile ranking), we linearly scale the raw score so that the mean within the lexicon is 0, and a SO $+5$ word is at the third standard deviation away from the mean; we choose a rather severe scaling so that there are only a handful of words in the lexicon whose absolute value is over 5, which SO-CAL is not designed for.

The results of this evaluation are shown on right side of Table 1. Profile ranking is once again dominant, almost 4 percentage points better than the second-best option. The ordering of the lexicons here is exactly the same as we saw for the word-level evaluation in SO-CAL, though SentiWordNet does somewhat better than would be expected from those scores. Again, regression does quite poorly despite having access to the same feature vectors as the ranking method. We note that our results here are also markedly better than all the other automatic lexicons compared by Taboada et al., namely a PMI-derived lexicon based on Google counts (Taboada et al., 2006), and a binary lexicon built by expanding entries in a thesaurus (Mohammad et al., 2009), and are even a bit better than using the human-tagged (binary) annotations from the General Inquirer (Stone et al., 1966), though we are still quite a long way from what is possible with the full manual SO-CAL dictionary. Since the quality of the lexicon is directly reflected in our polarity classification scores here, it is not surprising that our gold-standard lexicon is superior; in this context, it should be viewed as an upper bound. Nevertheless, we have strong evidence here that our co-occurrence profile ranking method is a step in the right direction relative to other methods for automatically building lexicons.

[4] For all of our evaluations in this paper we were careful never to use the score of a word which appeared in our training set; the drawback of this is that our training set size is not constant.

5 Formality Experiments

5.1 Word-level Evaluation

Though the lexicon is perhaps more fundamental to distinctions of polarity than is the case for formality, nevertheless formality is strongly expressed through word choice; for instance, in English using the word *dude* to address a socially-powerful stranger would generally be unacceptable, and it would be very strange to address a good friend as *sir*, except as a joke. These are not isolated examples: a huge portion of the vocabulary is marked to some degree in this fashion, and requires special attention when moving across text genres or social situations. Word length (in English, at least) and word frequency can be used as a simple proxy (longer, rarer words are, on average, more formal), but the example above belies this approach: *sir* is a shorter word than *dude*, and it is not immediately obvious that it would appear less in, for instance, a news corpus, than *dude*.

In previous work (Brooke et al., 2010), we used the LSA co-occurrence method of Turney and Littman (2003) discussed in the previous section to derive a formality lexicon using the ICWSM (which was the best among various corpora tested, including the BNC). For testing, we used a set of 399 synonym pairs that were pulled from a writing manual focused on word choice, *Choose the Right Word* (CTRW) (Hayakawa, 1994), where the author explicitly compared words for their formality, showing that co-occurrence was a better approach to identifying lexical formality than proxies related to word length or word frequency. We note that many of the distinctions in the CTRW set are quite subtle, for example *determine* vs. *ascertain*, both of which seem at least somewhat formal, though *ascertain* was judged by the expert to be the more formal of the two. In this section we will build a formality ranking using our profile ranking method, and show that it is better than the LSA method in the CTRW dataset. Here, we follow our earlier work in using a much smaller k value (20) than is typical for topical uses of LSA, which we found was better for this dataset, since major stylistic differences seem to be mostly captured in the first few dimensions after dimensionality reduction, a result which is consistent with the work of Biber (1988) looking at differences across registers.

Unlike for polarity, there is no resource available that offers a full scale of formality for a large number of words, and the set used in our initial work on formality has only extreme, handpicked words. In more recent work (Brooke and Hirst, 2013), we used a larger set of words (900) that included a variety of different styles that had been tagged by a group of 5 annotators. In that work, we did not use the term "formality," but one of our styles, *colloquial*, corresponds to the informal end of the spectrum, and two other styles, *objective* and *literary*, can both be viewed as social-distancing language.[5] The words tagged by annotators as belonging to neither of these categories will serve as the middle of this spectrum. Compared to our polarity lexicon, our training set therefore is much more coarse-grained (with only 3 rankings, as compared to 11 for polarity), but the pairwise relationships can be used to build a fine-grained scale. As before, we remove all words that overlap with our test set.

With respect to parameters, we use the same n as in the polarity experiments, but in development we saw better performance from a higher C (10,000) and a lower bottom bound on the document frequency range, $df\text{-}min = 10^2$. The latter might reflect the fact that rare vocabulary has a strong tendency to be associated with extreme formality, though there is also a limit to this, since if a word is so rare that it hardly ever co-occurs with anything, it cannot possibly be useful for training no matter how good an example of extreme formality it is. The results using these settings are shown in the left side of Table 2. Again, our profiling method is clearly better than lexicon-based alternatives. LSA outperforms PMI in the word-level task, a result which is consistent with our other work in stylistic lexical induction (Brooke and Hirst, 2013),[6] and all the co-occurrence methods are well above the word-length baseline.

Figure 1 contains a more detailed analysis of the influence of individual document frequency bands: we

[5]The objective dimension corresponds roughly to the style of a technical document, while the literary dimension involves flowery, even archaic language that suggests a literary sophistication. Contrast *so* with synonyms *therefore* (objective) and *thus* (literary), which are both more formal, but in different ways.

[6]We suspect this is due to the fact that LSA vectors encode information about word frequency: even when the vector norm is controlled for, we have found that the LSA vectors of high- and low-frequency words have consistently different distributions, which may help in identifying extremely low frequency, highly formal words appearing in the CTRW dataset; by contrast, PMI and other probability-based approaches seem to behave more erratically when presented with low-frequency items.

Table 2: Results of the formality experiments. The second column shows pairwise accuracy of different models identifying the more formal of two synonyms in CTRW test set. The third column shows average correlation with two human 5-point Likert-scale formality annotations of the 500 sentence test set.

Method	CTRW words	Sentence Evaluation
Word length	63.7	0.36
LSA	78.7	0.49
PMI	72.2	0.52
Profile ranking	**86.5**	**0.55**

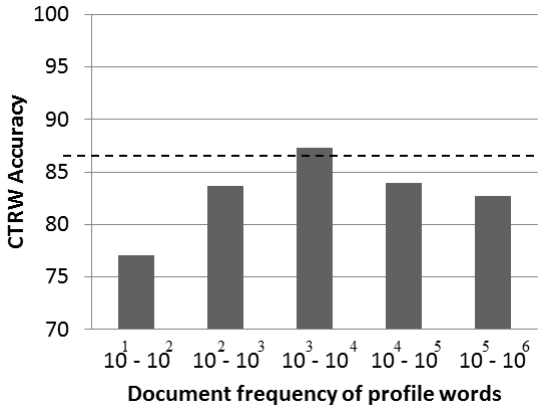


Figure 1: Pairwise accuracy in the CTRW test set for various frequency bands. Dotted line represents performance using best parameters from development phase, i.e. *max-df* $= 10^2$, *max-df* $= 10^5$.

built models for each of our frequency bands (ranges between two consecutive powers of ten), and tested them in the CTRW corpus. The flat dotted line represents the larger band we used based on development performance, 10^2–10^5. We see that accuracy peaks at 10^3 to 10^4 *df* band, at a value (87.3%) which is higher than we saw with the larger band chosen based on the development set. The words in this band are fairly uncommon, appearing less than once in 240 texts, but greater than once in 2400 texts; still, as a group they provide enough evidence to make a strong determination of formalty.

5.2 Sentence-level Evaluation

Lahiri and Lu (2011) report on the creation of 5-point Likert scale annotations of sentence-level formality, with two ratings for each of 500 sentences taken from separate texts in a diverse corpus which includes news, blogs, forums, and academic papers (Lahiri et al., 2011). In this section, we use this annotation to carry out an extrinsic evaluation of our lexical formality ratings. As far as we are aware, this is the first use of this annotation for evaluating metrics of formality. We extract all lexical words (verbs, adjectives, adverbs, and nouns, though we omit proper nouns) from the sentences and use the 3-way formality annotation with these words removed to create LSA, PMI, and profile ranking models, which are then used to create a formality lexicon for these words, using the same method we used to create the SO lexicon. Given a lexicon, we averaged the formality score across each sentence (ignoring duplicate items) to get a formality score for each sentence. We calculate Pearson's correlation coefficient between our score and each of the two annotators, and then average the result. For comparison, the correlation between the two human annotators is 0.60.

The results in Table 2 indicate that the preference for the profile ranking method seen in the CTRW set extends directly to sentence-level formality ranking, and the level of correlation reached by the profile ranking method approaches correlation between humans. This supports our claim that lexical choice is very important to formality: our results here indicate that humans with access to other indicators of formality (for instance, use or avoidance of particular syntactic constructions) agree only slightly more with each other than our lexicon-only model does with them.

6 Conclusion

In this paper we have presented a novel approach to determining where a word lies on a spectrum, using just counts of words that it tends to appear with and an SVM ranking algorithm, both of which are key components to its success. We have shown that it can be applied to at least two continuous attributes of interest in computational linguistics, namely polarity and formality, and that the benefits of this method relative to established alternatives are visible not just in direct lexicon evaluation, but also in the NLP tasks where these lexicons can be used. Even with a relatively small set of words to train with, we see little sign of overfitting, and although we have focused on a small set of words here, our method is efficient enough that it could easily be applied to a much larger set of lexicogrammatical units, though we will also have to derive ways to filter out unreliable assignments to reduce overall noise. Other future work will involve looking at other spectra, other languages, other supervised ranking models, and improving our performance generally by being more selective of profile words or training examples or by refining our rankings by including other sources of information such as WordNet.

Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council of Canada and the MITACS Elevate program. Thanks to Shibamouli Lahiri and Maite Taboada for the use of their resources and their feedback.

References

Alina Andreevskaia and Sabine Bergler. 2006. Semantic tag extraction from WordNet glosses. In *Proceedings of 5th International Conference on Language Resources and Evaluation (LREC '06)*.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May.

Douglas Biber. 1988. *Variation Across Speech and Writing*. Cambridge University Press.

Julian Brooke and Graeme Hirst. 2013. Hybrid models for lexical acquisition of correlated styles. In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP '13)*.

Julian Brooke, Tong Wang, and Graeme Hirst. 2010. Automatic acquisition of lexical formality. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*, Beijing.

Kevin Burton, Akshay Java, and Ian Soboroff. 2009. The ICWSM 2009 Spinn3r Dataset. In *Proceedings of the Third Annual Conference on Weblogs and Social Media (ICWSM '09)*, San Jose, CA.

John Carroll, Guido Minnen, Darren Pearce, Yvonne Canning, Siobhan Devlin, and John Tait. 1999. Simplifying English text for language impaired readers. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL'99)*.

Ilia Chetviorkin and Natalia Loukachevitch. 2012. Extraction of Russian sentiment lexicon for product meta-domain. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING '12)*.

Max Coltheart. 1980. *MRC Psycholinguistic Database User Manual: Version 1.* Birkbeck College.

Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL '13)*.

Gerard de Melo and Mohit Bansal. 2013. Good, great, excellent: Global inference of semantic intensities. *Transactions of the Association for Computational Linguistics*, 1.

Ahmed Hassan and Dragomir Radev. 2010. Identifying text polarity using random walks. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, (ACL '10).

Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics (ACL/EACL '97)*.

S.I. Hayakawa, editor. 1994. *Choose the Right Word*. HarperCollins Publishers, second edition. Revised by Eugene Ehrlich.

Francis Heylighen and Jean-Marc Dewaele. 2002. Variation in the contextuality of language: An empirical measure. *Foundations of Science*, 7(3):293–340.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *9th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '02)*.

Nobuhiro Kaji and Masaru Kitsuregawa. 2007. Building lexicon for sentiment analysis from massive collection of HTML documents. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '07)*.

Jaap Kamps, Maarten Marx, Robert J. Mokken, and Maarten de Rijke. 2004. Using WordNet to measure semantic orientation of adjectives. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC '04)*.

Hiroshi Kanayama and Tetsuya Nasukawa. 2006. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP '06)*.

Maurice Kendall. 1955. *Rank Correlation Methods*. Hafner.

Paul Kidwell, Guy Lebanon, and Kevyn Collins-Thompson. 2009. Statistical estimation of word acquisition with application to readability prediction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP'09)*.

Soo Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*.

Beata Beigman Klebanov, Nitin Madnani, and Jill Burstein. 2013. Using pivot-based paraphrasing and sentiment profiles to improve a subjectivity lexicon for essay data. *Transactions of the Association for Computational Linguistics*, 1.

Shibamouli Lahiri and Xiaofei Lu. 2011. Inter-rater agreement on sentence formality. http://arxiv.org/abs/1109.0069 .

Shibamouli Lahiri, Prasenjit Mitra, and Xiaofei Lu. 2011. Informality judgment at sentence level and experiments with formality score. In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing'11)*.

Zhifei Li and David Yarowsky. 2008. Mining and modeling relations between formal and informal Chinese phrases from web corpora. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*.

Haiying Li, Arthur. C. Graesser, and Zhiqiang Cai. 2013. Comparing two measures of formality. In *Proceedings of the Twenty-sixth International Florida Artificial Intelligence Research Society Conference*.

Saif Mohammad, Cody Dunne, and Bonnie Dorr. 2009. Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP '09)*.

Saif M. Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (SEMSTAR '13)*.

Alejandro Mosquera and Paloma Moreda. 2012. A qualitative analysis of informality levels in web 2.0 texts: The Facebook case study. In *Proceedings of the LREC workshop:@NLP can u tag #user_generated_content*, pages 23–29.

Charles E. Osgood, George Suci, and Percy Tannenbaum. 1957. *The measurement of meaning*. University of Illinois Press, Urbana, IL.

Kelly Peterson, Matt Hohensee, and Fei Xia. 2011. Email formality in the workplace: A case study on the Enron corpus. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL '11)*, Portland, Oregon.

Delip Rao and Deepak Ravichandra. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Lingusitics*.

Christian Scheible and Hinrich Schütze. 2013. Sentiment relevance. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL '13)*.

Fadi Abu Sheika and Diana Inkpen. 2012. Learning to classify documents according to formal and informal style. *Linguistic Issues in Language Technology*, 8.

Vera Sheinman and Takenobu Tokunaga. 2009. Adjscales: Visualizing differences between adjectives for language learners. *IEICE Transactions on Information and Systems*, 92(8):1542–1550.

Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilivie. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press.

Maite Taboada, Caroline Anthony, and Kimberly Voll. 2006. Methods for creating semantic orientation dictionaries. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC '06)*.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307.

Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting semantic orientations of words using spin model. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL '05)*.

Peter D. Turney and Michael Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21:315–346.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141.

Peter D. Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*.

Peter D. Turney. 2002. Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02)*.

Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 2010. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL '10)*.

Svitlana Volkova, Theresa Wilson, and David Yarowsky. 2013. Exploring sentiment in social media: Bootstrapping subjectivity clues from multilingual Twitter streams. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL '13)*.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP '05)*.

Unsupervised extraction of semantic relations using discourse cues

Juliette Conrath Stergos Afantenos Nicholas Asher Philippe Muller
IRIT, Université Toulouse & CNRS, Univ. Paul Sabatier, 118 Route de Narbonne, 31062 Toulouse
{firstname.lastname@irit.fr}

Abstract

This paper presents a knowledge base containing triples involving pairs of verbs associated with semantic or discourse relations. The relations in these triples are marked by discourse connectors between two adjacent instances of the verbs in the triple in the large French corpus, frWaC. We detail several measures that evaluate the relevance of the triples and the strength of their association. We use manual annotations to evaluate our method, and also study the coverage of our resource with respect to the discourse annotated corpus Annodis. Our positive results show the potential impact of our resource for discourse analysis tasks as well as other semantically oriented tasks like temporal and causal information extraction.

1 Introduction

Relational lexical resources, which describe semantic relations between lexical items, have traditionally focused on relations like synonymy or similarity in thesauri, perhaps including some hierarchical semantic relations like hyperonymy or hyponomy or part-whole relations as in the resource Wordnet (Felbaum, 1998). Some distributional thesauri contain more varied relations, see e.g. (Grefenstette, 1994), however these relations are not typed. The lexical semantics given by FrameNet (Baker et al., 1998) does include causal and temporal relations, as does Verbocean (Chklovski and Pantel, 2004), but coverage is limited and empirical validation of these resources is partial and still largely remains to be done.

Lexical relations, in particular between verbs, are nevertheless crucial for understanding natural language and for many information processing tasks. They are needed for textual inference, in which one has to infer certain relations between eventualities (Hashimoto et al., 2009; Tremper and Frank, 2013), for information extraction tasks, like finding temporal relations between eventualities mentioned in a text (UzZaman et al., 2013), for automatic summarization (Liu et al., 2007), and for discourse parsing in the absence of explicit discourse markers (Sporleder and Lascarides, 2008).

In this paper we report on our efforts to extract semantic relations essential to the analysis of discourse and its interpretation, in which links are made between units of text or rather their semantic representations as in (1) in virtue of semantic information about the two main verbs of those clauses.

(1) The candidate demonstrated his expertise during the interview. The committee was completely convinced.

We follow similar work on the extraction of causal, temporal, entailment and presuppositional relations from corpora (Do et al., 2011; Chambers and Jurafsky, 2008; Hashimoto et al., 2009; Tremper and Frank, 2013), though our goals and validation methods are different. While one of our goals is to use this information to improve performance in predicting discourse relations between clauses, we believe that such a lexical resource will have other uses in other tasks in which semantic information is needed.

Discourse analysis is a difficult task. Rhetorical relations are frequently implicit and require for their identification inference using diverse sources of lexical and compositional semantic information. In the Penn Discourse Treebank corpus for example, 52% of the discourse relations are unmarked (Prasad et

This work has been supported by the French agency Agence Nationale de la Recherche (ANR-12-CORD-0004).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2184–2194, Dublin, Ireland, August 23-29 2014.

al., 2008). Accordingly, annotation with discourse structure is a slow and error prone task, and relatively little annotated data is currently available ; and so machine learning approaches have had limited success in this area. Our approach addresses this problem, using non annotated data with features that can be automatically detected to find typical contexts (pairs of discourse units) in which various discourse relations occur. We suppose with (Sporleder and Lascarides, 2008; Braud and Denis, 2013) that such contexts display regular lexical associations, in particular with verbs in those discourse units. An explicit, manually compiled list of all possible associations between two verbs and the semantic relations they suggest is infeasible, so we present here an automatic method for compiling such a list, inspired by the Verbocean project (Chklovski and Pantel, 2004).

Our hypothesis, supported by existing corpora, is that adjacent clauses are often arguments of discourse relations. When these clauses contain certain adverbs or other discourse connectors, we can recover automatically one or more discourse relations that we associate with the main verbs of those clauses. We extract triples consisting of the two verbs and a semantic relation from a large corpus with the aim of inferring that such a pair of verbs can suggest the semantic relation even in the absence of an explicit discourse marker. We thus also suppose, with (Sporleder and Lascarides, 2008; Braud and Denis, 2013), that such discourse markers are at least partially redundant ; inferring a discourse relation between two clauses relies not only the marker but on the two verbs in the related clauses as well. All of our work has been done on French data.

Our paper is organized as follows. We describe first the knowledge base of verb semantic relation triples that we have constructed (section 2) ; we then present our methods for isolating verb pairs implicating discourse or temporal information (section 3). A third section describes our methods of evaluation (section 4) and a fourth discusses related work (section 5).

2 Exploring relations between verbs in a corpus

We built a knowledge base (V^2R) [1] using the frWaC corpus(Baroni et al., 2009). frWaC contains about 1.6 billion words and was collected on the Web on the .fr domain. We first parsed the documents in our corpus using BONSAI [2], which first produced a morpho-syntactic labeling using MElt (Denis and Sagot, 2012) and then a syntactic analysis in the form of dependency trees via a French version of the MaltParser (Nivre et al., 2007).

Our goal is to find pairs of verbs linked by a relation explicitly marked by a discourse connector in the corpus, as an indication of a regular semantic relation between the two verbs. The relations we have considered are common to most theories of discourse analysis, and they can be grouped into four classes (Prasad et al., 2008) : causal (*contingency*) relations, temporal relations, comparison relations (mainly contrast type relations), and expansion relations (e.g. elaboration or continuation).

To find explicitly marked relations, we used a lexicon of discourse connectors for French, the manually constructed LEXCONN resource (Roze et al., 2012) [3]. LEXCONN includes 358 connectors and gives their syntactic category as well as associated discourse relations inspired from (Asher and Lascarides, 2003). Some connectors are ambiguous in that they are associated with several relations. We used only the unambiguous connectors (263 in all) in LEXCONN, as a first step. We regrouped the LEXCONN relations into classes [4] : explanation relations (*parce que*/because) and result (*ainsi*/thus) form the causal class ; temporal relations (*puis, après que*/then,after that) form the narration group. We also considered other relations like contrast (*mais*/but), continuation (*et, encore*/and,again), background (*alors que*/while), temporal location (*quand, pendant que*/when), detachment (*de toutes façons*/anyway), elaboration (*en particulier*/in particular), alternation (*ou*/or), commentary (*au fait*/by the way), rephrasing (*du moins*/at least), and evidence (*effectivement*/indeed).

We searched our syntactically parsed corpus for connectors. When a connector is found and its syntactic category verified, if it is close enough to the root of the sentence (at most one dependency link from the root), we look for an inter-sentential link. The first verb of our pair corresponds in this case

1. Available as an SQLite database at `https://dl.dropboxusercontent.com/u/78938139/v2r_db`
2. `http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html` or (Candito et al., 2010)
3. Freely available at : `https://gforge.inria.fr/frs/download.php/31052/lexconn.tar.gz`.
4. We illustrate each relation with examples of potentially ambiguous markers.

to the last verb of the previous sentence in the case of connectors for narration, or to its main verb for all the other relations. We search for the second verb in the pair within a window of two dependency links after the connector. If the connector is not close enough to the root of the sentence, we look for a intra-sentential link. In this case, we look for the two verbs of the pair in the same sentence within a forward and backward window of two dependency links.

If two verbs are found, we examine their local context to better characterize their usage and to improve our results. If one of the verbs is a modal or support verb, we look for the verb dependent on the modal or support verb and use that as the verb in our pair (if it exists), while keeping the presence of the support verb in memory. Unlike support verbs, we use the presence of a negation or a reflexive particle in the local context to distinguish verbs with different meanings ; e.g., *comprendre*/understand vs. *ne pas comprendre*/not understand, *agir*/act vs. *s'agir*/concern are all distinct entries. To get at different verb senses, we search for idiomatic usage of prepositions using the Dicovalence resource (Van Den Eynde and Mertens, 2010), which contains valency frames for more than 3700 simple French verbs. We also use the Lefff resource (Sagot, 2010) to find idiomatic verbal locutions. We also encode other information that do not lead to distinct lexical entries : tense, and voice.

Relation	Distribution
contrast	50,104%
cause	33,108%
continuation	8,243%
narration	6,362%
background	1,853%
temporal localisation	0.177%
detachement	0.149%
elaboration	0.002%
alternation	0.002%

TABLE 1 – Distribution of relations in V^2R ;commentary, reformulation and evidence occur with negligible frequency.

Once we have obtained a list of verb pairs associated with a connector, we aggregate this data to get a list of triple types (verb1, verb2, relation). Given that we have used only unambiguous connectors (so classified by LEXCONN), the association of a relation with a connector is immediate. We associate to each triple type the number of intra-sentential, inter-sentential and total number of occurrences. The other features mentioned above are stored in a separate table.

Our method has isolated more than 1 million distinct types of triples for V^2R and 2 million occurrences, of which 95% are intra-sentential [5]. Among these triples, 6.2% have 5 or more occurrences.

Table 1 summarizes the distribution of triples by relation in V^2R. Note that triples with contrast and causal relations comprise the majority. This does not mean that these are the most frequent relations in the corpus but only that they are the most frequently marked by the connectors we considered. This makes for a very different distribution than that of the French manually annotated discourse corpus Annodis (Afantenos et al., 2012).

3 Measuring the association of a pair of verbs with a relation

In the last section we presented our extraction method. We now present the measures we have used to rank verb pairs with respect to the strength of their association with a particular discourse relation. We adapted versions of standard lexical association measures like PMI (*pointwise mutual information*) and their variants, as well as some measures specific to the association of a causal relation between items (Do et al., 2011). We also experimented with a new measure specifically designed for our knowledge base.

Measures of lexical association used in research on co-occurrences in distributional semantics pick out significant associations, taking into account the frequency of the related items. We examined over 10 measures ; we discuss the ones with the best results (see section 4). One simple measure, PMI, and its variants, *normalized*, *local* (Evert, 2005), *discounted* (Lin and Pantel, 2002), which are designed to reduce biases in the original measure, work well. The idea behind PMI is to estimate whether the probability of the co-occurrence of two items is greater than the *a priori* probability of the two items appearing independently. In distributional semantics, the measure is also used to estimate the significance of two items co-occurring with a particular grammatical dependency relation like the subject or object relation between an NP and a verb. This use of PMI measures over triples in distributional semantics fits perfectly with our task of measuring the significance of triples consisting of a pair of verbs and

5. The low proportion of inter-sentential occurrences comes from our conservative scheme for finding these occurrences, which uses only those connectors at the beginning of the second sentence. Other schemes are possible but would, we fear, introduce too much noise into the data.

a particular semantic or discourse relation ; our PMI measures estimate whether the co-occurrence of two items with a particular discourse relation is higher than the *a priori* probability of the three items occurring independently. Our measures consider co-occurrences of two lexical items in a certain relation denoted by an explicit discourse marker. PMI and normalized PMI are defined as :

$$PMI = \log(\frac{P(V_1, V_2, R)}{P(V_1) \times P(V_2) \times P(R)}) \qquad PMI\_normalized = \frac{PMI}{-2\log(P(V_1, V_2, R))}$$

Indeed, when we have a complete co-occurrence of the three items, we have : $P(V_1) = P(V_2) = P(R) = P(V_1, V_2, R)$, and $PMI = -2\log(P(V_1, V_2, R))$. The values of normalized PMI lie between -1 and 1, approaching -1 when the items never appear together, taking the value 0 in the case of independence, and the value 1 when they always appear together. We also considered a weighted PMI measure (Lin and Pantel, 2002) that corrects the bias of PMI for rare triples.

A specificity measure (Mirroshandel et al., 2013), originally used to measure the precision of subcategorization frames, also performed well :

$$specificity = \frac{1}{3} \times (\frac{P(V_1, V_2, R)}{\sum_i P(V_1, V_i, R)} + \frac{P(V_1, V_2, R)}{\sum_i P(V_i, V_2, R)} + \frac{P(V_1, V_2, R)}{\sum_i P(V_1, V_2, R_i)})$$

A version of Do et al. (2011)'s measure for triples involving causal relations did not fare so well on other types of relation. The definition of the measure can be found in (Do et al., 2011). [6]

Finally, we investigated a measure that evaluates the contribution of each element in the triple to the significance measure (this measure is similar to specificity).

$$W_{combined}(V_1, V_2, R) = \frac{1}{3}(w_{V_1} + w_{V_2} + w_R)$$

with : $w_{V_1} = \frac{P(V_1,V_2,R)}{\max_i(P(V_i,V_2,R))}$, $w_{V_2} = \frac{P(V_1,V_2,R)}{\max_i(P(V_1,V_i,R))}$, and $w_R = \frac{P(V_1,V_2,R)}{\max_i(P(V_1,V_2,R_i))}$.

4 Evaluating extracted relations

We evaluated V^2R in several ways ; we provided : (i) an intrinsic evaluation of the relations between verbs (section 4.1) and (ii) an extrinsic evaluation where we evaluated the coverage of the resource on a discourse annotated corpus and its potential to help in predicting discourse relations in contexts with no explicit marking (section 4.2).

4.1 Intrinsic evaluation

Our intrinsic evaluation first evaluates the feasibility of assigning an "inherent" semantic link to a verb pair, independently of any linguistic context. For example, is it possible to judge that there is a typical causality link between *push* and *fall*, in scenarios where they share some arguments (subject, object, ...), these scenarios being left to the annotator's imagination (section 4.1.1). In a second stage, we selected several verb pairs linked with different relations in V^2R, and 40 contexts in which these verbs occur together in the original corpus, to judge the semantic link in context (section 4.1.2).

In both cases we restricted the study to three relation groups : causal, contrastive, and narrative. These are the most often marked relations and correspond to different types of links with a meaningful semantic aspect (as opposed to the "continuation" relation for instance, which is often marked too).

4.1.1 Out of context evaluation

For out of context judgments, we adopted the following protocol : one of the authors chose for each relation 100 verbs with equivalent proportions of good and bad normalized PMI scores. Then the other

6. We simplified their measure by ignoring IDF (inverse document frequency) and the distance between the verbs, as neither measure applies to our task.

three authors judged the validity of associating each of the 300 pairs with the corresponding relation, without any knowledge of the source of these pairs.

We measured the inter-annotator agreements with Cohen's Kappa (Carletta, 1996), which resulted in : 0.17 for cause, 0.42 for narration and 0.56 for contrast as mean values. If a 0.6 kappa serves a measure for a feasible semantic judgment task, out of context judgments appear very difficult, with only contrastive pairs as a relative exception. We decided to only consider judgments about contrast, after an adjudication phase, and we evaluated the measures presented in section 3 to see if they could discriminate between the two verb groups, those judged positively or negatively according to human annotations. A Mann-Whitney U statistical test showed all of our measures to be discriminative, with the exception of raw co-occurrence counts for which p>0.05.

4.1.2 In context evaluation

We also judged associations in context. This task was easier and also gave more fine-grained results, because with it we can quantify the degree of association, and the typicality of the link, as a proportion of contexts where the two verbs appear together in a given semantic relation. We can then observe if this proportion is correlated with the association measures we already presented. Nevertheless, this is a costly way of evaluating a verb pair, as we require a number of judgments on each pair. It is also not easy to sample the possible pairs with different values to be able to observe significant correlations, because we cannot predict in advance how they will be judged by the annotators.

We selected 40 contexts for each of the 15 pairs of verbs we chose, 5 for each of the target relation (cause, narration, contrast). Selected pairs range over different values of normalized PMI, again chosen by one of the authors independently of the others, who annotated the 600 contexts. Prior to adjudication, raw agreement was 78% on average, for an average kappa of 0.46 (and a maximum of 0.49). These values seem moderately good, as the task is also rather difficult.

Verb pair	translation	association /human
Cause		
inviter/souhaiter	invite/wish	12.8%
promettre/élire	promise/elect	25.6%
aimer/trouver	like/find	38.5%
bénéficier/créer	benefit/create	51.3%
aider/gagner	help/win	53.8%
Contrast		
proposer/refuser	propose/refuse	59.0%
augmenter/diminuer	increase/decrease	64.1%
tenter/échouer	try/fail	64.1%
gagner/perdre	win/lose	71.8%
autoriser/interdire	authorize/forbid	74.4%
Narration		
parler/réfléchir	speak/think	42.5%
acheter/essayer	buy/try	70.0%
atteindre/traverser	reach/cross	77.5%
commencer/finir	begin/end	80.0%
envoyer/transmettre	send/transmit	82.5%

TABLE 2 – For each relation, the list of verb pairs manually evaluated in context (and an approximate translation), and the association percentage resulting from the adjudicated human annotation.

Table 2 shows the results after adjudication : for each pair, the proportion of contexts in which the considered relation is judged to appear.

We computed two correlation values between the association ratio in contexts manually annotated and each association measure considered : one based on all annotated contexts, and one on the subset of contexts devoid of explicit markers of a semantic relation (implicit contexts). The latter is important to quantify the actual impact of the method, since explicit marking is already used as the basis of verb association in the same corpus. Implicit contexts, however, never appeared in the computation of the verb pair associations.

	normalized PMI	specificity	W_combined	discounted PMI	PMI	local PMI	U_do	raw frequency
Global correlation	0.749	0.747	0.720	0.716	0.709	0.434	0.376	0.170
Correlation for implicit instances	0.806	0.760	0.738	0.761	0.756	0.553	0.499	0.242

TABLE 3 – Pearson correlation for the 15 pairs considered and measures from section 3, in decreasing order.

Table 3 shows that mutual information measures are well correlated with human annotations, and that our W_combined seems useful too. We also observed results on each relation separately, although one should be careful drawing conclusions from these results since the correlations are then computed on 5 points only. These results (not shown here) show a lot of variation between relations. The U_do measure, designed for causal relations, does indeed produce good results for these relations, but does not generalize well to our other chosen relations.

Also, local PMI seems to work very well on narration and causal relations. This needs to be confirmed with more verb pairs.

We conclude that the best three measures are : normalized PMI, specificity, and W_combined. The last two assign their maximal value to several pairs, so we used them in a lexicographical ordering to sort all associated pairs, using normalized PMI to break ties.

Verb pair	Translation	Relation
abandonner / *mener*	abandon / lead	background
ne pas s'arrêter / *rouler*	not stop / drive	narration
donner satisfaction sur / *réélire*	give satisfaction concerning / re-elect	continuation
emporter / *ne pas cesser*	take away / not stop	summary
emprunter / *assurer*	borrow / insure	cause
ne pas manquer / *prolonger*	not miss / prolong	detachment
ratifier / *trembler*	ratify / tremble	background
avoir honte / *faire pitié*	be ashamed / cause pity	cause
avoir droit / *cotiser pour*	be entitled / contribute to	temploc
ne pas représenter / *stéréotyper*	not represent / stereotype	temploc

TABLE 4 – Ten best triples in the database.

Table 4 shows the best triples with our lexicographical ranking.

4.2 Extrinsic evaluation

In order to evaluate the performance of our resource relative to its main intended application—predicting rhetorical relations in text, we intend to use our association measures as additional features to an inductive prediction model. Whether this evaluation produces results depends on the proportion of cases in which this information could help and on the coverage of our resource with respect to these cases. We used the Annodis corpus (Afantenos et al., 2012), a set of French texts annotated with rhetorical relations, for our study.

To improve existing models, a significant number of the predictions to be made must involve a verb pair for which we have information in the resource. A first indication of its usefulness is also that the verb pair appears most frequently with the relation group to which the annotation belongs, for instance the fact that two verbs are related with a causal relation whenever we want to predict an explanation. This is interesting only in the absence of an explicit marking of the target relation, i.e for implicit relations.

Beyond that, it should be interesting to use all the available information about other semantic relations too : for instance a potential causal link between two events could indicate the relevance of a temporal link for the prediction of a relation. We relied again on the Lexconn marker database. As an approximation we considered that a relation between two discourse units is explicit when a Lexconn marker is present in any of the two segments, and one of the potential senses of the marker is the annotated relation. This may overestimate the number of explicit instances but ensures that all implicit instances are indeed implicit (assuming a good enough coverage of the marker resource). The Annodis corpus lists rhetorical relations between elementary discourse units (EDUs), typically clauses, and complex discourse units (sets of EDUs) ; as a simplification we only consider EDUs, since the question of what is a main verb of a complex unit is difficult to answer. This is a relatively small corpus, as it includes about 2000 instances of relations between elementary discourse units.

Table 5 present results for coverage, for the main relations in the annotated corpus. Note that only a small part of the set of relations between EDUs is considered when we restrict instances to both EDUs with verbs (about 20% of the whole). It turns out that a lot of EDUs in Annodis are short segments (incises, detached segments, ...).

	global	narration	cause	contrast	elab.	cont.	BG	other
Annodis pairs	427	73	67	41	96	92	24	16
Annodis pairs $\in V^2R$	68.9	71.2	70.8	78.0	68.3	61.9	74.1	62.5
Annodis triples $\in V^2R$	26.5	34.2	50.0	70.7	0.0	20.6	11.1	0.0
Implicit Annodis pairs	83.4	71.2	79.2	36.6	99.0	94.8	88.9	100.0
Implicit Annodis pairs $\in V^2R$ (any relation)	56.9	52.1	54.2	31.7	67.3	58.8	66.7	62.5
Implicit Annodis triples $\in V^2R$ (with correct relation)	17.7	24.7	40.3	31.7	0.0	19.6	11.1	0.0

TABLE 5 – Coverage of verb pairs in V^2R with respect to EDU pairs in the Annodis corpus containing two verbs. Except for the first line, all numbers are percentages. Pair = verb pairs in the EDUs linked by a rhetorical relation R, Triple=verb pair associated with a relation R in V^2R, BG = Background, cont.=continuation, elab.=elaboration.

Our table includes : the proportion of verb pairs found in Annodis EDUs that appear in V^2R, the proportion of triples from Annodis that appear in V^2R (with the correct relation), and the restriction of these proportions to implicit contexts in Annodis. Except for a few exceptions due to lemmatisation errors, all verbs in Annodis are in V^2R in at least one pair, and we can see that the pairs in V^2R cover most of the pairs appearing in Annodis (almost 70% globally and between 60 and 80% depending on the relation), and a little less of implicit cases (around 55% on average). We note that a high proportion of the implicit cases contains verb pairs that have been collected in a marked context, even for rarely marked relations like elaboration or continuation—contexts with these relations are the majority in Annodis. Furthermore more than half of these contexts are associated with the right relation in V^2R. Thus the hypothesis of the partial redundancy of connectors appears useful when isolating verbal associations relevant for discourse from a large corpus. We also looked at semantic neighbors of the verbs in V^2R but this did not increase coverage significantly.

A good test of the predictive power of the semantic information we gathered is also to include the association measures as additional features to a predictive model, to improve classically low results on implicit discourse relations. The only available discursive corpus in French, Annodis, is small, and as shown above only about 400 instances have a verb in both related EDUs. We trained and tested a maximum entropy model with and without the association measures as features, on top of features presented in Muller et al. (2012), who trained a relation model on the same corpus. We did a 10-fold cross-validation on the 400 instance subset as evaluation, and did not find a significant difference between the two set-ups (F1 score was in the range .40–.42, similar to the cited paper), which is unsurprising

given the size of the subset. We plan to evaluate our method relative to discourse parsing by building an English resource like V^2R ; we will then be able to use the much larger PDTB corpus (10 times as large as Annodis) as a source of implicit discourse relations. This should prove a much more telling evaluation of the usefulness of association measures in predicting implicit discourse relations.

5 Related work

There are two different groups of related work. The first group aims to alleviate the lack of annotated data for discourse parsing by using a weakly supervised approach, exploiting the presence of discourse connectors in a large non-annotated corpus. Each pair of elementary discourse units is automatically annotated with the discourse relation triggered by the presence of the connector (connectors are often filtered for non-discursive uses). Those connectors are afterwards eliminated from the corpus so that the model trained on this dataset will not be informed by the presence of those connectors. The pioneering article in this group is Marcu and Echihabi (2002). Such learning methods with such "artificial data" obtain low scores, barely above chance as shown in Sporleder and Lascarides (2008). Braud and Denis (2013) observe that the performance of a classifier for the prediction of implicit relations is much lower when using "artificial" data than on "natural" data (implicit relations annotated by a human being). They propose a method which exploits these two different kinds of datasets together in various mixtures and on the level of the prediction algorithm, obtaining thus a significant improvement on the Annodis corpus. Our approach is different and complementary ; we isolate the semantic relations between pairs of verbs. We can use that as a feature on discourse units for discourse parsing but it has other uses as well.

A second group aims at identifying discourse relations (implicit or not) by focusing on the use of fine-grained lexical relations as another feature during the training phase. Most of this work focuses mainly on the use of lexical relations between two verbs. Chklovski and Pantel (2004), for example, rely on specific patterns constructed manually for each semantic relation between (*similarity, strength, antonymy, enablement* and *temporal happens-before*). They use the web as a corpus in order to estimate the PMI between a pattern and a pair of verbs (a precise measurement cannot be achieved over the web since the probability of a pattern is not precisely known over all the web). A threshold on the value of the PMI (manually fixed) permits thus to determine the pairs of verbs that are related to the relation denoted by the pattern. In the same spirit, Kozareva (2012) is using a weakly supervised approach for the extraction of pairs of verbs that are potentially implied in a *cause-effect* relation. Her method consists in using patterns applied to the web in order to extract pairs and generate new seeds. Do et al. (2011) focus on causal relations and take into account not only verbs but also event denoting nouns. According to this paper, an event is denoted by a predicate with a specific number of arguments and thus the association of the events is the sum of the association between predicates, between predicates and arguments and between arguments. Their association measures are based on PMI and are quite complex. Our results show that their measures do not generalize well to association with all discourse relations. Using Gigaword as a corpus and a reimplementation of Lin et al. (2014) they have extracted discourse relations. An inductive logic programming approach is finally used exploiting the interaction between causal pairs and discourse relations in order to extract causal links. Those papers focus on specific relations with the exception of Chklovski and Pantel (2004) who do not present a systematic evaluation of their results. An important difference of our approach is also to consider predicates and their negation as separate entries.

Finally, we mention the approaches which while focusing on the learning of discourse structures, nonetheless enrich their systems with lexical information. Feng and Hirst (2012) have used HILDA (Hernault et al., 2010) adding more features. A specific family of features represents lexical similarity based on the hierarchical distance in VERBNET and WORDNET. In a similar fashion, Wellner et al. (2006) focus on intra-sentential discourse relations adding lexical information on the features based on measures proposed by Lin (1998) calculated on the British National Corpus. Those approaches use thus only information on lexical similarity without semantically typing this link. The impact of this information seems limited. As far as evaluation is concerned, our method is similar to that followed in Tremper and Frank (2013) for implication relations combining in and out of context evaluation for verbal associations. Their inter-annotator agreement is similar to ours (0.42-0.44 of Kappa) with very different choices : the anno-

tators were supposed to discriminate verbal links between the different possible sub-cases. The pairs of verbs were identified by the system of Lin and Pantel. These authors also present a classification model among the different types of relationships, assuming that two verbs are semantically related.

6 Conclusions

We have presented a knowledge base of triples involving pairs of verbs associated with semantic or discourse relations. We extracted these triples from the large French corpus, frWaC, using discourse connectors as markers of relations between two adjacent clauses containing verbs. We investigated several measures to give the strength of association of a pair of verbs with a relation. We used manual annotations to evaluate our method and select the best measures, and we also studied the coverage of our resource on the discourse annotated corpus Annodis. Our positive results show our resource has the potential to help discourse analysis as well as other semantically oriented tasks.

References

Stergos Afantenos, Nicholas Asher, Farah Benamara, Myriam Bras, Cecile Fabre, Mai Ho-Dac, Anne Le Draoulec, Philippe Muller, Marie-Paul Pery-Woodley, Laurent Prevot, Josette Rebeyrolles, Ludovic Tanguy, Marianne Vergez-Couret, and Laure Vieu. 2012. An empirical resource for discovering cognitive principles of discourse organisation : the ANNODIS corpus. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Nicholas Asher and Alex Lascarides. 2003. *Logics of Conversation*. Studies in Natural Language Processing. Cambridge University Press, Cambridge, UK.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of the COLING-ACL*, Montreal, Canada.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web : a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3) :209–226.

Chloé Braud and Pascal Denis. 2013. Identification automatique des relations discursives "implicites" à partir de données annotées et de corpus bruts. In *TALN - 20ème conférence du Traitement Automatique du Langage Naturel 2013*, volume 1, pages 104–117, Sables d'Olonne, France, June.

Marie Candito, Benoît Crabbé, and Pascal Denis. 2010. Statistical french dependency parsing : Treebank conversion and first results. In *LREC*.

Jean Carletta. 1996. Assessing agreement on classification tasks : the kappa statistic. *Computational linguistics*, 22(2) :249–254.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised Learning of Narrative Event Chains. In *Proceedings of ACL-08 : HLT*, pages 789–797, Columbus, Ohio, June. Association for Computational Linguistics, Morristown, NJ, USA.

Timothy Chklovski and Patrick Pantel. 2004. Verbocean : Mining the web for fine-grained semantic verb relations. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 33–40, Barcelona, Spain, July. Association for Computational Linguistics.

P. Denis and B. Sagot. 2012. Coupling an annotated corpus and a lexicon for state-of-the-art pos tagging. *Language Resources and Evaluation*, (46) :721–736.

Quang Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Stefan Evert. 2005. *The statistics of word cooccurrences*. Ph.D. thesis, Stuttgart University.

C. Felbaum. 1998. *Wordnet, an Electronic Lexical Database for English*. Cambridge : MIT Press.

Vanessa Wei Feng and Graeme Hirst. 2012. Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 60–68, Jeju Island, Korea, July. Association for Computational Linguistics.

G. Grefenstette. 1994. *Explorations in automatic thesaurus discovery*. Springer.

Chikara Hashimoto, Kentaro Torisawa, Kow Kuroda, Stijn De Saeger, Masaki Murata, and Jun'ichi Kazama. 2009. Large-scale verb entailment acquisition from the Web. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1172–1181, Singapore, August. Association for Computational Linguistics.

Hugo Hernault, Helmut Prendinger, David A. duVerle, and Mitsuru Ishizuka. 2010. HILDA : A Discourse Parser Using Support Vector Machine Classification. *Dialogue and Discourse*, 1(3) :1–33.

Zornitsa Kozareva. 2012. Cause-effect relation learning. In *Workshop Proceedings of TextGraphs-7 : Graph-based Methods for Natural Language Processing*, pages 39–43, Jeju, Republic of Korea, July. Association for Computational Linguistics.

Dekang Lin and Patrick Pantel. 2002. Concept discovery from text. In *Proceedings of Coling 2002*, pages 1–7. Association for Computational Linguistics.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, 20(2) :151–184.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th ACL and 17th COLING joint conference*, volume 2, pages 768–774, Montreal.

Maofu Liu, Wenjie Li, Mingli Wu, and Qin Lu. 2007. Extractive summarization based on event term clustering. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 185–188, Prague, Czech Republic, June. Association for Computational Linguistics.

Daniel Marcu and Abdessamad Echihabi. 2002. An Unsupervised Approach to Recognizing Discourse Relations. In *Proceedings of ACL*, pages 368–375.

Seyed Abolghasem Mirroshandel, Alexis Nasr, and Benoît Sagot. 2013. Enforcing subcategorization constraints in a parser using sub-parses recombining. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 239–247, Atlanta, Georgia, June. Association for Computational Linguistics.

Philippe Muller, Stergos Afantenos, Pascal Denis, and Nicholas Asher. 2012. Constrained decoding for text-level discourse parsing. In *Proceedings of COLING 2012*, pages 1883–1900, Mumbai, India, December. The COLING 2012 Organizing Committee.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser : A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2) :95–135.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie L. Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of LREC 2008*.

Charlotte Roze, Laurence Danlos, and Philippe Muller. 2012. Lexconn : A french lexicon of discourse connectives. *Discours*, (10).

Benoît Sagot. 2010. The lefff, a freely available and large-coverage morphological and syntactic lexicon for french. In *7th international conference on Language Resources and Evaluation (LREC 2010)*.

Caroline Sporleder and Alex Lascarides. 2008. Using Automatically Labelled Examples to Classify Rhetorical Relations : An Assessment. *Natural Language Engineering*, 14(3) :369–416, July.

Galina Tremper and Anette Frank. 2013. A discriminative analysis of fine-grained semantic relations including presupposition : Annotation and classification. *Dialogue & Discourse*, 4(2) :282–322.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1 : Tempeval-3 : Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2 : Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

K. Van Den Eynde and P. Mertens, 2010. *Le dictionnaire de valence : Dicovalence*. Leuven : Université de Leuven. [http ://bach. arts. kuleuven. be/dicovalence/].

Ben Wellner, James Pustejovsky, Catherine Havasi, Anna Rumshisky, and Roser Saurí. 2006. Classification of discourse coherence relations : an exploratory study using multiple knowledge sources. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, SigDIAL '06, pages 117–125, Stroudsburg, PA, USA. Association for Computational Linguistics.

HARPY: Hypernyms and Alignment of Relational Paraphrases

Adam Grycner
Max-Planck Institute for Informatics
Campus E1.4, 66123
Saarbrücken, Germany
agrycner@mpi-inf.mpg.de

Gerhard Weikum
Max-Planck Institute for Informatics
Campus E1.4, 66123
Saarbrücken, Germany
weikum@mpi-inf.mpg.de

Abstract

Collections of relational paraphrases have been automatically constructed from large text corpora, as a WordNet counterpart for the realm of binary predicates and their surface forms. However, these resources fall short in their coverage of hypernymy links (subsumptions) among the synsets of phrases. This paper closes this gap by computing a high-quality alignment between the relational phrases of the Patty taxonomy, one of the largest collections of this kind, and the verb senses of WordNet. To this end, we devise judicious features and develop a graph-based alignment algorithm by adapting and extending the SimRank random-walk method. The resulting taxonomy of relational phrases and verb senses, coined HARPY, contains 20,812 synsets organized into a *Directed Acyclic Graph (DAG)* with 616,792 hypernymy links. Our empirical assessment, indicates that the alignment links between Patty and WordNet have high accuracy, with *Mean Reciprocal Rank (MRR)* score 0.7 and *Normalized Discounted Cumulative Gain (NDCG)* score 0.73. As an additional extrinsic value, HARPY provides fine-grained lexical types for the arguments of verb senses in WordNet.

1 Introduction

Motivation: This paper addresses the task of discovering and organizing paraphrases of relations between entities (Lin and Pantel, 2001; Fader et al., 2011; Nakashole et al., 2012; Moro and Navigli, 2012; Alfonseca et al., 2013). This task involves understanding that the phrases "travels to", "visits" and "on her tour through" (relating a person and a country) are synonymous and that "leader of" and "works with" (relating a person and an organization) are in a hypernymy relation: the former is subsumed by the latter. This kind of lexical knowledge can be harnessed for advanced tasks like question answering (Fader et al., 2013), search over web tables (Gupta et al., 2014), or event mining over news (Alfonseca et al., 2013).

Work along these lines has developed large repositories of relational paraphrases, most notably, the collections ReVerb (Fader et al., 2011), Patty (Nakashole et al., 2012), and WiSeNet (Moro and Navigli, 2012). The largest of these, Patty, contains ca. 350,000 synsets of phrases, each annotated with ontological types of their two arguments (e.g., person $\times$ country, or politician $\times$ political_party). However, the subsumption hierarchy of Patty is very sparse. It contains only 8,000 hypernymy links between phrases, and the entire taxonomy is kind of fragmented into a many-rooted DAG (directed acyclic graph). Moreover, the synsets are rather noisy in the long tail with low confidence. WiSeNet, an alternative resource, has ca. 40,000 synsets and no hypernymy links.

WordNet (Fellbaum, 1998), on the other hand, is a very rich resource on synonymy and hypernymy. However, its coverage of binary relations (as opposed to unary predicates, mostly nouns) is restricted to (mostly) single-word verbs. WordNet has ca. 13,767 verb synsets, organized into a hierarchy with 13,239 hypernymy links. Unlike Patty, though, WordNet does not associate verb senses with a lexical type signature for the subject and object arguments of a verb, and it is sparse in multi-word phrases.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2195–2204, Dublin, Ireland, August 23-29 2014.

Resources like VerbNet (Kipper et al., 2008) or FrameNet (Baker et al., 1998) aim to overcome these deficiencies, but are much smaller.

Goal and Approach: In this paper, our goal is to overcome the limitations of resources like Patty and WordNet. We want to reconcile the wealth of Patty's multi-word paraphrases with lexical typing, on one hand, and the clean hypernymy organization of WordNet verbs, on the other hand. To this end, we compute an alignment between the phrase synsets that Patty provides with the verb senses of WordNet. This has mutual benefits: 1) we enhance many Patty phrases with the clean hypernyms of WordNet, this way augmenting the subsumption hierarchy, and 2) we extend WordNet verb senses with the lexical type signatures derived from Patty. Our approach uses a variety of features from both of the two aligned resources, as well as further auxiliary sources. Algorithmically, we build on an advanced notion of random walks over graphs, known as SimRank (Jeh and Widom, 2002).

Contributions: Our method is able to construct a high-quality taxonomy of relational paraphrases, coined *HARPY*, that combines the richness of Patty with the clean hierarchy of WordNet. The algorithm for computing the alignment is efficient and robust. One can think of the alignment as a way of sense-disambiguating Patty phrases by mapping them to WordNet. HARPY links 20,812 of the Patty phrases to WordNet. Conversely, 4,789 out of 13,767 WordNet verb senses are enriched with information from Patty. We evaluate the quality of HARPY by extensive sampling with human assessment. We also demonstrate its benefit by the extrinsic use-case of annotating WordNet verb senses with lexical type signatures. All experimental data and the HARPY resource will be available on a public web site.

2 Related Work

With the proliferation of knowledge bases, like Freebase (Google Knowledge Graph), DBpedia, YAGO, or ConceptNet, there is a wealth of resources about entities and semantic classes (i.e., unary predicates and their instances). In contrast, the systematic compilation of paraphrases for relations (i.e., binary predicates) has received much less attention. Some of the knowledge-base projects, especially those that center on Open Information Extraction, make intensive use of surface patterns (e.g., verbal phrases) that indicate relations (e.g., (Carlson et al., 2010; Fader et al., 2011; Mausam et al., 2012; Speer and Havasi, 2012; Wu et al., 2012)); however, they do not organize these patterns into a WordNet-style taxonomy.

Prior work towards such taxonomies go back to the projects DIRT (Lin and Pantel, 2001), VerbOcean (Chklovski and Pantel, 2004), and VerbNet (Kipper et al., 2008). However, the resulting resources were mostly restricted to single verbs. ReVerb (Fader et al., 2011) extended these approaches by automatically mining entire phrases from Web contents, but still with focus on verbal structures. Patty (Nakashole et al., 2012) used sequence mining algorithms for gathering a general class of relational phrases, organizing them into synsets, and inferring lexical type signatures. WiseNet (Moro and Navigli, 2012) harnessed phrases from Wikipedia articles and clustered them into synsets of relational phrases. All of these works are fairly limited in their coverage of subsumptions (hypernymy) between relational phrases.

There is ample work on computing alignments among different kinds of lexical thesauri, dictionaries, taxonomies, ontologies, and other forms of linguistic or semantic resources. Prominent cases along these lines include the alignments between FrameNet and WordNet (Ferrández et al., 2010), VerbNet and PropBank (Palmer, 2009), Wikionary and WordNet (Meyer and Gurevych, 2012), and across multilingual WordNets and/or Wikipedia editions (e.g., (de Melo and Weikum, 2009; Navigli and Ponzetto, 2012)). For aligning ontologies based on OWL and RDF logics, there is a series of annual benchmark competitions (Grau et al., 2013). Most approaches are based on relatedness measures and context similarities between words or concepts and their neighborhoods in the respective resources (e.g., (Banerjee and Pedersen, 2003; Budanitsky and Hirst, 2006; Gabrilovich and Markovitch, 2007)). Algorithmically, this translates into a nearest-neighbor (most-similar) assignment between entries of different resources. More sophisticated methods use similarities merely to assign weights to relatedness edges in a graph, and then employ random walks on such a graph (e.g., (Pilehvar et al., 2013)). The prevalent method of this kind uses Personalized Page Rank (Haveliwala, 2002)), computing stationary probabilities for reaching nodes in one resource when starting random walks on a given node of the other resources (with randomized restarts).

Computing alignments between resources can sometimes be viewed as a task of disambiguation words or concepts in one resource by mapping them to the other resource (e.g., mapping Wiktionary entries onto WordNet senses). Thus, the huge body of work on word sense disambiguation (WSD) is relevant, too. Methodologically, this research also relies, to a large extent, on relatedness/similarity measures and random walks on appropriately constructed graphs. See (Navigli, 2009) for an extensive survey.

There is remotely related work on several other tasks in computational linguistics and text mining. These include semantic relatedness between concepts or words (e.g., (Gabrilovich and Markovitch, 2007; Pilehvar et al., 2013)), type inference for the arguments of a phrase (e.g., (Kozareva and Hovy, 2010; Nakashole et al., 2013)), and entailment among verbs (e.g., (Hashimoto et al., 2009)). The SemEval-2010 task on classification of semantic relations (Hendrickx et al., 2010) addressed the problem of predicting the relation for a given sentence and pair of nominals, but was limited to a small prespecified set of relations.

3 Constructing a Candidate Alignment Graph

The general idea of the main algorithm is to align phrase synsets from the Patty taxonomy with verb synsets in WordNet. To this end, we first construct a directed *candidate alignment graph (CAG)*. Section 4 will then discuss the actual alignment algorithm.

Vertices of the CAG represent

- synsets of *relational phrases* in Patty, or phrases for short,
- *verb senses* from WordNet, verbs for short,
- *features* of either phrases or verbs.

Edges of the CAG correspond to relations between phrases, verbs, and features. We consider three types of relations here: similarity, hypernymy, and vertex-features. Edges are weighted (see below).

Vertex Types: There are 6 kinds of vertices in the CAG. Since we aim to connect Patty *phrases* with WordNet *verbs*, these two are the main kinds of vertices. Additionally, the graph contains feature vertices representing *noun senses* from WordNet (nouns for short), *surface verbs* as occurring in sample texts, *sentence frames* from WordNet, and specifically derived *phrase-verb vertices* connecting phrases and verbs. The latter are constructed by combining each phrase with its top-10 most similar verb senses. To this end, we retrieve all verb synsets from WordNet and rank the verb synsets by the cosine similarity between the support sentences that Patty provides for its phrases (i.e., sentences from Wikipedia that contain instances of a phrase) and the usage examples in WordNet glosses. The resulting vertices are labeled by the combination of phrase id and verb-sense id. Having these combinations as vertices, rather than simply connecting phrases and verbs via edges, leads to a CAG structure that is better suited for our random walk algorithms (see Section 4). Table 1 gives examples for the 6 vertex types.

Relational Phrase	Verb Sense	Noun Sense	Surface Verb	Sentence Frame	Phrase-Verb Pair
[person] succeeded [person]	succeed2#verb	king1#noun	succeed	Somebody ----s somebody	(phrase_1, verb_sense_2)
[musician] played jazz with [musician]	play3#verb	music1#noun	play	Somebody ----s something	(phrase_2, verb_sense_3)

Table 1: Examples of vertex types

Edge Types: Edges in the graph represent 3 different types of relationships between vertices:

- For all relational phrases, all verb senses from WordNet and also all noun senses (as feature vertices), we capture their *hypernymy relations* as edges.
- We connect phrase-verb vertices with their constituents, phrase vertices and verb vertices, by *similarity edges*, with weights derived from the similarity computation.
- The remaining edges connect phrases or verbs with their respective feature vertices. There are 6 kinds of such *vertex-feature edges*, explained next.

Verb Features: The following features are associated with verb senses. A *lemma edge* connects a verb sense with one or more surface-verb vertices, as given in WordNet glosses. A *domain edge* edge connects a verb sense with noun senses that describe the usage domain of the verb (e.g. literature, politics). This information is retrieved from WordNet and the WordNet Domains project (Bentivogli et al., 2004). While the latter does not provide sense-disambiguated information, we need to add a mechanism which maps domain information to its WordNet noun sense counterpart. Therefore, we map domain surface nouns to their most frequent senses.

In addition, we harness the WordNet links of type *derivationally related form* to construct further edges between verb senses and noun-sense features in our CAG. The last type of edges for verb-sense features are *sentence frame edges*, between verb vertices and feature vertices of type sentence frame. WordNet for each verb sense provides information about its sentence frames. There are defined 35 possible sentence frames.

Phrase Features: Relational phrases are associated with the following features. A *verb-in-phrase edge* connects a phrase with a surface verb whenever the phrase contains the verb after lemmatization. Analogously to the domain edges for verb senses, we introduce *Wikipedia-category edges* between relational phrases and noun senses. Patty provides us with Wikipedia articles where instances of a phrase occur. We consider all Wikipedia categories of such an article as a source for related noun senses. We use ontological types of the articles and the categories and their mappings to Wordnet provided by the YAGO project (Suchanek et al., 2007). Finally, we also introduce *sentence-frame edges* between relational phrases and sentence-frame feature vertices. To avoid polluting the CAG with overly noisy connections, we apply specific tests. First, we check if the lexical argument types of a phrase and a frame are compatible (e.g., musician is compatible with person, but not with location). Second, we compare characteristic prepositions in the phrase and the frame. We create and edge only if these additional tests are affirmative.

Examples of vertices connected by the different edge types with verb vertices and phrase vertices are shown in Table 2 and 3, respectively.

Hypernymy	Similarity	Lemma	Domain	Derivationally Related Form	Sentence Frame
replace2#verb	(phrase_1, verb_sense_2)	"succeed", "come after"	politics1#noun	successor1#noun	Somebody `----s` somebody

Table 2: Vetices connected by different edges with vertex "succeed2#verb" of type verb.

Hypernymy	Similarity	Verbs in phrase	Wikipedia Category	Sentence Frame
[person] replaced [person]	(phrase_1, verb_sense_2)	"succeed"	politician1#noun	Somebody `----s` somebody

Table 3: Vetices connected by different edges with vertex "[person] succeeded [person]" of type phrase.

Edge Weights: All edges in the graph are weighted. The weights are derived from frequency counts of features and/or similarity scores, or are simply set to 1 for binary cases (e.g., hypernymy edges). Lemma edges between verb senses and surface verbs vertices are weighted in proportion to the frequency count of a verb sense, as given by WordNet. Wikipedia-category edges have weights based on the number of occurrences of a relational phrase in Wikipedia articles and the frequencies of categories. Similarity edges have weights set according to the cosine similarity between examples of a verb sense and examples of a relational phrase.

Finally, we normalize all weights in the graph by requiring that the sum of weights of the incoming edges is equal to 1 for every vertex. For the verb and phrase vertices, we perform an additional normalization so that each kind of edge has the same impact in terms of the total edge weight per edge kind.

The above procedure leads to a CAG with 238,437 vertices and 4,776,116 edges. Figure 1 shows an excerpt for illustration.

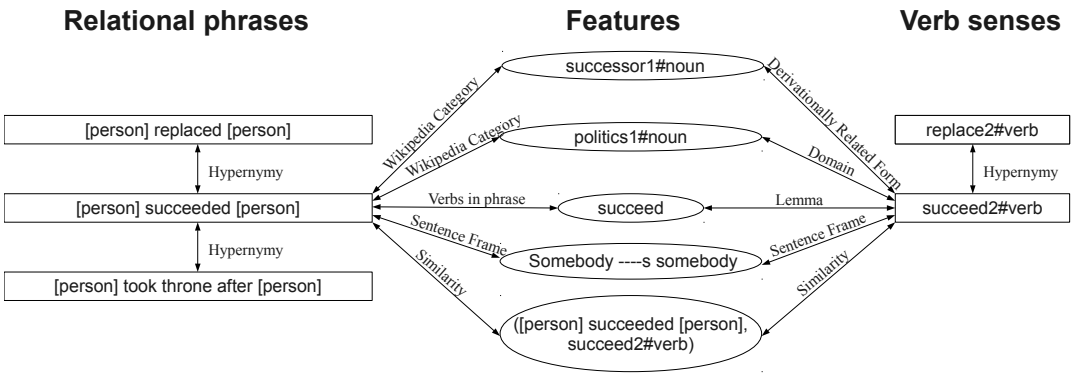


Figure 1: Excerpt from Candidate Alignment Graph

4 Alignment Algorithm

Our algorithm runs on the directed *candidate alignment graph (CAG)*. Intuitively, it aims to find "strong paths" between relational-phrase vertices and verb-sense vertices. We use random-walk methods to this end. For each relational phrase, we compute scores and a ranked list of verb senses to which the phrase likely corresponds. The top-ranked verb would ideally be the desired alignment.

SimRank: We employ the SimRank algorithm (Jeh and Widom, 2002), an advanced form of random walks. SimRank computes similarity scores between a pair of vertices in a weighted graph, based on the neighborhoods of the two vertices. The definition, formally given in Equation 1, is recursive: two vertices are similar if their neighborhoods are similar. In the standard SimRank equation, $I_i(a)$ represents the i^{th} (incoming) neighbor of vertex a, and C is a constant dampening factor.

$$s(a,b) = \frac{C}{|I(a)|\,|I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s\left(I_i(a), I_j(b)\right) \tag{1}$$

SimRank helps capturing long-distance dependencies between vertices in a graph. This would not be achieved by simpler similarity measures of context vectors. Note that SimRank is quite different from (Personalized) PageRank methods; SimRank can be seen as a random walk over pairs of nodes, not over individual nodes. During the CAG construction, we tried to keep the path lengths between phrase vertices and verb vertices uniform for all kinds of feature vertices, to avoid biasing the influence of specific features. Since the SimRank similarity is based on two random walks meeting, the method works best when all paths between source-target node pairs have even length. With this property SimRank produces better results; we introduced explicit phrase-verb vertices for this reason.

SimRank with Fingerprints: Unfortunately, SimRank has very high computational complexity: the run-time of a straightforward implementation is O(Kn^4), where n is the number of vertices in the graph and K is the number of iterations in an iterative fixpoint computation (in the style of the Jacobi method). However, there are much faster approximations of SimRank. We use a variant known as *SimRank with fingerprints* (Fogaras and Rácz, 2005) To approximate the SimRank score for two vertices, this method computes the *expected first meeting time* for two random walks originating from the two vertices (with randomized restarts). To this end, the method precomputes a fingerprint for each vertex a: a data structure holding the visiting probabilities of vertices for standard random walks originating in a. A fast implementation actually runs random walks a specified number of times, to estimate the visiting probabilities. For two vertices a and b, the expected number of hops until their random walks meet in a common vertex is then efficiently computed from the fingerprints of a and b. Moreover, this method allows computing the SimRank score for a pair of vertices on demand, only for vertex pairs of interest, rather than having to compute all $O(n^2)$ scores.

The original SimRank method works with unweighted graphs. In our setting, we modify transition probabilities according to edge weights. Our extended SimRank variant is equivalent to Equation 2,

where $W(a, b)$ denotes the weight of the edge between a and b. This equation is similar to the weighted variant of (Antonellis et al., 2007).

$$s_w(a, b) = C * \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} W(a, I_i(a)) * W(b, I_j(b)) * s_w(I_i(a), I_j(b)) \quad (2)$$

Unlike the original SimRank method, we also incorporate random jumps in the underlying random-walk model. Each vertex has a different random jump probability, explained next.

Random Jumps: The original SimRank definition favors vertices with smaller neighborhoods. To avoid this bias, we introduce a form of smoothing on the graph. Whenever a phrase vertex or verb vertex lacks some of the feature types that other vertices may have, we introduce an option for random jumps from the given vertex to any other vertex in the graph. For each missing kind of feature (e.g., domain feature or sentence-frame feature), we assign a probability mass of ϵ, a small constant, for a random jump. So if several features are missing, there is an accumulated probability for a jump. The target of a random jump is always chosen with uniform distribution. A final normalization of edge weights (with linear adjustment) ensures that the possible transitions from a vertex form a proper probability distribution. he method works also without smoothing (i.e., setting the constant to 0), but the results tend to be worse. The results are not very sensitive to the exact choice of the random-jump parameter.

Filtering and Candidate Pruning: The target of our alignment is the WordNet verb hierarchy, but not all relational phrases can be mapped into this target space. Therefore, we restrict ourselves to a subset of relational phrases that contain exactly one verb. This eliminates noun phrases (e.g. "father of") and phrases that contain multiple verbs (e.g. "succeed and died", "succeeded in persuading"). Noun phrases should be aligned to the WordNet noun hierarchy and it should be treated as a different task (using e.g. state-of-the-art work (Ponzetto and Navigli, 2010)). Multi-verb phrases often pose semantic difficulties. Note that the verbs in these phrases are always transitive verbs, as Patty is derived from subject-phrase-object structures in large corpora. We also used the cardinalities of the support sentences in Patty for pruning the noisy tail of phrases, by dropping all phrases that have only a single instance.

To avoid computing SimRank scores for every pair of vertices, we prune the search space as follows. We consider only pairs of relational phrases and verb senses which contain the same surface verb (with lemmatization).

Deriving Hypernymy Links: Once we have alignments between phrases and verbs, we derive hypernymy relations *among phrases* as follows. Whenever phrases p_1 and p_2 are aligned with verb senses v_1 and v_2, respectively, and v_1 is a direct or transitive hypernym of v_2, we infer that p_1 is a hypernym of p_2. We consider transitive hypernyms because not every WordNet verb sense has a phrase aligned with it; without transitivity we would obtain a very sparse hierarchy. By the acyclicity of the WordNet hypernymy structure, the process yields a proper DAG. However, the output contains redundant links (direct ones and transitive ones connecting the same pair of phrases); these are subsequently eliminated by a transitive reduction algorithm (Aho et al., 1972).

5 Evaluation

We evaluated the quality of the HARPY alignments by manual assessment of a large sample set, and compared it against several alternative methods.

Baselines: We compared our SimRank-based method against the following baselines, each given the same feature set:

- *Cosine Similarity:* for each relational phrase and verb sense, we create a contextual vector (in the spirit of distributional semantics) consisting of the features described in Section 3, with tf-idf-based weights (Manning et al., 2008). The alignment ranking is computed by the cosine similarity of tf-idf-weighted contextual vectors.
- *Modified Adsorption (MAD):* a label propagation algorithm (Talukdar and Crammer, 2009) run on the candidate alignment graph. In our setting, each relational phrase is a label. Initially, only the respective phrase vertices have this label. The algorithm propagates labels to other vertices, based on

the graph's edge weights. The top-k results for the alignment of a phrase are the verb senses with the highest probability for the phrase label. We use the Junto Label Propagation Toolkit [1].

- *Personalized PageRank (PPR):* a method for random walks with random jumps back to the start vertex (Haveliwala, 2002). For each phrase, a separate PPR is performed. The ranking of verb senses is produced by the visiting probabilities according to the PPR scores.
- *Most Frequent Sense (MSF):* For each phrase, we consider only verb senses that contain the same surface verb (with lemmatization), and rank them by the WordNet frequency information.

Assessment: We retrieved a random subset of 261 relational phrases considered for alignment, and showed the results of the different alignment methods to two human judges. For each relational phrase, we displayed its textual form, list of usage examples, and the top-5 ranked list of verb senses computed by each method under comparison. Each verb sense was enriched with information about its lemmas, its gloss, and examples. The evaluators were asked to identify the verb sense that is semantically equivalent to the given relational phrase (including the option of saying "none").

Quality Measures: As all methods compute a ranked list of verb senses for a given phrase where exactly one list item is correct, we use quality measures geared for such rankings: Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain ($NDCG$). In addition, we report on the precision for top-k results, for small k (1, 3, or 5). Here, a top-k result is considered good if the correct verb senses appears among the top-k alignments, for a given phrase.

Results: The results are shown in Table 4. Our method outperforms all baselines. Among the competitors, MFS shows the best performance. This is not so surprising; MFS is rarely outperformed in word sense disambiguation (McCarthy et al., 2004; Navigli and Lapata, 2010). Our gains over MFS are remarkable. In total, HARPY aligned 20,812 phrases to 4,789 verb senses, and also obtained 616,792 hypernymy links between phrases.

The evaluation process led to high inter-judge agreement, with Cohen's Kappa around 0.678. The number of samples, 261, was large enough for statistical significance: we performed a paired t-test for MRR, $NDCG$ and $Precision@1$ of the SimRank results against each of the baselines, and obtained p-values below 0.05.

	SimRank	MFS	PPR	MAD	Cosine
MRR	0.698	0.664	0.553	0.463	0.252
NDCG	0.733	0.705	0.584	0.51	0.279
Precision@1	0.571	0.517	0.41	0.318	0.161
Precision@3	0.793	0.778	0.644	0.594	0.307
Precision@5	0.874	0.866	0.736	0.67	0.391

Table 4: Evaluation

Tables 5 and 6 shows example results that HARPY computed. Table 5 has correct outputs. We see that HARPY manages to distinguish between the sport, musical, and theatrical senses of the verb "play". As shown in Table 6, HARPY also produces some spurious results, with various factors contributing to these errors. For example, the phrase "covered on album" was aligned with the first sense of "cover" since there is no musical sense for "cover" in WordNet. Other errors arise from mistakes in the original Patty repository of relational phrases. For example, the travel sense of the verb "head" was aligned with the phrase "head of" because "head of" and "head to" were in the same Patty synset. Yet another cause of problems is the extremely fine granularity of WordNet: even for humans it is often hard to distinguish between love as a state of liking and love as being enamored.

6 Extrinsic Study: Lexical Types for WordNet Verbs

As an extrinsic use-case for the HARPY resource, we studied the task of inferring lexical types for the subject and object arguments of a WordNet verb sense. For a given verb sense, we propagate the type signature of the relational phrase with the highest alignment score.

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

Relational phrase	Verb Sense	WordNet definition
[musician] played with [musician]	play3	play on an instrument
[actor] played [[det]] role in [event]	act3	play a role or part
[person] played hockey for [organization]	play1	participate in games or sport
[person] was shooting [person]	shoot2	kill by firing a missile
[movie] be shot in [city]	film1	make a film or photograph of something
[composition] written by [composer]	compose2	write music
[writer] writing at [organization]	write1	produce a literary work

Table 5: Correct examples

Relational phrase	Verb Sense	WordNet definition
[person] covered on album [artifact]	cover1	provide with a covering or cause to be covered
[person] head of [artifact]	head1	to go or travel towards
[person] becomes convinced that [person]	become1	enter or assume a certain state or condition
[person] is loved by [person]	love1	have a great affection or liking for
[wrestler] wrestled in [organization]	wrestle1	combat to overcome an opposing tendency or force

Table 6: Wrong alignment examples

For comparison, this procedure is performed with the HARPY alignments as well as the alignments by the baseline methods. We showed a uniformly sampled set of 261 results to human judges, who assessed as valid or invalid. Additionally, we had a set of the 100 most-confident results (those derived from the highest alignment scores) assessed in the same manner.

For the uniform samples, the type signature derived from HARPY had a precision of 0.46, whereas the best of the baselines (PPR and Cosine) achieved 0.39. For the top-100 samples, HARPY achieved a precision of 0.81. Table 7 shows some example results, demonstrating the added value beyond WordNet.

Domain	Range	Verb Sense	WordNet definition
country	country	export1	sell or transfer abroad
person	country	head2	be in charge of
organization	organization	own1	have ownership or possession of
person	person	predate1	be earlier in time; go back further
saint	organization	reverence1	regard with feelings of respect and reverence
person	artifact	rush5	run with the ball, in football
organization	person	sustain4	supply with necessities and support
musician	musician	play3	play on an instrument
football_player	athlete	pass20	throw (a ball) to another player
singer	composer	inspire2	supply the inspiration for
ruler	country	suppress1	to put down by force or authority
architect	city	design2	plan something for a specific role or purpose or effect
priest	saint	canonize2	treat as a sacred person
country	country	ally_with1	unite formally; of interest groups or countries
company	organization	deal13	sell
artifact	computer_game	port8	modify (software) for use on a different machine or platform

Table 7: Type inference examples by HARPY

7 Conclusion

HARPY is a new resource that aligns lexically typed multi-word phrases for binary relations with WordNet verb senses. By judiciously devising appropriate features and adapting and extending an advanced random-walk method, SimRank, we achieved high-quality alignments, as shown in our evaluation. This creates added value for both the resource of relational phrases, Patty, and WordNet. Phrases are now organized into a clean hypernymy hierarchy, an important aspect on which the Patty work fell short. WordNet verb senses, on the other hand, are extended by a rich set of paraphrases and also by lexical type signatures inherited from the phrases. We believe that this new resource is a useful asset for computational linguistics. As a future work, we plan to align additional resources like WiseNet (Moro and Navigli, 2012), FrameNet (Baker et al., 1998) or VerbNet (Kipper et al., 2008). The HARPY resource is publicly available at `www.mpi-inf.mpg.de/yago-naga/patty/`.

References

Alfred V. Aho, M. R. Garey, Jeffrey D. Ullman 1972. The Transitive Reduction of a Directed Graph. *SIAM J. Comput.*, 131–137.

Enrique Alfonseca, Daniele Pighin, and Guillermo Garrido. 2013. HEADY: News headline abstraction through event pattern clustering. *ACL (1)*, 1243–1253.

Ioannis Antonellis, Hector Garcia-Molina, and Chi-Chao Chang. 2007. Simrank++: Query rewriting through link analysis of the click graph. *CoRR*, abs/0712.0499.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. *COLING-ACL*, 86–90.

Satanjeev Banerjee and Ted Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. *IJCAI*, 805–810.

Luisa Bentivogli, Pamela Forner, Bernardo Magnini, and Emanuele Pianta. 2004. Revising the WordNet Domains hierarchy: Semantics, coverage and balancing. In *Proceedings of the Workshop on Multilingual Linguistic Ressources*, MLR '04, 101–108, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1): 13–47.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for Never-Ending Language Learning. *AAAI*

Timothy Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the web for fine-grained semantic verb relations. *EMNLP*, 33–40.

Gerard de Melo and Gerhard Weikum. 2009. Towards a universal wordnet by learning from combined evidence. *CIKM*, 513–522.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. *EMNLP*, 1535–1545.

Anthony Fader, Luke S. Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. *ACL (1)*, 1608–1618.

Christiane Fellbaum, George Miller (Editors). 1998. *WordNet An Electronic Lexical Database*. The MIT Press.

Óscar Ferrández, Michael Ellsworth, Rafael Muñoz, and Collin F. Baker. 2010. Aligning FrameNet and WordNet based on semantic neighborhoods. *LREC*.

Dániel Fogaras and Balázs Rácz. 2005. Scaling link-based similarity search. *WWW*, 641–650.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. *IJCAI*, 1606–1611.

Bernardo Cuenca Grau, Zlatan Dragisic, Kai Eckert, Jérôme Euzenat, Alfio Ferrara, Roger Granada, Valentina Ivanova, Ernesto Jiménez-Ruiz, Andreas Oskar Kempf, Patrick Lambrix, Andriy Nikolov, Heiko Paulheim, Dominique Ritze, François Scharffe, Pavel Shvaiko, Cássia Trojahn dos Santos, and Ondrej Zamazal. 2013. Results of the ontology alignment evaluation initiative 2013. *Ontology Matching*, volume 1111 of *CEUR Workshop Proceedings*, 61–100.

Rahul Gupta, Alon Halevy, Xuezhi Wang, Steven Whang, and Fei Wu. 2014. Biperpedia: An ontology for search applications. *Proc. 40th Int'l Conf. on Very Large Data Bases (PVLDB)*. 505–516 .

Chikara Hashimoto, Kentaro Torisawa, Kow Kuroda, Stijn De Saeger, Masaki Murata, and Jun'ichi Kazama. 2009. Large-scale verb entailment acquisition from the web. *EMNLP*, 1172–1181.

Taher H. Haveliwala. 2002. Topic-sensitive PageRank. *WWW*, 517–526.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. *Proceedings of SemEval-2*, Uppsala, Sweden.

Glen Jeh and Jennifer Widom. 2002. SimRank: a measure of structural-context similarity. *KDD*, 538–543.

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2008. A large-scale classification of English verbs. *Language Resources and Evaluation*, 42(1):21–40.

Zornitsa Kozareva and Eduard H. Hovy. 2010. Learning arguments and supertypes of semantic relations using recursive patterns. *ACL*, 1482–1491.

Dekang Lin and Patrick Pantel. 2001. DIRT @SBT@discovery of inference rules from text. *KDD*, 323–328.

Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze 2008. Scoring, Term Weighting, and the Vector Space Model. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, England, 2008, pp. 109–133.

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open language learning for information extraction. *EMNLP-CoNLL*, 523–534.

Diana McCarthy, Rob Koeling, Julie Weeds, and John A. Carroll. 2004. Finding predominant word senses in untagged text. *ACL*, 279–286.

Christian M. Meyer and Iryna Gurevych. 2012. To exhibit is not to loiter: A multilingual, sense-disambiguated Wiktionary for measuring verb similarity. *COLING*, 1763–1780.

Andrea Moro and Roberto Navigli. 2012. WiseNet: building a Wikipedia-based semantic network with ontologized relations. *CIKM*, 1672–1676.

Ndapandula Nakashole, Gerhard Weikum, and Fabian M. Suchanek. 2012. PATTY: A taxonomy of relational patterns with semantic types. *EMNLP-CoNLL*, 1135–1145.

Ndapandula Nakashole, Tomasz Tylenda, and Gerhard Weikum. 2013. Fine-grained semantic typing of emerging entities. *ACL (1)*, 1488–1497.

Roberto Navigli and Mirella Lapata. 2010. An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(4):678–692.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artif. Intell.*, 193:217–250.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2).

Martha. Palmer. 2009. SemLink: Linking PropBank, VerbNet and FrameNet. In *Proceedings of the Generative Lexicon ConferenceGenLex-09*, Pisa, Italy, Sept.

Mohammad Taher Pilehvar, David Jurgens, and Roberto Navigli. 2013. Align, disambiguate and walk: A unified approach for measuring semantic similarity. *ACL (1)*, 1341–1351.

Simone Paolo Ponzetto and Roberto Navigli. 2010. Knowledge-Rich Word Sense Disambiguation Rivaling Supervised Systems. *ACL*, 1522-1531.

Robert Speer and Catherine Havasi. 2012. Representing general relational knowledge in ConceptNet 5. *LREC*, pages 3679–3686.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: a core of semantic knowledge. *WWW*, 697–706.

Partha Pratim Talukdar and Koby Crammer. 2009. New regularized algorithms for transductive learning. *ECML/PKDD (2)*, volume 5782 of *Lecture Notes in Computer Science*, pages 442–457. Springer.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Qili Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. *SIGMOD Conference*, 481–492.

Limitations of MT Quality Estimation Supervised Systems: The Tails Prediction Problem

Erwan Moreau
CNGL and Computational Linguistics Group
Centre for Computing and Language Studies
School of Computer Science and Statistics
Trinity College Dublin
Dublin 2, Ireland
moreaue@cs.tcd.ie

Carl Vogel
Computational Linguistics Group
Centre for Computing and Language Studies
School of Computer Science and Statistics
Trinity College Dublin
Dublin 2, Ireland
vogel@cs.tcd.ie

Abstract

In this paper we address the question of the reliability of the predictions made by MT Quality Estimation (QE) systems. In particular, we show that standard supervised QE systems, usually trained to minimize MAE, make serious mistakes at predicting the quality of the sentences in the tails of the quality range. We describe the problem and propose several experiments to clarify their causes and effects. We use the WMT12 and WMT13 QE Shared Task datasets to prove that our claims hold in general and are not specific to a dataset or a system.

1 Introduction

Machine Translation (MT) Quality Estimation (QE) has become an important subject of study in the past few years (Callison-Burch et al., 2012; Bojar et al., 2013). This follows directly from the erratic quality of MT output in general: although MT is now widely used in professional contexts, it is still prone to many errors; therefore a careful post-editing stage, performed by human experts, is usually needed. In this context, QE can help carrying out this process more efficiently, and more specifically to help in the decision process between the automatic and the manual stages: if a reliable indication of quality is provided for every machine-translated sentence, the human effort can be reduced. For example, a very bad translation is worthless because the translator usually has to spend more time fixing it than she or he would have spent translating the sentence from scratch; thus it makes more sense in such cases to either send the sentence back to an alternative MT system (e.g. trained on a different corpus), or simply leave it untranslated for the translator. Clearly the advantage of using a QE system depends on the reliability of its predictions. If it makes too many errors, then it only confuses the translation workflow; in this case the translators would perform better without it.

The quality of an (automatic) QE system cannot be perfect, but it should be at least controllable. That is, it should be possible to assess the reliability of the predictions made by a system, for instance by estimating the level of confidence of the predictions. Hopefully, QE systems will progress towards this kind of behaviour, but currently the evaluation methods are not entirely satisfactory from this perspective. In particular, after describing our experimental setting in §2, we will observe in §3 that the use of the Mean Absolute Error[1] (MAE) as a global evaluation measure hides huge discrepancies in the distribution of errors among the range of scores. More precisely, supervised systems optimized to minimize the MAE have intrinsic flaws in the way they assess the tails of the quality range, i.e. the "very good" and the "very bad" sentences. In §4 we propose different ways to evaluate the impact of this problem, and also clarify what might be an important misunderstanding in what a QE system actually does (§4.2). Finally we propose in §5 several experiments: in §5.1 we show that the problem is not system-specific, and we test two ways to circumvent it in §5.2 and §5.3, but the price to pay in global performance is high.

[1] The MAE is defined as the mean over all instances of the absolute error, where the absolute error is the absolute value of the difference between the predicted and the actual value of the instance. Thus, the MAE score depends on the range of possible values (i.e., two datasets using different ranges cannot be compared).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2205–2216, Dublin, Ireland, August 23-29 2014.

2 Experimental Setup

2.1 Data

In this paper we use the three datasets from the WMT12 and WMT13 QE Shared Task (Callison-Burch et al., 2012; Bojar et al., 2013) which are intended to predict the quality of individual machine-translated sentences: the WMT12 task and the WMT13 task 1.1 and 1.3. The last two datasets are renamed *wmt13a* and *wmt13b* in the rest of this paper. These three datasets differ by the way quality is measured:

- *wmt12:* **effort scores**, which have been assigned by three professional post-editors according to predefined guidelines; scores range from 1: "the MT output is incomprehensible [..]" to 5: "the MT output is perfectly clear [..]". The dataset was cleaned to avoid the cases with a high level of disagreement, and the scores were post-processed to harmonize the scale between the judges.
- *wmt13a:* **HTER scores**, which measure the distance between the MT output and the post-edited sentence (Snover et al., 2006).
- *wmt13b:* **post-editing time**, that is, the time that the post-editor has spent correcting the MT output.

As a consequence, the set of scores have different characteristics: in *wmt12*, the distribution is highly discrete due to the integer values assigned by the judges. In *wmt13a* the distribution is more dense, whereas in *wmt13b* some values are spread extremely far from the mean.[2] General statistics for the datasets are given in table 1. In all datasets the input and MT output sentences are available to the system; the post-edited version of the sentences is also available, but it cannot be used by the QE systems (the test set post-edited sentences were provided only after the end of the task). We focus on predicting an absolute indication of quality rather than only ranking the sentences by quality; this is why we use the Mean Absolute Error (MAE) as the main evaluation measure rather than Spearman's correlation or DeltaAvg (Callison-Burch et al., 2012).

2.2 Supervised QE System

In the observations and experiments described in this paper we use a QE system which follows a standard supervised learning approach: it was trained on the full training set for every task considered; when the performance on the training set is observed, it was assessed using 10-fold cross-validation (thus obtaining a prediction for every sentence in the train set based on a 90% subset). We have used Quest[3] (Shah et al., 2013), an open-source tool for QE, to compute the 17 "black box features" which are also used in the WMT QE "baseline" system (see below). We have used Weka (Hall et al., 2009) (version 3.6.10), and after testing several options[4] we found that using the *SMOreg* algorithm (Smola and Schölkopf, 2004; Shevade et al., 2000) with an RBF kernel[5] was optimal with respect to the performance on the three datasets.

We did not perform any feature selection or parameter tuning, because our main goal was to build a generic system. Additionally we favor the ease of reproducibility over optimal performance, which is out of the scope of this paper. We want our system to be as generic as possible (but still performing decently, of course), because we need it to be fairly representative of standard, state-of-the-art, supervised learning QE systems. This is very important, since our observations and experiments are supposed to generalize to the current most common approaches in QE.

Our task of making the system representative of state-of-the-art QE systems has been greatly facilitated by the fact that the organizers of the WMT12 and WMT13 QE Shared Task provide for every task the performance of a so-called "baseline system". We can use exactly the same set of features and compare the results of our system against these obtained by this baseline system, which in turn does not deserve

[2] This is why we exclude the most striking outlier from the training set: 1115.906, line 294. The test set is left unchanged.

[3] `http://staffwww.dcs.shef.ac.uk/people/L.Specia/projects/quest.html` – last verified 05/14.

[4] In particular, M5P regression trees generally achieve nearly as good performance as SVM regression. We have also observed that at least the most important characteristics reported in this paper for an SVM system hold for M5P regression as well.

[5] With the default value C=1 and standardization of the features values.

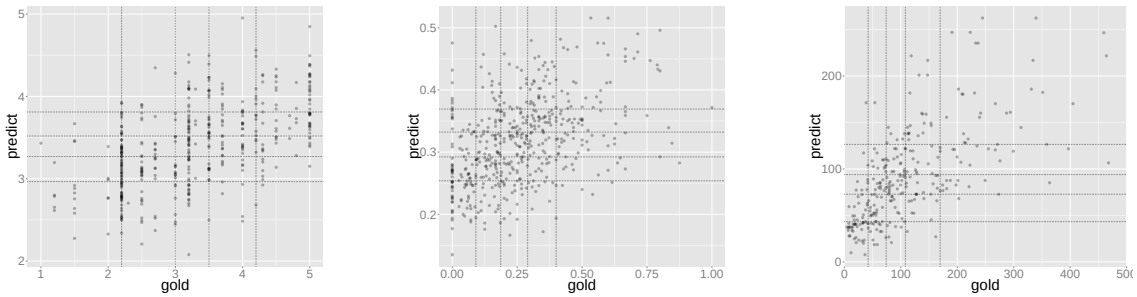
its name since it has actually always performed well in every task: it ranked 8th out of 20 in the WMT12 official ranking, 12th out of 17 in WMT13a, and 6th out of 14 in WMT13b (MAE ranking). Thus, we can simply check that our system performs as well as this baseline system to ensure that it is equivalent, and therefore probably reasonably similar to the other supervised systems submitted to the Shared Tasks which perform similarly.[6] Table 1 shows that our system performs roughly the same as the baseline system on the three datasets.

Dataset	Range of values	Quality direction	Statistics						Performance (test set)			
			Train set			Test set			Our system		Baseline system	
			instances	mean	std. dev.	instances	mean	std. dev.	cor.	MAE	cor.	MAE
wmt12	$[1, 5]$	$\rightarrow$	1832	3.44	0.88	422	3.29	0.98	0.56	0.69	0.58	0.69
wmt13a	$[0, 1]$	$\leftarrow$	2254	0.32	0.17	500	0.26	0.19	0.44	0.15	0.46	0.15
wmt13b	$[0, +\inf[$	$\leftarrow$	802	95.6	84.2	284	116.9	108.3	0.70	50.9	0.70	51.9

Table 1: **Datasets: statistics and performance.** Quality direction: $\rightarrow$ means that the quality is better when the score is higher, $\leftarrow$ means the opposite; "cor." is the Spearman's correlation.

3 The Tails Prediction Problem

In this section we mostly observe the training set (using cross-validation), in order to dismiss the possibility that the observed phenomenon is caused by the differences in the distributions of scores between the training set and the test set. Since it is easier for a supervised learning algorithm to annotate some data from the set it was trained on than from a different dataset, problems which appear with the former are very likely to appear as well (possibly accentuated) with the latter.



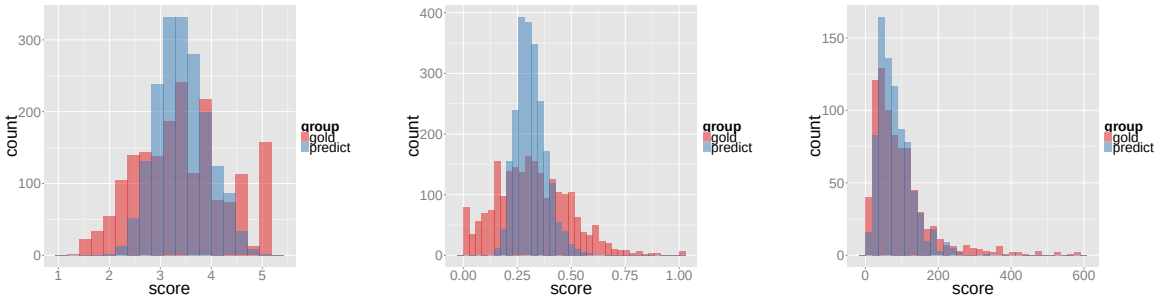
(a) wmt12. Spearman cor.: 0.53 (b) wmt13a. Spearman cor.: 0.37 (c) wmt13b. Spearman cor.: 0.62

Figure 1: **Scatter plots showing how predicted scores differ from gold scores (test set).** Every point (X,Y) corresponds to one sentence for which X is the gold score and Y the predicted score. Darker areas correspond to more dense areas; the vertical and horizontal lines indicate the frontiers of 20%-quantiles for both variables (for instance, the points which are on the right side of the rightmost vertical line account for the 20% highest gold scores). Remark: a few outliers are not visible on the wmt13b plot (their gold scores are higher than 500, and their predicted scores are lower than 250).

Figure 1 shows that the points are very scattered and do not follow the diagonal very closely, but also that the range of predicted scores is significantly different from the range of gold scores: no sentence is predicted below 2 for wmt12, above 0.55 for wmt13a and above 260 for wmt13b, whereas the corresponding range of gold scores is much wider. Figure 2, which shows the distribution of gold vs. predicted scores for the training sets, gives a more precise picture of this difference: in all three datasets, the predicted scores tend to belong to a smaller set of values centered approximately around the mean. There are clearly more predicted values than gold values in this area, and this is confirmed by the much smaller standard deviation for the predicted scores.

It is possible to obtain a clearer picture by "flattening" the distribution, that is, instead of drawing histograms in which points with the same value (or a close value) are accumulated, we represent every

[6] In section 5.1 we also check more specifically that our observations hold for most of the systems submitted to WMT12.

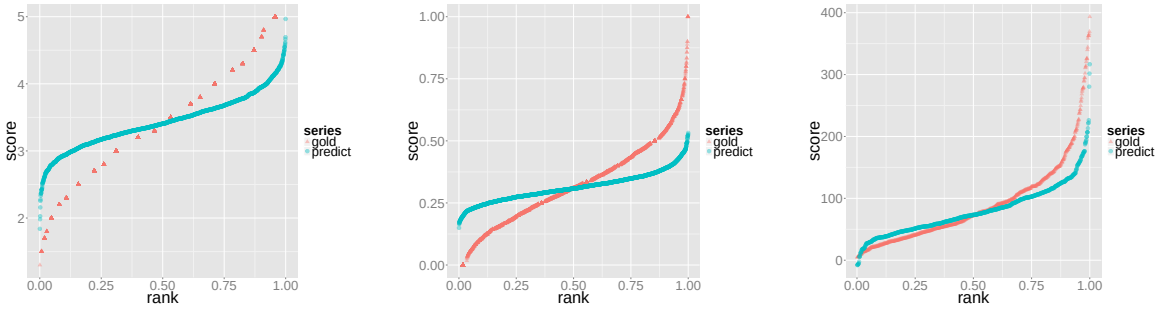


(a) wmt12. $\sigma_G = 0.88,\ \sigma_P = 0.50$ (b) wmt13a. $\sigma_G = 0.17,\ \sigma_P = 0.07$ (c) wmt13b. $\sigma_G = 84.2,\ \sigma_P = 46.5$

Figure 2: **Combined distributions of the gold scores and predicted scores** on the training set for the three datasets. σ_G (resp. σ_P) is the standard deviation for gold (resp. predicted) scores.

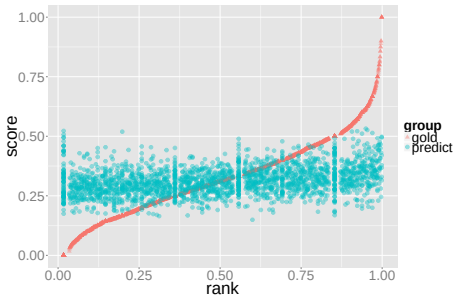
point on the X axis and sort the values on this axis, so that their actual value can be observed on the Y axis, as shown on figure 3. This figure shows that, in all three cases, the predicted scores are tightly clustered around the median, which is the point where the two curves cross each other. If the system predicted scores according to the distribution it observed on the training set, the two curves would be close; instead, they clearly diverge from each other as the distance to the median increases. This means that the model tends globally to overestimate the points below the median and, symmetrically, underestimate the points above the median (though the symmetry is degraded in 3c, since the range is unbound to the right).

In figure 3 the two sets of points are sorted independently: the sentence (x, y) on the curve of gold scores is different from the one with the same x on the curve of predicted scores. Yet this observation of "tightened" predicted scores cannot be fully understood without taking into account the risk of error in the prediction process, as it was visible on the scatter plots in figure 1. Thus it is also useful to look at the sorted scores, but with their corresponding predicted score (for the same sentence) plotted on the same x coordinate; this what is shown on figure 4, for the *wmt13a* dataset only (because the phenomenon is the most accentuated in this dataset, and scores conveniently belong to $[0, 1]$). On figure 4a one can see that the set of predicted scores are mostly contained in a slightly inclined rectangle; clearly they do not follow the curve of gold scores, but here one can see why: the fact that there are many points at the same level on the Y axis along the whole X axis shows that the algorithm cannot make a clear distinction between the different levels of quality. For example, there are approximately as many scores predicted around 0.3 which correspond to actually very good (rank near 0) and very bad sentences (rank near 1). From a different perspective, figure 4b shows very clearly that the farther the gold score of a sentence is from the mean (0.32), the more likely it is to be predicted with a large error.

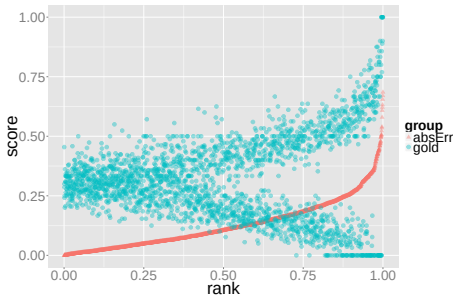


(a) wmt12 (b) wmt13a (c) wmt13b

Figure 3: **Sorted gold and predicted scores** for the three datasets. The two sets of scores are sorted independently. The X axis is the normalized rank (0 to 1 instead of 1 to the total number of sentences), so that it is easier to observe the quantiles. Example: for wmt13a, the lowest fourth of gold scores ranges from 0 to around 0.20, whereas the lowest fourth of predicted scores ranges from 0.125 to around 0.27. Remark: on the wmt13b plot the scores higher than 400 are not visible (all are gold scores).



(a) **Gold scores as reference.** Sentences are sorted by their gold score; the X axis gives their corresponding rank; the *predicted* score of a sentence is plotted on the same abscissa, thus showing both the gold (in red) and predicted score (in blue) of the sentence on the Y axis. The predicted scores which appear on the same vertical line correspond to different sentences which have the same (or very close) gold scores.



(b) **Absolute error as reference.** Sentences are sorted by their absolute error; the X axis gives their corresponding rank; the *gold* score of a given sentence is plotted on the same abscissa, thus showing both the error (in red) and gold score (in blue) of the sentence on the Y axis. The gold scores which appear on the same vertical line correspond to different sentences which have the same (or a very close) absolute error.

Figure 4: **wmt13a, training set: sentences sorted by gold score (left) or absolute error (right).**

The issue is constant among the datasets, but with a variable impact. To some extent, it could be summarized in the following way: it appears that the system does not try to predict the actual quality of the sentences, but instead applies a simple optimization strategy; since a large majority of sentences belong to a relatively small range of values in the middle of the full possible range of scores, predicting any score outside this range is taking a big risk. Consequently it is safer, in order to minimize the error rate, to ignore (or barely take into account) the rare cases which belong to the tails. Hence the system ends doing the opposite of what is usually expected from a quality estimation system: the most common cases are rather accurately recognized, but the most striking anomalies are left undetected or poorly labelled as such. This behaviour can be explained by the following reasons:[7]

- **The supervised learning optimization criterion is very often the minimization of the MAE,**[8] as in our system. This leads the algorithm to favor the interval of scores where there are many instances, since their weight is more important in the average.
- **The datasets are unbalanced**, which is certainly realistic in terms of application, but it also encourages the algorithm to assign scores in the interval which would be the "default class" in a classification problem; that is, without any clear indication in the features, it is strategically wiser to bet on the most probable answer.
- **The risk is lower with respect to MAE** to assign a score in the middle of the range of possible values rather than at the extremes. For instance if the range is $[1, 5]$ the maximum absolute error at 3 is 2, whereas it is 4 at 1 or 5. However, at least for *wmt13a*, the data shows that, if this hypothesis had a real impact, the predicted scores would be closer to 0.5 than to the mean 0.32.

4 Detecting and Evaluating the Tails Quality

4.1 Possible Measures

We propose below different measures intended to evaluate the impact of the tails prediction problem. Since it can be defined as an increased level of error for sentences which are far from the mean, a simple first measure is the correlation between the distance from the gold score to the mean and the absolute error: this value reflects whether the errors are higher in the tails than close to the mean and to what extent (in other words, it measures how strong the divergence observed on the right part of figure 4b is). Table 2 shows how high Pearson's correlation is in our data.

A simple way to measure the performance locally in the tails is to consider the task as a binary classification problem, as if we were only interested in recognizing whether a sentence belongs to a particular

[7] The first two reasons are actually closely related, they only show different aspects of the same problem.

[8] Especially in the WMT QE tasks, since this is the main evaluation measure for the scoring task.

wmt12						wmt13a						wmt13b					
train			test			train			test			train			test		
all	>0	<0	all	>0	<0	all	>0	<0	all	>0	<0	all	>0	<0	all	>0	<0
0.58	0.54	0.64	0.62	0.64	0.65	0.82	0.84	0.78	0.76	0.86	0.76	0.80	0.89	-0.18	0.83	0.91	-0.01

Table 2: **Correlation between the absolute error and the distance to the mean of the gold score.** ">0" (resp. "<0") is the correlation when taking only into account the scores above (resp. below) the mean; this gives a more precise picture for the top/bottom quality scores. For example, in *wmt13b* the top quality (lowest) scores are very well predicted, as opposed to the bottom quality (highest) scores.

subset of scores. For example, the frontier between the classes can be fixed between the 90% lowest scores (negative) and the 10% highest (positive): it is then possible to observe the last 10% using the standard evaluation measures: precision (proportion of true positive among the sentences labeled as positive), recall (proportion of sentences labeled as positive among all positive sentences) and F1-score (harmonic mean of the precision and recall).[9] The values of these measures are given for three thresholds in table 3. As expected, the recall is extremely low in the tails; it is even 0 in most cases for the 5% threshold, which means that the system does not assign any score in the 5% top/bottom of the range observed on the training data.

Data+tail	5%				10%				20%			
	limit	P	R	F1	limit	P	R	F1	limit	P	R	F1
wmt12 B	≤ 2.0	–	0.0	–	≤ 2.3	0.33	0.01	0.02	≤ 2.7	0.73	0.16	0.26
T	≥ 5.0	–	0.0	–	≥ 4.7	0.50	0.02	0.04	≥ 4.2	0.65	0.18	0.28
wmt13a B	≥ 0.62	–	0.0	–	≥ 0.54	–	0.0	–	≥ 0.47	0.62	0.11	0.19
T	≤ 0.06	–	0.0	–	≤ 0.11	–	0.0	–	≤ 0.17	0.50	0.01	0.01
wmt13b B	≥ 272	–	0.0	–	≥ 186	0.76	0.26	0.39	≥ 134	0.71	0.43	0.54
T	≤ 18.2	0.5	0.05	0.09	≤ 24.8	0.18	0.06	0.10	≤ 35.7	0.52	0.30	0.38

Table 3: **Local classification measures (test set).** "T" (resp. "B") refers to the top (resp. bottom) quality tail; P/R/F1 are the standard Precision/Recall/F1-score.[10] Example: 10% of the scores for the *wmt12* training data are higher than 4.7 (top quality tail); among the gold scores in the test set which are higher than this 4.7 threshold, only 2% are predicted as higher than 4.7 (recall); and among the scores predicted as higher than 4.7, exactly 50% are actually higher than 4.7 (precision).

Additionally, we have separately proposed a measure which aims to evaluate the ranking error locally (Moreau and Vogel, 2013). The same idea can be applied to scoring errors: the Local MAE (LMAE) can be computed on a particular range of scores. The difference with global MAE is that, for a given sentence, the gold score or the predicted score can belong to the range while the other does not. This is why there are two versions of this measure: gold-based LMAE and prediction-based LMAE, which, as their names suggest, take into account only the gold scores (resp. predicted scores) which belong to the range in the absolute difference $\mid gold - predicted \mid$, as defined in definition 4.1.

Definition 1 (Local MAE (LMAE))**.** *Let S be a set of sentences, and D the interval of possible scores:*

[9] In the observations which follow we choose to set the limits (5%, etc.) based on the training set even though the test set is observed. In other words, the absolute score corresponding to the percentage is calculated using the training set gold scores, which might differ from the value calculated from the test set. The disadvantage is that the number of values in the test set in the corresponding range does not necessarily correspond to the percentage, but this way the limits do not depend on the test set, so that values obtained on different test sets would be comparable.

[10] We consider the $N\%$ limits computed from the range of gold scores, and not from the range of predicted scores: this makes more sense because otherwise the system is not evaluated against the actual scores in the tails, but since the range of predicted scores is actually smaller than the range of gold scores, sometimes there are no predicted scores at all in this range of values (especially for the lowest values of N, e.g. 5%). For example in the *wmt12* dataset 5% of the gold scores are below 2, but the system does not predict any value below 2. In such a case we consider that this is equivalent to a classifier which decides not to label any instance in a given category. Since there are no instances labelled as positive at all, the precision is undefined, which makes the F1-score undefined as well. The corresponding cells are marked as "–" in tables 3 and 6.

for every sentence $s \in S$, $predicted(s) \in D, gold(s) \in D$. *For any subinterval* $I \subseteq D$:[11]

$$LMAE_{gold} = mean\left(\left\{ \left| \, gold(s) - predicted(s) \, \right| \, \middle| \, s \text{ such that } gold(s) \in I \right\} \right)$$
$$LMAE_{pred} = mean\left(\left\{ \left| \, gold(s) - predicted(s) \, \right| \, \middle| \, s \text{ such that } predicted(s) \in I \right\} \right)$$

To some extent, the gold-based LMAE (resp. prediction-based) is similar to a recall measure (resp. precision) because it takes into account the true positive and the false negative (resp. the true positive and the false positive) with respect to the range. This can be observed in table 4, which gives the values of these two measures for three thresholds on the three datasets: $LMAE_{gold}$ is almost always much higher than the global MAE, whereas there $LMAE_{pred}$ is often close to or lower than the global MAE. This is because, compared to the gold scores, the top or bottom predicted scores are closer to the centre of the range. Therefore the sentences taken into account include some actual "tails sentences" (for which the absolute error is high), but they can also contain many sentences which actually belong to the area (for which the absolute error is low).

		5%		10%		20%	
Data+tail	(Global) MAE	$LMAE_{gold}$	$LMAE_{pred}$	$LMAE_{gold}$	$LMAE_{pred}$	$LMAE_{gold}$	$LMAE_{pred}$
wmt12 B	0.69	1.37	0.47	1.02	0.57	1.02	0.62
T		1.08	0.68	1.08	0.67	0.89	0.68
wmt13a B	0.15	0.35	0.18	0.28	0.18	0.19	0.17
T		0.28	0.12	0.28	0.13	0.24	0.13
wmt13b B	50.9	264	154	192	129	135	90.3
T		26.1	22.9	24.5	27.4	27.4	25.2

Table 4: **Local MAE evaluation (test set).** "T" (resp. "B") refers to the top (resp. bottom) quality tail. Example: for the wmt13b data, among the 10% actual top quality sentences (i.e. the 10% lowest gold scores), the mean absolute error is 26.1. This is lower than the global MAE (50.9), as opposed to all the other cases; this confirms that the top quality tail in *wmt13b* is particularly well predicted (this is certainly a consequence of the strongly skewed distribution in this dataset).

4.2 The Post-edited Sentences Test

A good way to evaluate the discrepancies in the reliability of the quality scores in the tails is to apply the QE system to a set of very good or very bad sentences. Thankfully the post-edited versions of the sentences were provided with the WMT datasets; since by definition their quality is perfect, they make a perfect case for such a test.[12] In theory, all these sentences should be assigned a score close to top quality.[13] For every dataset we run the same QE system, i.e., we compute the features for the post-edited sentences using Quest, then apply the model built with the regular training data to these features. We tried with both the post-edited version of the training set and test set, when provided.[14]

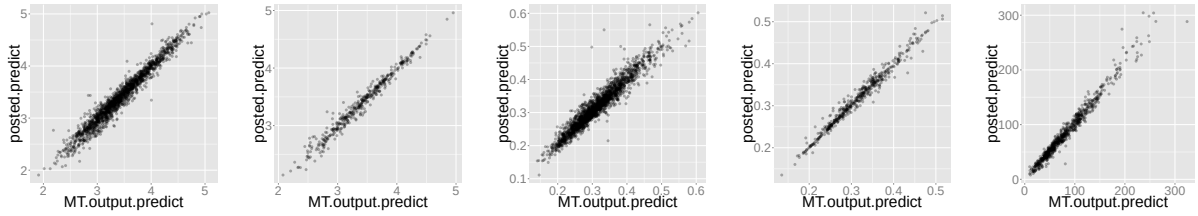
Our original goal was to observe how high the error rate was globally, but it turned out that the predicted scores follow a distribution which is very similar to the one followed by the MT output (the means are very close as well, which implies that the MAE is very high). This led us to observe how the MT output scores and the post-edited version scores are correlated. In most cases the two scores are very close, as shown on figure 5. This is obviously a very serious issue, since it means that, in general, the system is not able to distinguish between a sentence which needs correction and the same sentence after correction.

[11] Remark: if $I = D$, $LMAE_{gold} = LMAE_{pred} = MAE$.

[12] Independent assessment of the post-edited sentences is, of course, not guaranteed to yield the judgement that they would not benefit from further editing, though.

[13] That is, 5 for the *wmt12* dataset and 0 for the *wmt13a* dataset (since HTER scores measure the distance against the post-edited version, and here we compare the post-edited sentence against itself); the *wmt13b* dataset is based on post-edited time, so there is no exact value corresponding to perfect sentences but the scores should very low.

[14] The post-edited version was not available for the wmt13b test set.



(a) wmt12, train set; cor.=0.977; mean diff.=-0.03=-0.03σ

(b) wmt12, test set; cor.=0.986; mean diff.=-0.02=-0.02σ

(c) wmt13a, train set; cor.=0.962; mean diff.=0.004=0.02σ

(d) wmt13a, test set; cor.=0.987; mean diff.=0.002=0.01σ

(e) wmt13b, train set; cor.=0.985; mean diff.=3.14=0.04σ

Figure 5: **MT output predicted scores vs. post-edited predicted scores** "mean diff." is the mean of the difference between the post-edited score and the MT output score; it is also expressed as a multiple of σ, where σ is the standard deviation of the MT output gold scores (specific to each particular dataset).

It is however able to see a slight difference at the document level: we have performed a paired Student's test for each dataset, which shows that the mean of the scores predicted for the post-edited sentences is significantly lower in the *wmt12* case and higher in the *wmt13a* and *wmt13b* cases (as expected by the definition of scores) than the scores predicted for the MT output sentences. Nevertheless, the mean difference is extremely low (see figure 5), never higher than 0.04 standard deviations.

Furthermore, there is no visible impact of the quality of the MT output, although one would expect the correlation to be lower for low quality sentences: by definition, there are more differences between the MT output and the post-edited version for these sentences, so it should be easier for the system to detect the different level of quality between the two. In other words, it is quite understandable that the system does not detect the difference for an MT output of relatively good quality, but the fact the post-edited version of the really bad translations are also rated as really bad is a major issue. It must be remembered that we are not refering to a flaw solely in our own system, but nearly across the board in the state of the art systems.

These observations, which hold for every dataset, show that QE systems do not capture the actual quality of the sentences: instead, it seems that what they measure is probably the *difficulty of machine-translating a sentence*. Indeed, the set of Quest features that we use contains many features which depend only on the source sentences. Moreover, this conclusion is consistent with the fact that Biçici et al. (2013) obtain very good results on the WMT12 dataset using only the source sentences.

Explaining this observation with precision would require a more detailed analysis which is out of the scope of this paper. Nevertheless, it is fairly clear that the features which are used fail to capture the subtlety and/or the diversity of the difference between a faulty sentence and its corrected version; this might be because a single sentence does not offer enough clues for the system to make such a fine-grained distinction, in which case it would be necessary to rethink the definition of the QE problem.

In other work, we examine linguistic quality of items in relation to reference corpora (Moreau and Vogel, 2013; **?**). By comparison to the supervised learning studied here, such work is weakly supervised since there is no use of absolute scores. This yields a version of the QE problem that may be deemed too relativistic, but does represent an alternative approach. Unfortunately, because of the very difference in the use of absolute scores, they cannot be directly compared on this. Thus, we focus here on empirical exploration of the nature of the problem in estimating quality in the case of supervised learning.

5 Experiments

In this section we devise several experiments intended to explore different aspects of the problem in more detail. In particular, we try to evaluate the impact of the possible causes described in §3: first we show in §5.1 that it affects most QE systems, especially those optimized to minimize MAE. Then in §5.2 and §5.3 we confirm that the distribution of the training set is a major cause of the issue by showing that alternative distributions have different effects.

5.1 Tails Prediction for WMT12 Participating Systems

In order to test if the tails prediction problem is general to most supervised QE systems, we apply the local performance measures to the scores predicted on the test set by the participating systems in WMT12.[15] Table 5 shows some detailed results for the four best systems at WMT12. It confirms that the predictions made for the tails are generally significantly worse than they are globally, and especially that the systems tend to predict very few values at the ends of the range of values: recall in the 10% bottom or top scores is never higher than 12%.[16] It is also worth noticing that the first system, which performs significantly better than the others, is the only one which was not optimized to minimize the MAE but to maximize the DeltaAvg score (Soricut et al., 2012). In particular, this system obtains a recall higher than the others in most cases (especially in the 5% and 10% tails), which is certainly due to the fact that it assigns more scores far from the mean (in other words, this system takes more risk). This tends to confirm our hypothesis that the minimization of the MAE as learning criterion is one of the causes of the problem.

System ID	Global MAE	Correlation dist.mean. vs abs.err.	Bottom 5% R	Bottom 5% G-LMAE	Bottom 10% R	Bottom 10% G-LMAE	Bottom 20% R	Bottom 20% G-LMAE	Top 5% R	Top 5% G-LMAE	Top 10% R	Top 10% G-LMAE	Top 20% R	Top 20% G-LMAE
SDLLW_M5PbestDeltaAvg	0.61	0.49	0.05	1.02	0.07	0.76	0.32	0.76	0.02	0.96	0.12	0.99	0.26	0.84
UU_best	0.64	0.53	0.0	1.21	0.07	0.91	0.26	0.91	0.0	1.02	0.04	1.01	0.22	0.81
SDLLW_SVM	0.64	0.55	0.0	1.33	0.0	0.98	0.17	0.98	0.02	0.89	0.06	0.91	0.32	0.75
UU_bltk	0.64	0.58	0.0	1.22	0.06	0.91	0.27	0.91	0.0	1.07	0.02	1.05	0.27	0.83

Table 5: **Tails prediction quality for the 4 best systems at WMT12 (test set).** The second column contains the correlation between the distance to the mean and the absolute error; the columns R and G-LMAE contain respectively the recall and the gold-based local MAE scores (see §4.1).

5.2 Adding the Post-edited Sentences to the Training Set

In this experiment we use the post-edited sentences again (see §4.2), but this time adding them to the training set in order to observe the impact on the test set.[17] These instances are progressively added to the official training set (in random order). We focus on the top quality tail, since it is the one which is expected to benefit from adding sentences with top scores to the training set. Figure 6 shows how the local MAE scores improve as post-edited instances are added. Only the gold-based LMAE scores are represented, because these provide a recall-like information and the observations show that recall (in the tails) is the main weakness of QE systems (see §4.1).

As expected, in all cases adding top quality sentences to the training set makes the system decrease the error rate in the top quality tail. Of course this local improvement comes at the price of degrading the global performance, although for the *wmt13a* dataset (fig. 6b) the global error even improves until almost half of the sentences have been added. In the case of the *wmt13b* dataset (fig. 6c), since the QE system was already very good in predicting the top quality sentences (the LMAE is even better than the global MAE), the improvement is smaller and proportionally more costly for the global performance.

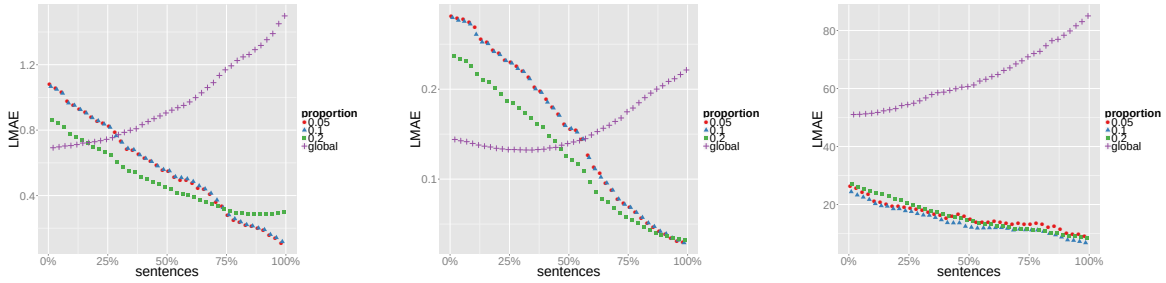
5.3 Balancing the Training Set

In this final experiment, we resample the training set (with replacement), in order to balance the gold scores over the full range of values. Since we can only use the discrete gold scores provided with the original training set, we compute a (random) uniform distribution but select the closest available score (randomly picking an instance among those with this score). The resulting distribution is not uniform, and the training set contains many duplicate instances; therefore, the resulting training set is unlikely to yield very good results in general, but it is no longer subject to the "statistical attraction" towards the mean that we have observed.

[15] These values were kindly provided by the organizers of the WMT12 QE Shared Task.

[16] This is true for all but 3 participating systems, and these exceptions correspond to systems which performed worse globally.

[17] We assign perfect scores to all these sentences: 5 for *wmt12*, 0 for *wmt13a*; for *wmt13b*, we use the mean of the time spent for the sentences in the training set which were left unmodified: there are 23 such sentences, and the mean is 16.19s.



(a) wmt12. (b) wmt13a. (c) wmt13b.

Figure 6: **Improvement of gold-based LMAE as post-edited sentences are added to the train set.** Example: in *wmt12*, the gold-based LMAE for the top 20% sentences is higher than 0.8 when the system is trained only on the official train set (0% of the post-edited sentences added), but reaches 0.4 when about half of the post-edited sentences are added to the training set. However the global MAE (which takes all the sentences into account) increases from 0.7 (0%) to 0.9 (50% of the post-edited sentences added): since the system assigns more scores in the top tail, it makes larger errors globally. Remark: the MAE and LMAE values are measured on the same set of sentences for every percentage on the X axis.

The model obtained from the balanced training set has been applied to the original test set. Table 6 gives the local results observed in the tails: in most cases, the recall increases drastically compared to using the regular training set, or is at least identical,[18] causing a great increase in the F1-scores as well. The LMAE scores do not show such an improvement, in fact the mean error is often higher than with the regular training set. This is due to the fact that the system is forced to assign scores far from the "easy cases" around the mean, therefore makes much bigger mistakes than in the previous case. As expected, the global MAE scores for *wmt13a* and *wmt13b* are much higher than the original MAE values (0.27 and 110.7 respectively, i.e. about twice the original values). Interestingly, the MAE stays almost constant (0.71 instead of 0.69) for *wmt12*. The correlation between the distance to the mean and the mean absolute decreases to 0.42, 0.05 and 0.26 for *wmt12*, *wmt13a* and *wmt13b*, respectively.

Data+tail	Classification measures												Local MAE measures					
	5%				10%				20%				5%		10%		20%	
	limit	P	R	F1	limit	P	R	F1	limit	P	R	F1	gold	pred.	gold	pred.	gold	pred.
wmt12 B	≤ 2.0	**0.31**	**0.18**	**0.23**	≤ 2.3	**0.49**	**0.20**	**0.28**	≤ 2.7	0.66	**0.46**	**0.54**	**0.94**	0.82	**0.73**	0.75	**0.73**	0.67
T	≥ 5.0	–	0.0	–	≥ 4.7	0.50	0.02	0.04	≥ 4.2	**0.69**	**0.19**	**0.29**	1.28	0.72	1.26	**0.65**	1.07	0.71
wmt13a B	≥ 0.62	**0.15**	**0.70**	**0.24**	≥ 0.54	**0.17**	**0.75**	**0.28**	≥ 0.47	0.21	**0.83**	**0.33**	**0.14**	0.60	**0.13**	0.55	**0.14**	0.49
T	≤ 0.06	–	0.0	–	≤ 0.11	–	0.0	–	≤ 0.17	**0.67**	**0.02**	**0.04**	0.58	**0.11**	0.57	0.14	0.44	0.15
wmt13b B	≥ 272	**0.15**	**0.55**	**0.23**	≥ 186	0.27	**0.78**	**0.41**	≥ 134	0.38	**0.91**	0.54	**156**	243	**118**	235	**101**	199
T	≤ 18.2	0.20	**0.20**	**0.20**	≤ 24.8	**0.30**	**0.19**	**0.24**	≤ 35.7	**0.55**	0.26	0.35	128	61	114	45	110	41

Table 6: **Local evaluation of the test set using a balanced training set.** Cells in bold show an improvement over the corresponding value with the original training set, as given in tables 3 and 4. The classification limits were computed on the original training set.

6 Conclusion and Future Work

To conclude, we have shown that there are very serious issues with the way supervised QE systems are built: they tend to be unable to reliably evaluate both the worst and the best quality sentences. Furthermore, they cannot distinguish between a faulty MT output sentence and its post-edited version. We have also shown that it is possible to improve the detection of the best/worst sentences by altering the distribution of the training set; however the question whether this can be achieved while maintaining a decent level of global performance remains open. But even if the cost in global performance is high,

[18]The only exception is the 20% top quality recall of the wmt13b dataset. This is certainly due to the very particular distribution of scores in this dataset, and to the fact that the top quality tail was already predicted reliably in the regular version.

the techniques that we have tested could be useful in some specific applications of QE (for example, if the recall in the tails is more important than the precision).

We think that these observations raise questions about the definition of the QE problem. It might actually be necessary to define different kinds of QE tasks: depending on the targeted application (e.g. estimating post-editing time, retraining the MT model, discarding the worst sentences, etc.), there could be a specific setting which is more appropriate in terms of supervised/unsupervised learning, evaluation measure, precision/recall trade-off, etc. For instance, minimizing the MAE does not seem compatible with detecting anomalies, but might be relevant for estimating the cost of post-editing. Similarly, under the hypothesis that the sentence level is not sufficiently rich in information in order to obtain accurate predictions, an intermediate level of granularity might be considered (e.g. at paragraph level).

Finally, another great challenge with respect to the reliability of QE systems is their consistency when applied to different test sets, or more generally their dependency on the training set: in the perspective of applications, it is very important to know what level of confidence can be expected when applying a QE system or model to a new document.

Acknowledgements

We are grateful to Lucia Specia, Radu Soricut and Christian Buck, the organizers of the WMT 2012 and 2013 Shared Task on Quality Estimation, for releasing all the data related to the competition, including post-edited sentences, features sets, etc.

This research is supported by Science Foundation Ireland (Grant 12/CE/I2267) as part of the Centre for Next Generation Localisation (`www.cngl.ie`) funding at Trinity College, University of Dublin.

The graphics in this paper were created with `R` (R Core Team, 2012), using the `ggplot2` library (Wickham, 2009).

References

Ergun Biçici, Declan Groves, and Josef Genabith. 2013. Predicting sentence translation quality using extrinsic and language independent features. *Machine Translation*, 27(3-4):171–192.

Ondrej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Eighth Workshop on Statistical Machine Translation*, WMT-2013, pages 1–44, Sofia, Bulgaria.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, Montreal, Canada, June. Association for Computational Linguistics.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. 2009. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Erwan Moreau and Carl Vogel. 2013. Weakly supervised approaches for quality estimation. *Machine Translation*, 27(3):pp 257–280, September.

R Core Team, 2012. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Kashif Shah, Eleftherios Avramidis, Ergun Biçici, and Lucia Specia. 2013. Quest - design, implementation and extensions of a framework for machine translation quality estimation. *Prague Bull. Math. Linguistics*, 100:19–30.

S.K. Shevade, SS Keerthi, C. Bhattacharyya, and K.R.K. Murthy. 2000. Improvements to the SMO algorithm for SVM regression. *Neural Networks, IEEE Transactions on*, 11(5):1188–1193.

A.J. Smola and B. Schölkopf. 2004. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *In Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Radu Soricut, Nguyen Bach, and Ziyuan Wang. 2012. The SDL Language Weaver systems in the WMT12 Quality Estimation shared task. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 145–151, Montréal, Canada, June. Association for Computational Linguistics.

Hadley Wickham. 2009. *ggplot2: elegant graphics for data analysis*. Springer New York.

Augment Dependency-to-String Translation with Fixed and Floating Structures

Jun Xie[†] **Jinan Xu**[‡] **Qun Liu**[†§]
†Key Laboratory of Intelligent Information Processing,
Institute of Computing Technology,Chinese Academy of Sciences
xiejun@ict.ac.cn
‡School of Computer and Information Technology, Beijing Jiaotong University
xja2010@gmail.com
§School of Computing, Dublin City University
qliu@computing.dcu.ie

Abstract

In this paper, we propose an augmented dependency-to-string model to combine the merits of both the head-dependents relations at handling long distance reordering and the fixed and floating structures at handling local reordering. For this purpose, we first compactly represent both the head-dependent relation and the fixed and floating structures into translation rules; second, in decoding we build "on-the-fly" new translation rules from the compact translation rules that can incorporate non-syntactic phrases into translations, thus alleviate the non-syntactic phrase coverage problem of dependency-to-string translation (Xie et al., 2011). Large-scale experiments on Chinese-to-English translation show that our augmented dependency-to-string model gains significant improvement of averaged +0.85 BLEU scores on three test sets over the dependency-to-string model.

1 Introduction

As a representation holding both syntactic and semantic information, dependency grammar has been attracting more and more attention in statistical machine translation. Lin (2004) took paths as the elementary structures and proposed a path-based transfer model. Quirk et al. (2005) extended path to treelets (connected subgraphs of dependency trees) and put forward dependency treelet translation. Ding and Palmer (2005) proposed a model on the basis of dependency insertion grammar. Shen et al. (2008) employed the fixed and floating structures as elementary structures and proposed a string-to-dependency model with state-of-the-art performance. Xie et al. (2011) employs head-dependents relations as elementary structures and proposed a dependency-to-string model with good long distance reordering property. A head-dependents relation (HDR) is composed of a head and all its dependents, which can be viewed as an instance of a sentence pattern or phrase pattern.

However, since dependency trees are much flatter than constituency trees, the dependency-to-string model suffers more severe non-syntactic phrase coverage problem (Meng et al., 2013) than constituency-based models (Galley et al., 2004; Liu et al., 2006; Huang et al., 2006). Non-syntactic phrases are those phrases that can not be covered by whole subtrees. To address this problem, Meng et al. (2013) proposed to translate with both constituency and dependency trees, which can incorporate non-syntactic phrases covered by the constituents of the constituency trees. This model requires both constituency and dependency trees, thus may suffer from both constituency and dependency parse errors. Additionally, there are only few languages that have both constituency and dependency parsers, which limits its practical use.

In this paper, we propose to address non-syntactic phrase coverage problem of the dependency-to-string model without resort to extra resources (Section 3). To this end, we augment the dependency-to-string model at two aspects. First, we combine the merits of both the head-dependent relations and the fixed and floating structures (Shen et al., 2008), and compactly represent these two kinds of knowlege into augmented HDR rules (Section 3.1). We acquire the augmented HDR rules automatically from the

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2217–2226, Dublin, Ireland, August 23-29 2014.

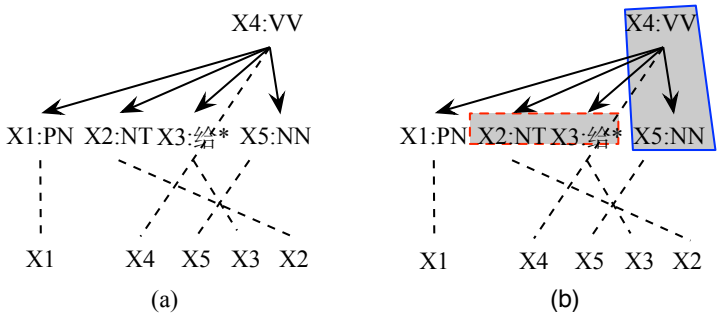


Figure 1: Examples of an HDR rule (a) and an augmented HDR rule (b). Where each "*" denotes a substitute site which is a compact representation of a whole subtree. The shadow with line border indicates a fixed structure and the shadow with dash line border indicates a floating structure.

word-aligned source dependency tree and target string paris (Section 3.2). In decoding we propose an "on-the-fly" rule building strategy, which builds new translation rules from the augmented HDR rules and incorporates non-syntactic phrases into translations (Section 3.4). Large-scale experiments (Section 4) on Chinese-to-English translation show that our augmented model gains significant improvement of averaged +0.85 BLEU points on three test sets over the dependency-to-string model.

2 Background

For convenience of the description of our augmented dependency-to-string model, we first briefly review the dependency-to-string model and the fixed and floating structures of string-to-dependency model (Shen et al., 2008).

2.1 Dependency-to-String Translation

The dependency-to-string model (Xie et al., 2011) takes head-dependents relations as the elementary structures of dependency trees, and represents the translation rules with the source side as HDRs and the target side as string. Since the HDRs in essence relate to phrase patterns and sentence patterns, the HDR rules specify the reordering of these patterns. For example, Figure 1 (a) is an example HDR rule, which represents a reordering manner of a sentence pattern composed of a proper noun (X1:PN), a temporal noun (X2:NT), an prepositional phrase relate to "给 (give)" (X3:给), a verb (X4:VV) and a noun (X5:NN).

With the HDR rules, the dependency-to-string model gets rid of the extra reordering heuristics and reordering models of the previous models (Lin, 2004; Ding and Palmer, 2005; Quirk et al., 2005). More importantly, the model shows state-of-the-art performance and exhibits good long distance reordering property.

2.2 Fixed and Floating Structures

The fixed structures and floating structures are fundamental structures of the string-to-dependency model (Shen et al., 2008), which are introduced to handle the coverage of non-constituent rules. Given the dependency tree $d_1 d_2 ... d_n$ of a sentence $f_1 f_2 ... f_n$, where d_i indicates the parent word index of word f_i.

Definition 1. *A dependency structure $d_{i...j}$ is **fixed** on the head h, where $h \in [i, j]$, if and only if it meets the following conditions:*

- $d_h \notin [i, j]$
- $\forall k \in [i, j]$ *and* $k \neq h$, $d_k \in [i, j]$
- $\forall k \notin [i, j]$, $d_k = h$ *or* $d_k \notin [i, j]$

A fixed structure describes a fragment with a sub-root, where all the children of the sub-root are complete.

Definition 2. *A dependency tree $d_{i...j}$ is **floating** with children C, for a non-empty set $C \subseteq i, ..., j$, if and only if it meets the following conditions:*

- $\exists h \notin [i, j], s.t. \forall k \in C, d_k = h$
- $\forall k \in [i, j]$ *and* $k \notin C$, $d_k \in [i, j]$
- $\forall k \notin [i, j]$, $d_k \notin [i, j]$

A floating structure consists of sibling nodes of a common head, but the head itself is unspecified.

In nature, the fixed and floating structures represent the phrases under the structural constraint of dependency trees, most of them are non-syntactic phrases.

The HDRs are good at handling long distance dependencies, while the fixed and floating structures excels at handling local reordering. This encourages us to address the non-syntactic phrase coverage problem of dependency-to-string model by exploiting these two kinds of structures.

3 Augmented Dependency-to-String Translation

In the following, we will describe our augmented dependency-to-string model in detail, including the augmented HDR rules (Section 3.1), rule acquisition (Section 3.2) and "on-the-fly" rule building in decoding (Section 3.4).

3.1 Augmented HDR rules

Our augmented HDR rules aim at combining the merits of both the HDRs at handling long distance reordering and the fixed and floating structures at handling local reordering. For this purpose, we augment the HDR rules (Xie et al., 2011) by labelling the HDRs with the fixed and floating structures.

Figure 1 (b) shows an example augmented HDR rule. Which is an augmented version of the HDR rule Figure 1 (a) by labelling it with a fixed structure (shadow with line border) and a floating structure (shadow with dash line border). The labeled fixed and floating structures indicate the bilingual phrases that we can incorporate in this sentence pattern.

3.2 Rule Acquisition

Given a word-aligned parallel corpus defined as a set of triples $\langle T, e, A \rangle$, where T is a dependency tree of source sentence f_1^J, e_1^I is the target sentence and A is an alignment relation between f_1^J and e_1^I, we acquire the augmented HDR rules by three steps: tree annotation, acceptable HDR identification and rule induction. The process is similar with that of Xie et al. (2011). However, we make some extensions so that we can take the fixed and floating structures into account.

3.2.1 Tree Annotation

Besides annotating each node of T with *head span* and *dependency span* as Xie et al. (2011), we also label the tree with consistent fixed and floating structures.

Definition 3. *The **head span** hsp(n) of a node n is the closure of the set taking the index of the target words aligned to n as its elements.*

The *closure* of a set contains all the elements between the minimum and the maximum of the set and each element has only one copy. For example, the closure of set $\{1, 3\}$ is $\{1, 2, 3\}$.

We say a head span is *consistent* with alignment if the bilingual phrase it covers is consistent with the alignment (Koehn et al., 2003).

Definition 4. *Given a subtree T' rooted at n, the **dependency span** dsp(n) of n is the closure of the union of the consistent head spans of all the nodes of T'.*

$$dsp(n) = closure(\bigcup_{\substack{n' \in T' \\ hsp(n') \text{ is consistent}}} hsp(n'))$$

If no head spans of all the nodes of T' are consistent, $dsp(n) = \varnothing$.

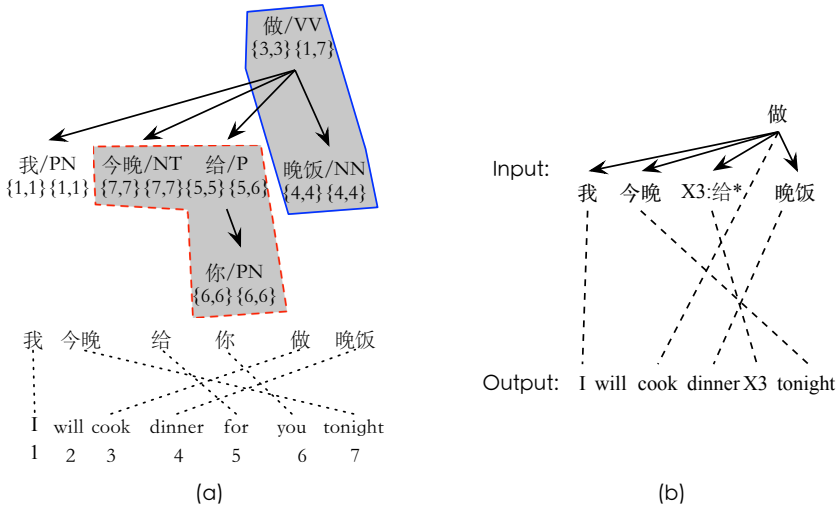


Figure 2: An example annotated dependency tree (a) and an example lexicalized augmented HDR rule (b) induced from the top-level HDR of (a). Each node of the dependency tree is annotated with two spans: head span (the former) and dependency span (the latter). The shadows denote a consistent fixed structure (shadow with line border) and a floating structure (shadow with dash line border). The "*" denotes a substitute site.

Definition 5. *A fixed or floating structure is **consistent** with alignment if the phrase it covers is consistent with alignment.*

Tree annotation can be readily accomplished by a single post-order traversal of dependency tree T. For each accessed node n, annotate it with head span and dependency span according to A. If n is an internal node, enumerate all the fixed and floating structures relate to n, and label those consistent ones on T. Repeat the above process till the root is accessed.

Figure 2 (a) shows an example annotated dependency tree. Where each node is annotated with two spans: head span (the former) and dependency span (the latter). Moreover, the dependency tree is also labeled with two consistent fixed and floating structures that cover phrases "做 饭" and "今晚 给 你" respectively.

3.2.2 Acceptable HDR Identification

From the annotated dependency tree, we identify the HDRs that are suitable for rule induction. These HDRs are called as acceptable HDRs. To this end, we traverse the annotated dependency tree in post-order and identify the HDRs with the following properties:

- for the head, its head span is consistent;
- for the dependents, the dependency span of each dependent should not be $\varnothing$ unless the dependent is a leaf node;
- the intersection of the head span of the head and the dependency spans of the dependents is $\varnothing$ (or do not overlap).

Different from those acceptable HDRs of Xie et al. (2011), the acceptable HDRs here may be labeled with fixed and floating structures. For example, the top level of Figure 2 (a) is an acceptable HDR, which is labeled with a fixed structure and a floating structures. Typically an acceptable HDR has three types of nodes: leaf node (of the dependency tree), internal node (of the dependency tree) and head node (an internal node function as the head of the HDR).

3.2.3 Rule Induction

From each acceptable HDR, we induce a set of lexicalized and unlexicalized augmented HDR rules. This process is similar with that of Xie et al. (2011) except that here we have to consider the consistent fixed

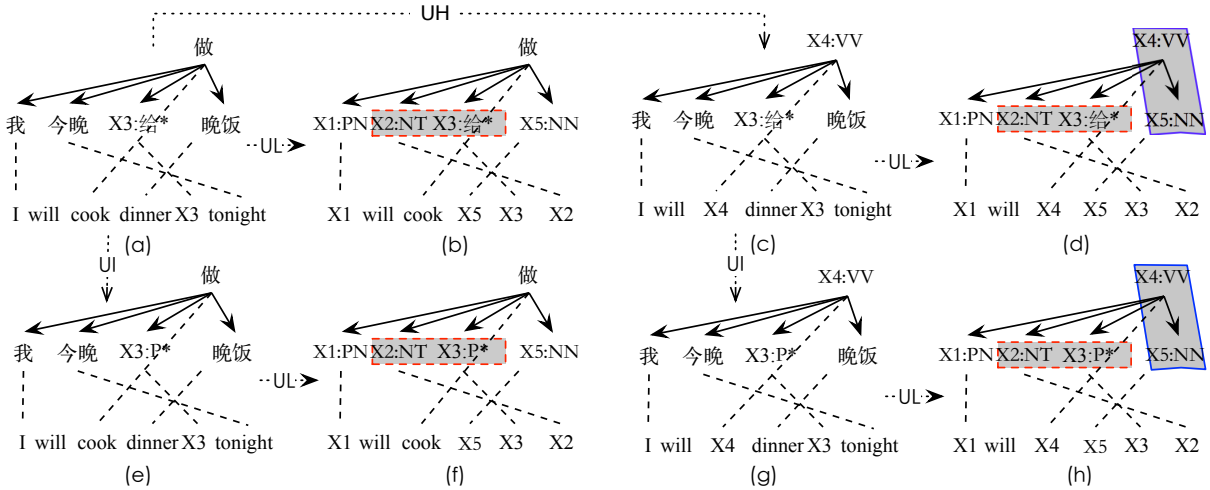


Figure 3: Lexicalized augmented HDR rule (a) and unlexicalized augmented HDR rules (b)∼(h) induced from the top level HDR of the annotated dependency tree in Figure 2. Where "UH", "UI" and "UL" denotes "unlexicalize head", "unlexicalize internal" and "unlexicalize leaf" , respectively. The shadows with line border denote fixed structures and the shadows with dash line border denotes floating structures.

and floating structures.

First, we induce a lexicalized augmented HDR rule with the following principles:

1. extract the HDR, mark each internal node as a variable, and label the HDR with the floating structures that cover only variables. This forms the input of a lexicalized rule.

2. generate the target string according to head span of the head and the dependency spans of the related dependents, and turn the word sequences covered by the dependency spans of the internal nodes into variables. This forms the output of a lexicalized rule.

Figure 2 (b) illustrates a lexicalized augmented HDR rule induced from the top-level HDR of the annotated dependency tree Figure 2 (a).

From each lexicalized augmented HDR rule (along with the acceptable HDR), we then induce a set of unlexicalized augmented HDR rules with the following principles:

1. turn each type (leaf, internal or head) of nodes simultaneously into variables;
2. when turning a head or leaf node into a variable, change the counterpart of the target side into the variable; label the unlexicalized HDR with the fixed and floating structures that cover only variables.
3. when turing an internal node into a variable, keep the counterpart of the target side unchanged.

Totally, we will obtain eight types of augmented HDR rules from an acceptable HDR. In this paper, we call the lexicalized and unlexicalized HDRs generated by the above process as instances of the HDR.

Figure 3 illustrates the rule induction of seven unlexicalized augmented HDR rules (b)-(h) from lexicalized augmented HDR rule (a). Where "UH", "UI" and "UL" on the dash arrows indicate "unlexicalize head", "unlexicalize internal" and "unlexicalized leaf", respectively.

3.2.4 Probability Estimation

We take the augmented HDR rules acquired from word-aligned parallel corpus as the observed data, and employ relative frequency estimation to calculate the translation probabilities of the rules. Note that, here we take the labeled fixed and floating structures of the augmented HDR rules as indicators of bilingual phrases that can be incorporated in the sentence patterns and phrases patterns represented by the HDRs. So we consider only the HDRs when counting the augmented HDR rules.

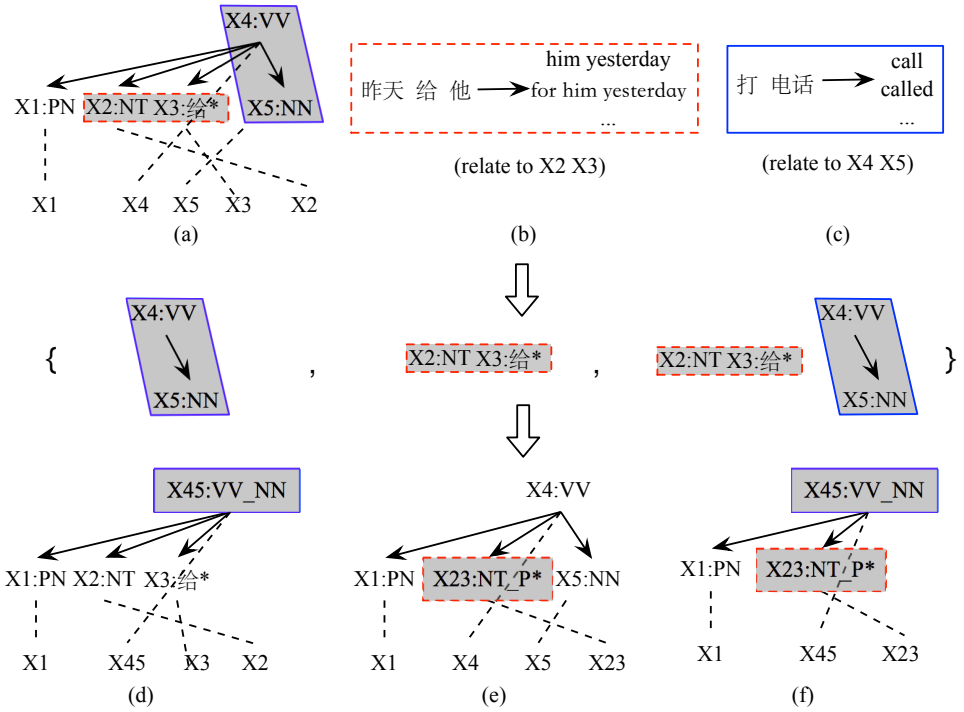


Figure 4: Illustration of "on-the-fly" translation rule building.

3.3 The model

Following Och and Ney (2002), we adopt a general log-linear model for our augmented dependency-to-string model. Let d be a derivation that converts a source dependency tree T into a target string e. The probability of derivation d is defined as:

$$P(d) \propto \prod_i \phi_i(d)^{\lambda_i} \quad (1)$$

where ϕ_i are features defined on derivation and λ_i are feature weights.

In our implementation, we make use of eleven features, including seven features inherited from the dependency-to-string model:

- translation probabilities $P(f|e)$ and $P(e|f)$ and lexical translation probabilities $P_{lex}(f|e)$ and $P_{lex}(e|f)$ of augmented HDR rules
- rule penalty $exp(-1)$
- language model $P_{lm}(e)$
- word penalty $exp(|e|)$, where $|e|$ is the length of the generated target string

and four extra features for bilingual phrases relate to fixed and floating structures:

- translation probabilities $P_{bp}(f|e)$ and $P_{bp}(e|f)$ and lexical translation probabilities $P_{bp\_lex}(f|e)$ and $P_{bp\_lex}(e|f)$ of bilingual phrases

3.4 "On-the-Fly" Decoding

The task of the decoder is to find the best derivation from all possible derivations. Our decoder is based on bottom-up chart parsing, which characterizes at "on-the-fly" translation rule building.

Given an input dependency tree T, the decoder traverses it in post-order. For each accessed node n, the decoder first enumerates all instances of the HDR rooted at n as we do in rule induction, and checks for matched augmented HDR rules. If a matched rule is labeled with fixed and floating structures, the decoder builds new translation rules "on the fly" with the following principles:

1. check the phrases covered by the labeled fixed and floating structures for matched bilingual phrases;
2. if there are no matched bilingual phrases for all labeled fixed and floating structures, take the augmented HDR rule as a HDR rule of dependency-to-string model; otherwise,
   - enumerate all combinations of the fixed and floating structures with matched bilingual phrases;
   - for each combination, build a new translation rule by turning the variable sequences covered by the fixed and floating structures into new variables;
   - the new-built rule inherits the translation probabilities of the deriving augmented HDR rule, and the new variables take the matched bilingual phrases as their translation hypothesis.

Figure 4 illustrates the "on-the-fly" rule building process. Suppose augmented HDR rule (a) is the matched rule, and bilingual phrases (b) and (c) match the phrases covered by the labeled fixed and floating structures of (a). There will be three combinations of the labeled fixed and floating structures as shown in the middle of Figure 4. For each combination, the decoder builds a new translation rule by turning variable sequences "X2:NT X3:给*" and/or "X4:VV X5:NN" into new variables "X23:NT_P*" and/or "X45:VV_NN". And we will obtain three new translation rules (d)-(f) that can incorporate non-syntactic phrases into translations.

If there are no matched rules, the decoder builds a pseudo translation rule with monotonic reordering.

The decoder then employs cube pruning (Chiang, 2007; Huang and Chiang, 2007) to generate k-best hypothesis with integrated language model for node n.

Repeat the above process till the root of T is accessed. The hypothesis with the highest score is output as translation.

4 Experiments

We evaluated our augmented model by comparison with dependency-to-string model and hierarchical phrase-based model on Chinese-to-English translation in terms of BLEU (Papineni et al., 2002).

4.1 Experimental Setup

The parallel training corpus include 1.25M Chinese-English sentence pairs.[1] We parse the Chinese sentences with Stanford Parser (Klein and Manning, 2003) into projective dependency trees, obtain word alignment by running GIZA++ (Och and Ney, 2003) in both directions and applying "grow-diag-final" refinement (Koehn et al., 2003), and train a 4-gram language model by SRI Language Modeling Toolkit (Stolcke, 2002) with Kneser-Ney smoothing on the Xinhua portion of the Gigaword corpus.

We take NIST MT Evaluation test set 2002 as our development set, 2003 (MT03), 2004 (MT04) and 2005 (MT05) as our test sets, evaluate the quality of translations by *case insensitive* NIST BLEU-4 metric[2], tune the feature weights by Max-BLEU strategy with MERT (Och, 2003), and check the statistical difference between the systems with significance test (Collins et al., 2005).

4.2 Systems

We take "Moses-Chart" of Moses[3] (Koehn et al., 2007) as hierarchical phrase-based model baseline. In our experiments, we use the default settings.

Both the dependency-to-string baseline and our augmented model employ the same settings as those of Xie et al. (2011), with the beam threshold, beam size and rule size are set to 10^{-3}, 200 and 100 respectively. And both systems employ bilingual phrases with length ≤ 7 extracted by Moses.

4.3 Experiment results

Table 1 shows the results of the BLEU scores of the three systems. Where "dep2str" and "dep2str-aug" denote dependency-to-string model baseline and our augmented dependency-to-string model, respectively. As we can see, "dep2str" shows better performance (+0.31 BLEU on average) than "Moses-Chart"

[1] From LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

[2] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v11b.pl

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

System	Rule#	MT03	MT04	MT05	Average
Moses-Chart	116.4M	34.65	36.47	34.39	35.17
dep2str	37M+32.5M	34.92	36.82	34.71	35.48
dep2str-aug	37M+32.5M	**35.66*****(+0.74)**	**37.61*****(+0.79)**	**35.74*****(+1.03)**	**36.33 (+0.85)**

Table 1: Statistics of the extracted rules and BLEU scores (%) on the test sets of the three systems. Where "37M+32.5M" denotes 37M rules and 32.5M bilingual phrases. And "*" indicates *dep2str-aug* are statistically better than *dep2str* with $p < 0.01$.

Source: 桑帕约 对 葡 中 两 国 在 世博会 事务 方面 的 合作 寄予 厚望 。

Reference 1: *Sampaio has placed high hopes on the Portuguese-Sino cooperation in the World Expo.*

Reference 2: *Sampaio expressed his high expectations on the Sino-Portuguese cooperation in the work of the world exposition.*

Moses-Chart: Sampaio on cooperation between the two countries in the world expo affairs Portugal and China places great .

Dep2Str: President placed great cooperation between Portugal and China , the two countries in the World Expo affairs .

Dep2Str-aug: Sampaio placed high expectations of the Portuguese - Chinese cooperation in World Expo affairs .

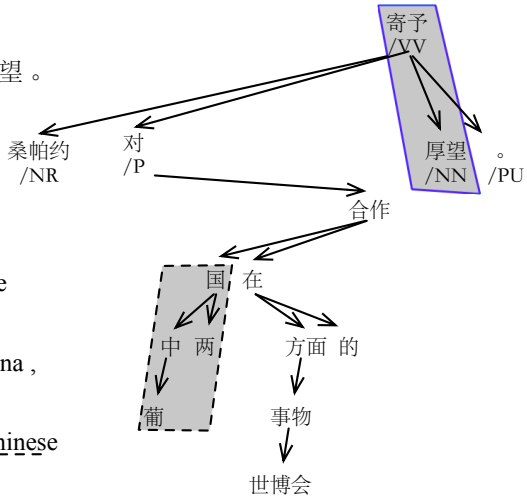


Figure 5: Translation examples of "Moses-Chart", "dep2str" and "dep2str-aug". The line border shadow denotes the phrases successfully captured by "dep2str-aug".

and is a strong baseline."Dep2str-aug" gains significant improvements of +0.74, +0.79 and +1.03 BLEU points over "dep2str" on the test sets, respectively.

Additionally, we compare the actual translations generated by "Moses-Chart", "dep2str" and "dep2str-aug". Figure 5 shows the translations of these three systems on a sentence of MT05. The source sentence holds a common sentence pattern in Chinese, which is composed of a proper noun, a verb, a noun and a prepositional phrases (corresponding to the top level of the dependency tree on the right). However, the preposition phrase related to "对" holds nine words, thus the simple pattern becomes a long distance dependency that challenges SMT systems. Limited by the phrase-based rules, "Moses-Chart" fails to capture the sentence pattern and outputs a messy translation with little sense. "Dep2str", resorting to HDR rules, successfully captures the pattern and outputs a translation with correct reordering, but it is still hard to understand. With the help of augmented HDR rules, "dep2str-aug" captures both the sentence pattern and non-syntactic phrase "寄予 厚望" and gives an translation with good adequacy and fluency.

These results reveal the merits of our augmented dependency-to-string model at handling both long distance reordering (with HDR) and local reordering (with fixed and floating structures), which is promising for translating language pairs that are syntactically divergent.

5 Conclusion and Future Work

In this paper, we propose an augmented dependency-to-string model to address the non-syntactic phrase coverage problem for dependency-to-string model. To this purpose, we make two important augmentations to the dependency-to-string model. First, we propose an compact representation to combine both head-dependent relation and the fixed and floating structures into translation rules. Second, in decoding we build "on the fly" new translation rules from the compact translation rules and incorporate non-syntactic phrases into translations. By this way, we can combine the merits of both head-dependents relation at handling long distance reordering and bilingual phrases at handling local reordering. Large-

scale experiments show that our augmented dependency-to-string model gains significant improvements over the dependency-to-string model.

In the future work, we would like to incorporate semantic knowledge such as typed dependencies and WordNet[4] (Miller, 1995) so as to better direct the process of translation.

Acknowledgments

The authors were supported by National Nature Science Foundation of China (Contract 61370130 and 61379086). Liu was partially supported by the Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the CNGL at Dublin City University. We sincerely thank the anonymous reviewers for their careful review and insightful suggestions.

References

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the ACL 2005*, pages 531–540, Ann Arbor, Michigan, June.

Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of ACL 2005*, pages 271–279.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA, May 2 - May 7.

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of ACL 2007*, pages 144–151, Prague, Czech Republic, June.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, pages 66–73.

Dan Klein and Christopher D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *In Advances in Neural Information Processing Systems 15 (NIPS)*, pages 3–10. MIT Press.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Edmonton, Canada, July.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL 2005: Interactive Poster and Demonstration Sessions*, pages 177–180.

Dekang Lin. 2004. A path-based transfer model for machine translation. In *Proceedings of Coling 2004*, pages 625–630, Geneva, Switzerland, Aug 23–Aug 27.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of ACL 2006*, pages 609–616, Sydney, Australia, July.

Fandong Meng, Jun Xie, Linfeng Song, Yajuan Lü, and Qun Liu. 2013. Translation with source constituency and dependency trees. In *Proceedings of EMNLP 2013*, pages 1066–1076, Seattle, Washington, USA, October.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL 2002*, pages 295–302, Philadelphia, Pennsylvania, USA, July.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

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

[4]http://wordnet.princeton.edu

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL 2002*, pages 311–318, Philadelphia, Pennsylvania, USA, July.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of ACL 2005*, pages 271–279.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL 2008: HLT*, pages 577–585, Columbus, Ohio, June.

Andreas Stolcke. 2002. Srilm - an extensible language modeling toolkit. In *Proceedings of ICSLP, volume 30*, pages 901–904.

Jun Xie, Haitao Mi, and Qun Liu. 2011. A novel dependency-to-string model for statistical machine translation. In *Proceedings of EMNLP 2011*, pages 216–226, Edinburgh, Scotland, UK., July.

Soft Dependency Matching for Hierarchical Phrase-based Machine Translation

Hailong Cao[1], Dongdong Zhang[2], Ming Zhou[2] and Tiejun Zhao[1]
[1]Harbin Institute of Technology, Harbin, P.R. China
[2]Microsoft Research Asia, Beijing, P.R. China
{hailong, tjzhao}@mtlab.hit.edu.cn
{Dongdong.Zhang, mingzhou}@microsoft.com

Abstract

This paper proposes a soft dependency matching model for hierarchical phrase-based (HPB) machine translation. When a HPB rule is extracted, we enrich it with dependency knowledge automatically learnt from the training data. The dependency knowledge not only encodes the dependency relations between the components inside the rule, but also contains the dependency relations between the rule and its context. When a rule is applied to translate a sentence, the dependency knowledge is used to compute the syntactic structural consistency of the rule against the dependency tree of the sentence. We characterize the structure consistency by three features and integrate them into the standard SMT log-linear model to guide the translation process. Our method is evaluated on multiple Chinese-to-English machine translation test sets. The experimental results show that our soft matching model achieves 0.7-1.4 BLEU points improvements over a strong baseline of an in-house implemented HPB translation system.

1 Introduction

HPB model (Chiang, 2007) is widely used and has consistently delivered state-of-the-art performance. This model extends the phrase-based model (Koehn et al., 2003) by using the formal synchronous grammar to well capture the recursiveness of language during translation. In a formal synchronous grammar, the syntactic unit could be any sequence of contiguous terminals and non-terminals, which may not necessarily satisfy the linguistic constraints. HPB model is powerful to cover any format of translation pairs, but it might introduce ungrammatical rules and produce poor quality translations.

To generate grammatical translations, lots of syntax-based models have been proposed by Galley et al. (2004), Liu et al. (2006), Huang et al. (2006), Mi et al. (2008), Shen et al. (2008), Xie et al. (2011), Zhang et al. (2008), etc. In these models, the syntactic units should be compatible with the syntactic structure of either the source sentence or the target sentence. These approaches can generate more grammatical translations by capturing the structural difference between language pairs. However, these models need special efforts to capture non-syntactic translation knowledge to improve the translation performance.

It is desired to combine the advantages of syntax-based models and the HPB model (Stein et al., 2010). There has been much work trying to improve HPB model by incorporating syntax information. Marton and Resnik (2008) leverage linguistic constituents to constrain the decoding softly. Some work go further to augment the non-terminals in HPB rules with syntactic tags which depend on the syntactic structure covered by the non-terminals (Zollmann and Venugopal, 2006; Chiang, 2010; Li et al., 2012; Huang et al., 2013). For example, given below HPB rules (1-4), the source non-terminal X could be refined into NP or PP as shown in rules (5-8) respectively.

(1) <借 了 X, borrowed X> (2) <借 了 X, lent X>
(3) <X_1 借 了 X_2, borrowed X_2 X_1> (4) <X_1 借 了 X_2, X_1 borrowed X_2>

(5) <借了 NP, borrowed X> (6) <借了 NP, lent X>
(7) <PP 借了 NP, borrow X_2 X_1> (8) <NP 借了 NP, X_1 lent X_2>

Although augmenting the non-terminals with syntactic tags in these methods achieved better results for HPB model, they have limitations that the syntax information on the non-terminals are not discrim-

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2227–2236, Dublin, Ireland, August 23-29 2014.

inative enough due to the limited context covered by the HPB rule. For example, rule (5) and (6) are still not discriminative when translating below two sentences (9) and (10).

(9) 我向他借了一本书(I borrowed a book from him) (10) 我借了一本书给他(I lent a book to him)

where the common phrase "借了一本书" appear in both sentences. Obviously, although rule (5) and (6) share same source sides, rule (5) can only be applied to the translation of sentence (9) and rule (6) to sentence (10). Otherwise, inappropriate application will lead to wrong translations. Rule (5) and (6) are not discriminative due to no consideration of their outside context during the translation.

Motivated by such observation, we proposed an alternative approach, called soft dependency matching model, to incorporate into each HPB rule the source syntactic dependencies connecting the contents inside the rule with the context outside the rule. The dependency knowledge associated with HPB rules is automatically learnt from bilingual training corpus. They make HPB rules discriminative according to global context.

Figure 1. Dependency information associated with two rules. LC and RC mean the source context on the left and right of the rule respectively.

Figure 1 shows two rules associated with different dependencies. The first one is applicable to the case when some word on the left side depends on the word "借" in the rule, and the second one is applicable to the case when the word "借" in the rule depends on some word on the right side.

During SMT decoding, first we parse the source sentence to get the dependency tree. When a HPB rule is applied to translate the sentence, we calculate structural consistency between the dependency knowledge associated with the rule and dependency tree structure of the source sentence. The consistency degree is integrated into the SMT log-linear model as features to encourage syntactic hypotheses and penalize the hypotheses violating syntactic constraints.

Compared with previous work that incorporate syntax knowledge into HPB model, the advantage of our soft dependency matching model is:

- It not only captures the dependency relations between the components inside the rule, but also models the dependency relations between the rule and its context from a global view.
- Without increasing the amount of rules or the searching space, our model can capture the syntactic variation for all of the rules (syntactic or non-syntactic, well-formed or ill-formed).
- Our model can take advantage of the dependency knowledge on both terminals and non-terminals.

We evaluate the performance of our soft dependency matching model on Chinese-to-English translation task. Experimental results show that our method can achieve the improvements of 0.7-1.4 BLEU points over the baseline HPB model on multiple NIST MT evaluation test sets.

2 Related Work

Ever since the invention of phrase-based model, a lot of efforts have been made to incorporate linguistic syntax. Cherry(2008) and Marton and Resnik (2008) leverage linguistic constituent to constrain the decoding softly. In their methods, a translation hypothesis gets an extra credit if it respects the parse tree but may incur a cost if it violates a constituent boundary. The soft constrain based methods achieved promising results on various language pairs. One problem of these methods is that exactly matching syntactic constraints cannot always guarantee a good translation, and violating syntactic structure does not always induce a poor translation. It could be more reasonable if the credit and penalty is learnt from the parallel training data. In this work, we learn this kind of constrain knowledge directly from the syntactic structures over the training corpus.

Xiong et al. (2009) present a method that automatically learns syntactic constraints from training data for the ITG based translation (Wu, 1997; Xiong et al., 2006). They utilize the syntactic constraints to estimates the extent to which a span is bracketable. Though the effect was demonstrated on the ITG based model, the method is also applicable to the HPB model. The main difference between Xiong et al. (2009) and our work is that we try to estimate the structural consistency of each rule

against the source syntax tree. For rules which are same in the source side but different in the target side, our method will distinguish the inconsistency degree for different rules. While, for such rules, Xiong et al. (2009) will give a same score which will be used to compete with rules in other spans.

More recently, Huang et al. (2013) associate each non-terminal with the distribution of tags that is used to measure the consistency of syntactic compatibility of the translation rule on source spans. Our work is similar to Huang et al. (2013) since we also represent the syntactic variation of translation rules in the form of distribution. The main difference is that they annotate non-terminals with head POS tags while we use dependency triples (over both terminals and non-terminals) to explicitly represent both the dependency relations inside the rule, and that between the rule and its context.

Both above related work and our work need parse the source sentence to get syntactic context before decoding. There are also some methods incorporating syntax information without the need of online parsing the source sentences (Zollmann and Venugopal, 2006; Shen et al, 2009; Chiang, 2010). They parse the training data to label the non-terminals with syntactic tags. During the bottom-up decoding, the tags are used to model the substitution of non-terminals in a soft way (Shen et al, 2009; Chiang, 2010) or in a hard way (Zollmann and Venugopal, 2006).

Gao et al. (2011) derive soft constraints from the source dependency parsing for the HPB translation. They focus on the relative order of each dependent word and its head word after translation, while our method models whether the dependency information of a rule matches the context or not.

Our work utilizes contextual information around translation rules. In this sense, it is similar to He et al. (2008) and Liu et al. (2008). The main difference between their work and our work is that they leverage lexical context for rule selection while we focus on the syntactic contextual information.

3 Hierarchical Phrase based Machine Translation

Our model proposed in this paper is an extension of the HPB model (Chiang, 2007). Formally, HPB model is a weighted synchronous context free grammar. It employs a generalization of the standard plain phrase extraction approach in order to acquire the synchronous rules of the grammar directly from word-aligned parallel text. Rules have the form of:

$$X \rightarrow < \gamma, \alpha, \sim >$$

where X is a nonterminal, γ and α are both strings of terminals and non-terminals from source and target side respectively, and $\sim$ is a one-to-one correspondence between nonterminal occurrences in γ and α. Associated with each rule is a set of feature functions with the form $f_i(\gamma, \alpha)$. These feature functions are combined into a log-linear model. When a rule is applied during SMT decoding, its score is calculated as:

$$\sum_i \lambda_i \cdot f_i(\gamma, \alpha)$$

where λ_i is the weight associated with feature function $f_i(\gamma, \alpha)$. The feature weights are typically optimized using minimum error rate training algorithm (Och, 2003).

4 Soft Dependency Matching Model

In order to incorporate syntactic knowledge to refine both the word ordering and word sense disambiguation for HPB model, we propose a soft dependency matching model (SDMM). It extends HPB rule into a form which is named as SDMM rule:

$$X \rightarrow < \gamma, \alpha, \sim, \mathrm{RDT} >$$

where RDT(rule's dependency triples) is a set of dependency triples defined on source string γ. Each element in RDT is a triple representing dependency knowledge in the form:

$$\{m\text{-}h\text{-}l\}$$

where *m* and *h* are the dependent and head respectively, *l* is the label of the dependency relation type. *m* and *h* could be any of terminals, non-terminals, LC and RC, where LC denotes the left context and RC the right context.

In the following two sub-sections, we will explain the details of SDMM rule extensions for both plain phrases (i.e., there are no non-terminals in both γ and α) and hierarchical rules (i.e., there are at

least one non-terminal in both γ and α) respectively. For simplicity, we ignore the correspondence ~ in the representations of both HPB rules and SDMM rules.

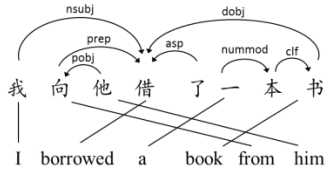


Figure 2: An illustration of a dependency parse tree for the source side of a word-aligned parallel sentences pair.

4.1 SDMM Over Plain Phrase Rules

Figure 2 illustrates a parallel sentence together with word alignments and source dependency parse tree, from which we can extract the phrase pairs of HPB rules like:

(11) < 一 本 书, a book > (12) < 借 了 一 本 书, borrowed a book >

By incorporating syntactic knowledge, we can extend these HPB rules into SDMM rules as shown in Figure 3(a) and Figure 3(b) respectively.

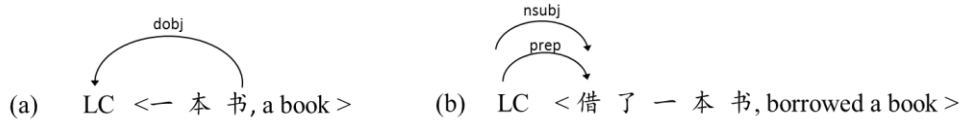


Figure 3: An illustration of two phrase pairs annotated with a set of dependency triples.

Formally, the RDT corresponding to phrase pair (11) is {书-LC-dobj}. The RDT corresponding to phrase pair (12) is {LC-借-nsubj, LC-借-prep}.

Now we describe how to build the RDT when a phrase pair is extracted from a sentence pair during the training step. First, we initialize RDT to be empty. Then, for each dependency triple "*m*-*h*-*l*" in the parse tree of the source sentence, if either *m* or *h* is covered by the source phrase in the rule, we add it to RDT. However, if both *m* and *h* are covered by the source phrase, we will ignore it because it holds less syntactic information beyond HPB rule itself. For example, the dependency triple "一-本-nummod" is excluded from RDT for both phrase pair (11) and phrase pair (12). In addition, we do not add the dependency triple "*m*-*h*-*l*" into RDT if both *m* and *h* are not contained in source phrase, because it is not related to phrase pair at all. The dependency triple "他-向-pobj" is such a case for both phrase pair (11) and phrase pair (12).

Finally, we normalize the word in RDT that is not covered by the source phrase with either LC (stands for the left context) or RC (stands for the right context) according to its relative position to the source phrase. For example, in the RDT for phrase pair (11), we normalize "书-借-dobj" as "书-LC-dobj" since the word "借" is not covered by the source phrase and it is treated as left context.

Note that for each context word outside the source phrase, we only record whether it is on the left or on the right of phrase. We do not further consider its lexical form and its distance to the source phrase. For example, in the two dependency triples in Figure 3(b), both the dependent word "我" and "向" are normalized into LC. In this way, we can generalize the dependency triples in RDT and alleviate the data sparseness problem. In fact, there might be duplicated dependency triples for a phrase pair. In this case, we only keep one of them.

4.2 SDMM over Hierarchical Rules

Hierarchical rules are usually generated by substituting sub-phrases with non-terminals from plain phrase pairs. For example, given the parallel sentence and the two phrase pairs in Section 4.1, we can get a hierarchical rule like:

<借 了 X, borrowed X>

To extend hierarchical rules into SDMM rules, we add dependency information to source terminals or non-terminals in RDT. Figure 4 shows an example representing an SDMM rule:

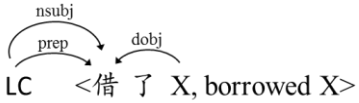


Figure 4: An illustration of a hierarchical rule annotated with a set of dependency triples.

The generation of SDMM rules over hierarchical rules is similar to that of plain phrase rules. The only difference lies in processing the non-terminals, whose dependencies are inferred from the words they covered. For example, the RDT of the above SDMM rule would be:{LC-借-nsubj, LC-借-prep, X-借-dobj}

Similarly, any dependencies over two terminals contained in the source rule are not included in RDT, and dependencies inferred from same non-terminals are excluded as well. In addition, dependencies between two non-terminals are ignored.

4.3 SDMM Rule Composing

A same HPB rule (either plain phrase pair or a hierarchical rule) can be extracted from different bilingual sentences. Therefore, the same HPB rule could be extended into multiple SDMM rules. For example, given a parallel sentence pair shown in Figure 5,

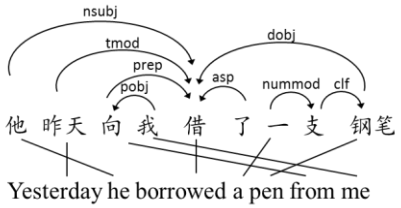


Figure 5: An example of a dependency tree over the source sentence together with the word-aligned target sentence.

we might get a SDMM rule as shown in Figure 6. Compared to the SDMM rule in Figure 4, there is an additional dependency triple "LC-借-tmod" in RDT.

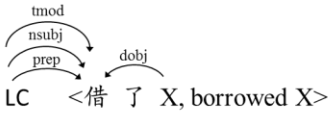


Figure 6: An illustration of dependency triples associated to a hierarchical rule.

Intuitively, we can process SDMM rules independently although they share the same information of HPB rules. However, this will exacerbate the data sparseness problem and make the computation inefficient due to dramatically increased model size. An alternative way is only to keep the most frequent RDT information for the same HPB rules. Though this can get a very concise model, a lot of useful syntactic information might be lost.

We propose a balanced composing method to make a trade-off between knowledge representation and computation efficiency of SDMM rules. Suppose there are more than one SDMM rules with different RDT_i but the same HPB rule, we compose them by the union and get the new form of RDT as:

$$\mathrm{RDT} = \bigcup_i \mathrm{RDT_i}$$

In addition, we record the frequency of HPB rule as well as that of each dependency triple in RDT as:

$$\#(\mathrm{X} \rightarrow < \gamma, \alpha, \sim >) \ , \ \#(t_i, \mathrm{X} \rightarrow < \gamma, \alpha, \sim >)$$

where $\#(\mathrm{X} \rightarrow < \gamma, \alpha, \sim >)$ is the number of times that HPB rule $\mathrm{X} \rightarrow < \gamma, \alpha, \sim >$ is extracted from the training data, and $\#(t_i, \mathrm{X} \rightarrow < \gamma, \alpha, \sim >)$ is the frequency that t_i and $\mathrm{X} \rightarrow < \gamma, \alpha, \sim >$ co-occur. For example, suppose SDMM rules in Figure 4 and Figure 6 occurs 9 and 1 times respectively, we can compose them into the form as shown in Figure 7.

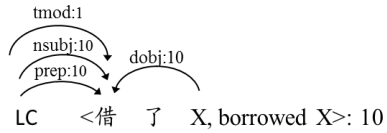


Figure 7: Composed form of the dependency annotation of a rule. The integers following the colons denote occurring times.

Therefore, the composed SDMM rule will be represented by the original HPB rule <借 了 X, borrowed X> together with RDT and its frequency information shown in Table 1.

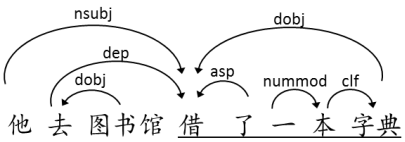
RDT	#
{ LC-借-tmod,	1
LC-借-nsubj,	10
LC-借-prep,	10
X - 借-dobj }	10

Table 1. The RDT and its frequency information of a composed SDMM rule.

4.4 Consistency of SDMM Rules

So far we have described how to enrich a rule with RDT in the training step. Now we introduce how to use the RDT of each rule to guide the translation process.

In the decoding, we parse the source sentence to get the dependency parse tree as shown in Figure 8. When we apply a rule to get a partial translation for a span, we also extract a set of dependency triples based on the parse tree in the exact same way that is used in the training step. We denote this by CDT (context dependency triples). Suppose the rule <借 了 X, borrowed X> is applied to translate the underlined span in Figure 8, then the CDT for the rule is: {LC-借-nsubj, LC-借- dep, X-借-dobj}.



Ref: He went to the library to borrow a dictionary

Figure 8: A sentence to be translated and its dependency parse tree.

In order to evaluate whether a SDMM rule is applicable to translate a sentence or not from the syntactic view, we model the structural consistency of SDMM rule against source dependency tree by calculating the matching degree between RDT and CDT. The example in Figure 9 illustrates how we compute the matching degree between the SDMM rule in Figure 7 and CDT over the source dependency tree in Figure 8. We estimate the matching degree based on three sets including the relative complement set of CDT in RDT, the intersection set of RDT and CDT, and the relative complement set of RDT in CDT.

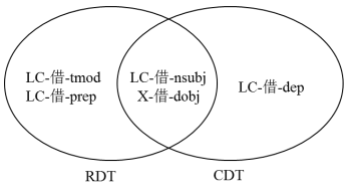


Figure 9: Three different sets of dependency triples to model the structural consistency of syntactic matching.

The statistics over above three sets are leveraged to design three features which are incorporated into SMT log-linear model to encourage and penalize various syntactic motivated hypotheses. The first feature is called as the lost dependency triple feature f_l. It is calculated based on the set RDT\CDT as:

$$f_l = \sum_{t \in RDT \setminus CDT} \delta(\#(t, X \rightarrow < \gamma, \alpha, \sim >) == \#(X \rightarrow < \gamma, \alpha, \sim >))$$

where δ is the indicator function whose value is one if and only if the condition is true, otherwise its value is zero. The motivation of f_l is that: if a dependency triple which always co-occur with the HPB rule is not observed in CDT, it indicates the current SDMM rule may mismatch with the source sentence and therefore we need to penalize its application. In Figure 9, "LC-借-prep" is such a dependency triple. However, for the less frequent dependency triples in RDT such as "LC-借-tmod" in Figure 8, there is no penalty on it although it is not found in CDT.

The second feature is the unexpected dependency triple feature f_u, which is computed as :

$$f_u = |\mathrm{CDT} \backslash \mathrm{RDT}|$$

This feature is the number of dependency triples in CDT that never co-occur with the rule in the training data. In Figure 9, "LC-借-dep" is such a case. Intuitively, the higher the value f_u is, the higher inconsistency degree is, because it means that many dependency triples in CDT are never observed in the training corpus. We should discourage the application of the corresponding SDMM rule.

The third feature is the matched dependency triple feature f_m which is calculated based on RDT∩CDT. It is directly used to model the structural consistency over all the dependency triples in RDT∩CDT for the application of HPB rule $\mathrm{X} \rightarrow < \gamma, \alpha, \sim >$. Formally, f_m is defined as the sum of log probability of each dependency triple in RDT∩CDT conditioned on the HPB rule:

$$f_m = \sum_{t \in \mathrm{RDT} \cap \mathrm{CDT}} \log(P(t|\mathrm{X} \rightarrow < \gamma, \alpha, \sim >))$$

where $\mathrm{P}(t|\mathrm{X} \rightarrow < \gamma, \alpha, \sim >)$ is the probability of a dependency triple t associated to a HPB rule $\mathrm{X} \rightarrow < \gamma, \alpha, \sim >$. We estimate it based on the relative frequency and experimentally use the adding 0.5 smoothing.

5 Experiments

5.1 Experimental Settings

Our baseline is the re-implementation of the Hiero system (Chiang, 2007). When our soft dependency matching model is integrated, the HPB rule is extended into the form of $\mathrm{X} \rightarrow < \gamma, \alpha, \sim, \mathrm{RDT} >$ and the score is calculated by:

$$\sum_i \lambda_i \cdot f_i(\gamma, \alpha) + \lambda_l \cdot f_l(\gamma, \alpha, \mathrm{RDT}) + \lambda_u \cdot f_u(\gamma, \alpha, \mathrm{RDT}) + \lambda_m \cdot f_m(\gamma, \alpha, \mathrm{RDT})$$

where the additional three features are defined in Section 4.3, λ_l, λ_u and λ_m are corresponding feature weights.

We test our soft dependency matching model on a Chinese-English translation task. The NIST06 evaluation data was used as our development set to tune the feature weights, and NIST04, NIST05 and NIST08 evaluation data are our test sets. We first conduct experiments by using the FBIS parallel corpus, and then further test the performance of our method on a large scale training corpus.

Word alignment is performed by GIZA++ (Och and Ney, 2000) in both directions with the default setting. 4-gram language model is trained over the Xinhua portion of LDC English Gigaword Version 3.0 and the English part of the bilingual training data. Feature weights are tuned with the minimum error rate training algorithm (Och, 2003).Translation performance is measured with case-insensitive BLEU4 score (Papineni et al., 2002).

All the Chinese sentences in the training set, development set and test set are parsed by an in-house developed dependency parser based on shift-reduce algorithm (Zhang and Nivre, 2011). There are 45 named grammatical relations plus a default relation representing unknown cases. The detailed descriptions about dependency parsing are explained in Chang et al. (2009).

5.2 Experimental Results on FBIS Corpus

We first conduct experiments by using the FBIS parallel corpus to train the model of both the baseline and the soft dependency matching model. Table 2 shows the statistics of FBIS corpus after the preprocessing.

	#sentences	#words
Chinese	128,832	3,016,570
English	128,832	3,922,816

Table 2. The statistics of FBIS corpus

The evaluation results over FBIS corpus are reported in Table 3. The first row shows the results of baseline, the next three rows show the effect of three features respectively and the last row gives the result when all features are integrated together. Based on Table 3, we can see that each individual feature improves the performance. Among all integrated features, the third feature f_m is the most effective one. The best performance is achieved when using all three features, where we get 1.4, 0.9 and 1.2 BLEU points improvements respectively over the baseline on three test sets.

	NIST04	NIST05	NIST08
Baseline	33.53	32.97	25.08
Baseline+f_l	34.59	33.44	25.69
Baseline+f_u	34.48	33.59	25.51
Baseline+f_m	34.73	33.74	25.76
Baseline+f_l+f_u+f_m	**34.96**	**33.91**	**26.28**

Table 3. Translation performance over BLEU% when models are trained on the FBIS corpus.

5.3 Experimental Results on Large Scale Corpus

To further test the effect of our soft dependency matching model, we use a large scale corpus released by LDC. The catalog number of them is LDC2003E07, LDC2003E14, LDC2005T06, LDC2005T10, LDC2005E83, LDC2006E26, LDC2006E34, LDC2006E85 and LDC2006E92. There are 498K sentence pairs, 12.1M Chinese words and 13.8M English words. Table 4 summarizes the translation performance on the large scale of corpus. Our model is still effective when we train the translation system on large scale data. We get 1.3, 0.7 and 1.0 BLEU point improvements over the baseline on three test sets respectively, which shows that our method can consistently improve HPB system over different sized training corpus.

	NIST04	NIST05	NIST08
Baseline	38.72	37.59	29.03
Baseline+f_l+f_u+f_m	**40.00**	**38.34**	**30.06**

Table 4. Translation performance over BLEU% when models are trained on a large scale parallel corpus.

5.4 Decoding Cost

Incorporating syntax can improve the translation performance, but it might increase the SMT decoding complexity. One advantage of our method is that it does not increase the amount of translation rules, so the searching space is not enlarged. Table 5 shows the decoding time comparison with the baseline when models are trained on the FBIS corpus. The average decoding time per sentence is only increased by about 12% due to the parsing of source sentences and the computation of the features. We believe that this is acceptable given the performance gain.

	NIST04	NIST05	NIST08
Baseline	0.67sec	0.78sec	0.50sec
Baseline+f_l+f_u+f_m	0.88sec	0.87sec	0.56sec

Table 5. The average decoding time per sentence, measured in second/sentence.

6 Conclusion and Future Work

We proposed a soft dependency matching model for HPB machine translation. We enrich the HPB rule with dependency knowledge learnt from the training data. The dependency knowledge allows our model to capture the both the dependency relations inside the rule and the dependency relations between the rule and its context from a global view. During decoding, the syntax structural consistency of rules against source dependency tree is calculated and converted into SMT log-linear model fea-

tures to guide the translation process. The experimental results show that our soft matching model achieves significant improvements over a strong baseline of an in-house implemented HPB system.

In future work, there is much room to improve the performance via our method. First, we can discriminatively learn the contribution of the dependency knowledge of each rule based on the training data. Second, we can go beyond the current "bag of dependency triples" representation by composing them hierarchically to capture deep syntactic information. Third, section 2 has discussed the theoretical difference with related work on adding source syntax into the HPB model, we are interested in empirically comparing our method with them and combining it with them to get further improvement.

Acknowledgments

We thank anonymous reviewers for insightful comments. The work of Hailong Cao is sponsored by Microsoft Research Asia Star Track Visiting Young Faculty Program. The work of HIT is also funded by the project of National Natural Science Foundation of China (No. 61173073) and International Science & Technology Cooperation Program of China (No. 2014DFA11350).

Reference

Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009. Discriminative Reordering with Chinese Grammatical Relations Features. In Proceedings of *NAACL* Workshop on SSST.

Colin Cherry. 2008. Cohesive Phrase-based Decoding for Statistical Machine Translation. In Proceedings of *ACL*.

David Chiang. 2007. Hierarchical Phrase-based Translation. Computational Linguistics, 33(2):201–228.

Yang Gao, Philipp Koehn, and Alexandra Birch. 2011. Soft Dependency Constraints for Reordering in Hierarchical Phrase-Based Translation. In Proceedings of *EMNLP*.

Zhongjun He, Qun Liu, Shouxun Lin. 2008. Improving Statistical Machine Translation using Lexicalized Rule Selection. In Proceedings of *COLING*.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended-domain of locality. In Proceedings of *AMTA.*

Zhongqiang Huang, Martin Čmejrek, and Bowen Zhou. 2010. Soft Syntactic Constraints for Hierarchical Phrase-based Translation Using Latent Syntactic Distributions. In Proceedings of *EMNLP*.

Zhongqiang Huang, Jacob Devlin, and Rabih Zbib. 2013. Factored Soft Syntactic Contraints for Hierarchical Machine Translation. In Proceedings of *EMNLP*.

Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In Proceedings of *ACL*.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In Proceedings of *ACL*.

Philipp Koehn, Franz Josef Och, Daniel Marcu. 2003. Statistical phrase based translation. In Proceedings of *NAACL.*

Junhui Li, Zhaopeng Tu, Guodong Zhou, and Josef van Genabith. 2012. Using Syntactic Head Information in Hierarchical Phrase-based Translation. In Proceedings of *WMT*.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree to-string alignment template for statistical machine translation. In Proceedings of *ACL.*

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In Proceedings of *ACL*.

Haitao Mi and Liang Huang. 2008. Forest-based translation rule extraction. In Proceedings of *EMNLP*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In Proceedings of *ACL*.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In Proceedings of *ACL.*

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

Daniel Stein, Stephan Peitz, David Vilar, and Hermann Ney. 2010. A Cocktail of Deep Syntactic Features for Hierarchical Machine Translation. In Conference of the Association for Machine Translation in the Americas.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. Computational Linguistics, 23(3):377–404.

Jun Xie, Haitao Mi and Qun Liu. 2011. A Novel Dependency-to-String Model for Statistical Machine Translation. In Proceedings of *EMNLP*.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum Entropy Based Phrase Reordering Model for Statistical Machine Translation. In Proceedings of *ACL*.

Deyi Xiong, Min Zhang, Aiti Aw, Haizhou Li. 2009. A Syntax-Driven Bracketing Model for Phrase-Based Translation. In Proceedings of *ACL*.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In Proceedings of *NAACL* Workshop on SMT.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In Proceedings of *ACL*.

Yue Zhang and Joakim Nivre. 2011. Transition-based Dependency Parsing with Rich Non-local Features In Proceedings of *ACL*.

Using Spreading Activation to Evaluate and Improve Ontologies

Rónan Mac an tSaoir
Watson Group, IBM Ireland
ronanate@ie.ibm.com

Abstract

In this paper, we explore the relationship between the human-encoded semantics of ontologies and their application to natural language processing (NLP) tasks, such as word-sense disambiguation (WSD), for which such ontologies may not have been originally designed. We present a method for assessing the semantic content of an ontology with respect to a target domain, by spreading activation over a graph that represents instances of ontology concepts and relationships, in domain text. Our proposed method has several advantages beyond existing ontology metrics. By identifying bias or imbalance in the ontology, we can suggest target areas for improvement, and simultaneously facilitate the automated optimisation of the graph for use in the chosen NLP task. On applying this method to the Unified Medical Language System (UMLS) ontology, we significantly outperformed existing graph-based methods for WSD in biomedical NLP (0.82 accuracy). The subsequent introduction of a fall-back mechanism, using word-sense probability, achieved state of the art for unsupervised biomedical WSD (0.89 accuracy).

1 Introduction

Although ontologies do encode human knowledge, the degree to which these artefacts represent the entire scope of semantics in a target domain is difficult to quantify. Since few ontologies offer large enough scope to cater for an entire domain in natural language, merging of multiple ontologies is often necessary (Noy, 2004). This further compounds the problem of assessing the semantic relevance of the merged resource. The collective semantics in multiple source ontologies can often overlap inconsistently, and negotiation of meaning so that the associated set of concepts and relationships in the ontology remains balanced, is critical. The merging process is usually reserved for domain experts, who focus on ontology portions in which they specialise. It's generally a case of painstakingly mapping individual concepts between component data sets, to ensure semantic integrity (Jiménez et al, 2012). Coordinating collaborative ontology editing and merging is a related and well-known problem (Jiménez et al, 2011).

Existing ontology metrics generally focus on structural and logical semantics (Sicilia et al, 2012). Assessing how closely ontologies match the semantics of natural language text, or identifying specific portions of an ontology which require further development, are more difficult tasks. We have identified a robust method for this assessment. This method involves static analysis of a graph representing ontology instances and inter-concept relationships, to address apparent imbalances that hinder spreading activation in the graph. When accuracy and relevance for the task improves, the modified graph or activation strategy identifies portions of interest for further development. Many ontologies used in NLP today are not designed for this (Guarino et al, 2009), and a flexible, automatic evaluation method is useful.

We focused on the Unified Medical Language System (UMLS) as a typical ontology (NLM, 2013), displaying many of the problems associated with use of ontologies in NLP, including merged terminology, strongly overlapping semantic categories, inconsistent levels of structural depth, as well as inconsistent coverage of associated instance data (Pisanelli et al, 1998). We chose to assess this ontology with respect to word sense disambiguation (WSD), which is commonly accepted to be one of the most difficult tasks in NLP (Navigli, 2009). We used the MSH-WSD corpus for testing purposes, which commonly used in assessing methods for biomedical WSD (Jimeno Yepes and Aronson, 2012; McInnes et al, 2011; Gad el Rab et al, 2013). Using node-centric graph metrics, we identified portions of the ontology which were not conducive to WSD via spreading activation. After appropriately modifying the activation strategy, we achieved state of the art performance in graph-based biomedical WSD (0.82).

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2237–2248, Dublin, Ireland, August 23-29 2014.

2 Background

2.1 Ontologies

An ontology, in computer science, is defined as an 'explicit specification of a shared conceptualization' (Gruber, 1993), where a conceptualization may be some subset of real-world semantics, with respect to the requirements for a given task. It can contain concepts or classes of object, object properties, and inter-concept relationships, as well as instances of these in the target domain. Such structured resources facilitate the sharing and re-use of domain knowledge, and are invaluable for NLP applications. A primary example of such a resource is the UMLS, provided by the National Library of Medicine (NLM, 2013). The data set consists of a large lexicon, including millions of instance surface forms, in conjunction with an ontology of concepts and inter-concept relationships in the medical domain. It is composed of 139 different source ontologies or terminologies, each of which have their own labels, descriptions and semantic perspective (e.g. FMA[2] for the body, and RXNORM[3] for drugs, as well as more general ontologies like SNOMED[4]). An example ontology is shown in Figure 1 below.

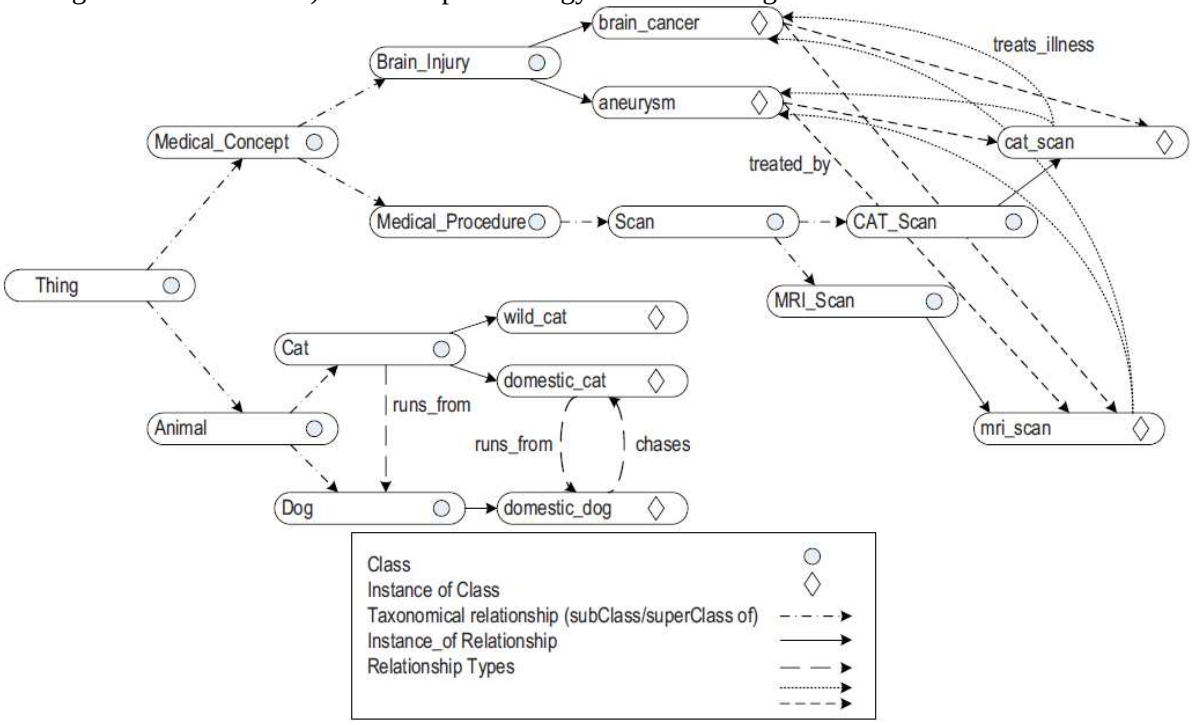


Figure 1. A simple (and incomplete) ontology describing ambiguous senses of the word "cat"

2.2 Ontology Evaluation

Evaluating ontology semantics commonly focuses on the structural and logical nature of the resource. Related efforts may use logical reasoning to ensure that the semantics are internally consistent, or use the structure and labels of another ontology as a baseline, assuming that textual labels for synonymous concepts will be consistent between sources (Vrandecic and Sure, 2007; Ma, 2013). A metric which goes beyond these and evaluates the semantic relevance to a given task is sorely needed (Vrandecic and Sure, 2007). While metrics that examine the completeness of an ontology's content are suggested in the literature (Tartir et al, 2005), these metrics reflect a high-level summary of the content. The evaluation of this content, independent of the ontology itself, and at a sufficiently fine-grained level to suggest areas for improvement, would be of significant additional benefit.

Vrandecic and Sure (2007) recognise the paucity of metrics that take the ontology semantics into account. In terms of semantic quality, they propose leveraging a logical reasoner to evaluate that an ontology is consistent within the context of its own assertions. However, there is no objective analysis of the semantic content with respect to real world human knowledge. Ma et al (2013) point out that prior

[2] FMA: http://sig.biostr.washington.edu/projects/fm/AboutFM.html
[3] RXNORM: http://bioportal.bioontology.org/ontologies/RXNORM
[4] SNOMED: http://bioportal.bioontology.org/ontologies/SNOMEDCT

ontology metrics neglect implicit semantic knowledge. They acknowledge the utility of a graph structure in representing the content of an ontology, and assert that this structure preserves well the semantics of the ontology. However, they do not proceed to examine the ontology in the context of a real-world semantic evaluation. By limiting the scope of comparison to sets of related ontologies, they work on the assumption that similarly labelled concepts and structures are roughly equivalent. Additionally, Sicilia et al (2012) suggest that there is no obvious metric to identify when an ontology needs to be improved.

We propose that graphs composed of instances of ontology concepts and relationships, along with associated unique identifiers, are a less naïve approach to semantic matching than textual labels. We suggest an objective analysis of how annotated instances of ontology concepts and relationships interact, by a process such as spreading activation in an associated graph, would be more reflective of the proximity of the evaluated ontology to the semantics of the target domain text. We also suggest that analysis of particular characteristics of the graph, that amplify or hinder this activation process, are helpful in identifying specific portions of the associated ontology that require further development. Interestingly, the use of spreading activation as a method for ontology assessment has already been carried out previously (Fang and Evermann, 2010). In that case however, the spreading activation was in the context of cognitive psychology, where test subjects manually assessed ontology content. An automated approach, leveraging the same principles, without the requirement for human reviewers, would be of great value.

2.3 Word Sense Disambiguation

WSD is one of the most critical tasks in NLP (Navigli, 2009), and is often described as AI complete. Navigli (2009) identifies several main categories of approach to WSD, namely knowledge based, supervised and unsupervised methods. He proposes knowledge based methods as the most useful in the medium to long term, for several reasons. He points to the availability of knowledge resources such as WordNet, Yago, and DBPedia, resources which are actively developed and enriched, as a starting point of significant value. He also suggests that supervised approaches are better for categorisation tasks like part-of-speech (POS) tagging, rather than tasks that require more fine grained detail such as real-world word-sense disambiguation. As an example of this, consider that the process of disambiguating the correct POS for a word may involve the selection of one from a set of possible POS tags. One such tagset, widely used for English, is the Penn Treebank tagset consisting of 36 separate tags. The UMLS data set, however, contains close to 3 million[5] distinct senses.

Though WSD is still widely regarded as an unsolved problem, supervised approaches to WSD generally perform well. Navigli (2009) suggests that this is due to the lack of real-world considerations in development and testing of WSD methods. We can consider the MSH-WSD corpus as an example demonstrating typical limitations when compared with the requirements for a real-world system. MSH-WSD is a commonly used data set in biomedical WSD, using sense IDs from UMLS, and consisting of approximately 37,000 separate documents or abstracts, where a single ambiguous sense is annotated with the correct UMLS sense ID. A WSD system need only identify this single sense correctly (regardless of the other words in the document), in order to score highly. Additionally, there are a total of 423 distinct word-senses annotated in this test set, greatly reducing the scope of the task involved from approximately 3 million possible senses in the full UMLS. As a result, this data set is not a strong reflection of what is required in real-world biomedical NLP applications, where a high percentage of the words in a given document or context must be assigned their correct senses.

It is generally accepted that unsupervised methods for WSD minimise the cost of developing a suitable application, by relying on features that may be extracted directly from the target domain text, or alternatively using existing knowledge in some form. The latter are often referred to as knowledge-based (KB) methods. For supervised WSD a gold-standard is required input, where manually curated data sets facilitate the training of robust machine learning algorithms. Supervised methods generally outperform unsupervised (Agirre et al, 2010), but are limited by the cost of developing the required training data. However, as mentioned previously, these systems may not perform so well in real world WSD scenarios.

In a biomedical context, there are several examples of both supervised and unsupervised (including knowledge-based) approaches. Most unsupervised approaches leverage the UMLS to some extent, and build on that knowledge using methods like Automated Corpus Extraction (Jimeno Yepes and Aronson, 2012) and Information Content Similarity (McInnes et al, 2011). The commonly cited example of a

[5] UMLS stats: http://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus/release/statistics.html

supervised approach that consistently outperforms known unsupervised approaches is Naïve Bayes (Jimeno Yepes and Aronson, 2012; McInnes et al, 2011), achieving 0.94 accuracy on the common data set, although as we've outlined previously, the search space for a correct tag in the chosen data set (with a total number of 423 senses) is much smaller than would be the case in a real-world system.

Several recent approaches to biomedical WSD leverage structured knowledge in the form of a graph. Examples range from the use of co-occurrence data from a domain-specific corpus (Agirre et al, 2006), to variations of PageRank (Agirre and Soroa, 2009; Agirre et al, 2010), to the representation of an ontology, or portion of an ontology, as a graph (Gad El-Rab et al, 2013). Ontologies are often used as a source from which to build the required graph, as they are readily available in many domains, and provide a starting point of high-quality semantic knowledge. As identified previously, the lexical ontology Wordnet is a commonly used resource in open domain WSD. Similarly, in the biomedical domain, the UMLS is equally common. Hybrid approaches leveraging both general lexical semantics like WordNet with domain-specific semantics like UMLS are not as common however, but have been used with promising results in other related NLP tasks such as anaphora resolution (Liang and Lin, 2005).

Graph based methods have not performed as well as other unsupervised approaches, like Machine Readable Dictionaries: 0.8070 (Jimeno Yepes and Aronson, 2012), semi-supervised Automated Corpus Extraction methods: 0.8383 (Jimeno Yepes and Aronson, 2012), and co-occurrence metrics: 0.78 (McInnes and Pedersen, 2013). A recent approach (El-Rab et al, 2013) achieved mixed results with respect to particular terms in the MSH-WSD test corpus, achieving an overall accuracy of 0.603. State of the art accuracy for graph-based methods, in unsupervised biomedical WSD, was 0.72 (McInnes et al, 2011). State of the art in overall unsupervised biomedical WSD was 0.87 (Jimeno-Yepes and Aronson, 2012).

2.4 Spreading Activation

The theory of spreading activation was first proposed by Quillian (1966), in a model of human semantic memory. Quillian proposed an abstract model of human memory, in order to artificially represent the means by which a human's brain might process and understand the semantics of natural language. This model was enhanced by Collins and Quillian (1969) for retrieval tasks, and further modified by Collins and Loftus (1975). The latter provided inspiration for research in many other related fields, from cognitive psychology to neuroscience, to natural language processing, among others (Pace-Sigge, 2013).

The basic premise of spreading activation is related to that of connectionism in artificial intelligence, which uses similar models for neural networks to reflect the fan-out effect of electrical signal in the human brain. In the case of neural networks, a vertex in the graph could represent a single neuron, and edges could represent synapses. In information retrieval (Crestani, 1997) and word-sense disambiguation (Tsatsaronis et al, 2007), generally vertices will represent word-senses and edges will represent some form of relationship, either lexical or semantic linkage, between these senses.

An example implementation is 'Galaxy', developed as part of the Nepomuk Social Semantic Desktop[6], which uses spreading activation to perform clustering on a graph. Instead of traditional methods of hard clustering, which partition a graph into different groups, Galaxy performs soft clustering, which involves identifying a sub-graph located around a set of input nodes, and then finding the focus of this sub-graph. The same implementation provides a configurable weighting model that allows modification of starting weights associated with semantic types, edges and individual nodes in the graph. This has already been used in various scenarios, such as social network analysis and dynamic semantic publication of web content[7], and may also be applied to any set of graph-structured data (Troussov et al, 2008).

By discovering instances of ontology concepts in domain text, using the set of unique identifiers for instances, we can activate corresponding nodes in the graph, from where a signal will traverse outward across adjacent nodes, activating these in turn. As the signal spreads farther from a source node, it gets weaker by an amount specified in an associated weighting model for nodes and edges in the graph. If the signal spreads from multiple nearby source nodes, the signal will combine, and points of overlap will be activated to a greater degree. The nodes which accumulate the most activation are deemed to be the focus nodes for the context. The resulting activated portion of the graph will reflect the inherent meaning of the document, in so far as the ontology's defined semantics will allow.

[6] http://dev.nepomuk.semanticdesktop.org/wiki/TextAnalytics#IBM
[7] http://www.bbc.co.uk/blogs/bbcinternet/2010/07/bbc_world_cup_2010_dynamic_sem.html

To demonstrate this process in action, we will draw examples from the ontology previously defined above. Figure 2 describes the resulting instance graph for the ontology described in Figure 1, on which we can perform spreading activation using instances in text. Firstly, consider the set of surface forms associated with concept instances in table 1. If we annotate the set of contexts below with this lexicon, we can then use the annotations to activate the graph. Nodes that are well connected may benefit from the potential overlap of signal coming from other adjacent nodes. Instances are *italicised* below.

- The *cats* result for the patient's *brain tumour* was assessed by the Doctor.
- *Tigers* and *lions* are *cats* that live in the wild. These *cats* are not afraid of *dogs*.
- The patient survived the *brain tumour*, but died of an allergic reaction to their neighbour's *cats*.

In each example, the ambiguous term is the word “cats”, which can variously refer to: cat_scan, wild_cat and domestic_cat. The surrounding context of each instance contains other concept instances that may help to disambiguate the correct sense of “cats”. In the first example, the nodes representing wild_cat, domestic_cat, cat_scan and brain_cancer will be activated. Since brain_cancer and cat_scan are relatively well connected in the graph, and are also adjacent to one another, the spreading activation will return these nodes as the most likely interpretation of the content.

In the second example, the correct instance is wild_cat. However, this node is isolated in the graph, since there were no associated relationships in the ontology linking this particular instance to other nodes. Since the instance of the class Dog is connected to domestic_cat, these nodes may amplify each other’s signal to a greater degree than is possible at the isolated node wild_cat. It is therefore likely that unless the weighting model is reconfigured, we are unlikely to obtain the correct output. The relevance of isolated nodes may be boosted by increasing the rate of signal decay on other nodes in the graph. However, there is a risk in doing so, since the connectedness of instances in the ontology is likely a better reflection of the semantic content. It would be better to suggest that the ontology would benefit from further development, for example to introduce the ideas of habitat or fear.

The final example demonstrates a more subtle bias in the ontology’s semantics, and the corresponding graph. The overlapping signal from cat_scan and brain_cancer suggests that cat_scan will be returned instead of domestic_cat. Resolving this ambiguity in the graph may require modification of the weighting model, or further development. An advantage in this case however, is the different semantic categories involved: the classes of Cat and Scan. Re-weighting the starting activation signal on the basis of a semantic category is less risky than re-weighting the entire set of nodes in the graph. Even so, further development of the ontology, e.g. to introduce the idea of animal allergies, would be beneficial.

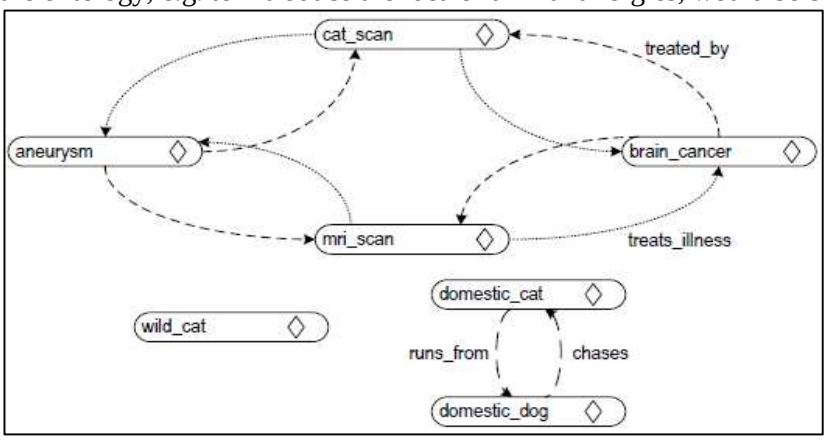


Figure 2. Graph representation of the sample ontology.

Instance ID	Associated Surface Forms
wild_cat	{lions, tigers, cat, cats, cub}
domestic_cat	{cat, cats, kitten }
domestic_dog	{dog, dogs, puppy}
brain_cancer	{brain carcinoma, brain tumour}
cat_scan	{cat, cats, cat scan}

Table 1. Example surface forms for instance data.

3 Method

3.1 Ontology Instance Graph

We extracted data from the UMLS Metathesaurus (MT) and Semantic Network (SN) and built a triple store in RDF/XML[8] format, defining owl:Class and owl:ObjectProperty to reflect concepts and relationships. Using the Galaxy API described in section 2.4, we built a spreading activation network, i.e. a directed graph between instance IDs (vertices) and associated relationships (edges). In order to narrow the proximity between the semantics of domain text and the chosen ontology, we chose to build a graph of instance data. The SN is a high level ontology, and therefore to assume that all relationships between Classes are applicable to all instances would have produced many incorrect assertions, such as "All Drugs have the set of All Drugs as ingredients". Therefore, only instances of relationships that explicitly linked individual concept IDs (Concept Unique Identifiers, CUIs) were used. Across the entire SN, a single CUI may have various types of semantic interactions with other nodes, for example in the context of Drugs and treated Diseases, or separately, in the context of Chemicals and associated Compounds. The UMLS CUIs were used as instance IDs to link surface forms in the text to nodes in the graph.

It is important to point out that the UMLS ontology by no means uses the full expressivity of OWL. However, the general use of spreading activation over a graph derived from ontology content, is not so limited. In other domains, and for ontologies that use the full range of OWL expression, as long as the graph is built from a source that expresses other semantic qualities (e.g. cardinality), the spreading activation strategy will still apply. For example, in the context of our sample ontology, consider activating "cat", the signal spreading to an additional adjacent node for the concept of "four legs", and then other concepts with four legs, such as "dog", becoming activated. The Galaxy API fully supports this.

3.2 Test Corpus and Metric Calculation

We chose to use the MSH-WSD test corpus as our gold-standard. This is a common test set used across the literature in biomedical WSD. The metrics we used were Precision, Recall, FMeasure and Accuracy, whereas prior research mainly focuses on Accuracy. In WSD, a true positive is a disambiguated output that matches a gold-standard, and a false positive is output that does not match. As traditional WSD algorithms are designed to generate output for every word in the text, recall and precision are the same value. However, our algorithm works on the principle of semantic relevance, and there is no guaranteed output; senses with sufficient weight after spreading activation will be displayed. Therefore, we have chosen to take a closer look at precision and recall, which is discussed in more detail in section 4.

Prior literature in biomedical WSD uses older versions of UMLS data, e.g. 2009AB (McInnes et al, 2011). We chose to focus on the 2013AA release of UMLS, in order to assess the most recent version of the ontology's semantic content, and in order to facilitate a useful modification of the current data, which could be leveraged by contemporary NLP systems. This affected the comparison of test results using the MSH-WSD data set.

3.3 Lexical Annotation

In conjunction with the graph described above, we constructed a set of lexical dictionaries that linked UMLS CUIs or instance IDs, to portions of text in a document. These portions of text, otherwise known as surface forms, consisted of potentially many different strings associated with each ID. An example of a data entry for a single UMLS CUI is in table 2 below. Dictionaries were compiled for each semantic category in the UMLS SN, with overlapping associations between ID and textual surface form.

CUI	Semantic Type	Surface Form (Text)
C0018787	BodyPartOrRegion	heart
		cardiac structure
		heart structure
		coronary
		four chambered heart
		the human heart

Table 2. Surface forms associated with the concept "Heart", UMLS CUI: C0018787.

[8] http://www.w3.org/TR/rdf-syntax-grammar/

In order to maximise the potential for spreading activation across the graph, we performed several modifications to the underlying lexical data in UMLS MT, to increase the variations of surface form associated with instances of concepts. Our reasoning for this is as follows: the more instances of concepts that occur in the text, the more nodes that get activated in the graph, and consequently the more opportunities for the activation method to spread out and activate the set of concepts most relevant to the semantics of the document text. For a simple example of this process, please see section 2.4. Examples of transformations carried out in the data are presented in table 3, below.

Pre-existing Term	Transformation Type	New Alternate Surface Form
leg, right	Alternating Comma	right leg
brain cancer	Noun Phrase	cancer of the brain
CANCER	Casing Variants	Cancer
Anaemia	Spelling Variants	Anæmia
Immunoglobulin g	Acronym	Ig
Immunoglobulin g	Term + Acronym	Immunoglobulin g (Ig)

Table 3. Examples of UMLS data transformations applied.

The use of a lexical part-of-speech tagger was particularly effective in filtering out instances of concepts that were obviously introducing unhelpful noise. Some exemplary cases were the Amino Acids "on", "at" and "in" (prepositions), and the GeneOrGenome "was" (verb). UMLS concepts that directly overlapped with words that did not display an appropriate part-of-speech for a true concept (such as adjective or noun), were removed from the document metadata, and thereby not considered as input for spreading activation. For this POS Filter, we chose to use the MaxEntropy model from OpenNLP[9].

3.4 Spreading Activation Strategy

The initial activation strategy was to set starting weights for all semantic categories to a value of 1. Decay factor of the spreading signal at each node in the graph was set to an initial value of 0.5, when the graph was built. The initial threshold of semantic relevance was set to 0.1, and instances retaining a semantic value higher than this would be considered relevant. The lexical annotations from the previous step were used as input to the activation process, and nodes in the graph from instances in the text were assigned their starting weight, according to the number of semantic categories, and their associated weights. As the signal is spread from these starting nodes, the decay factor is applied, reducing the signal strength. For each successive node, the signal is similarly reduced until it falls below the specified threshold, and the activation process is completed. It is important to note that the ambiguity in word-senses may not be entirely removed once the spreading activation has finished. The consequences of this will depend on the particular end-goal. In the case of WSD, we are only interested in obtaining a single most appropriate CUI for a given surface form. We therefore kept only the highest weighted CUI in our system output. In the context of other NLP tasks, such as for named-entity inference or question answering and hypothesis generation (Ferucci et al, 2011), it can be useful to preserve multiple ambiguous outputs for later processing.

It was clear from the outset that simply building a graph of the ontology instance data and semantic relationships was not sufficient to score highly in the WSD task. El-Rab et al (2013), who used the UMLS SN structure for graph-based WSD, reported an overall accuracy of (0.603) on the MSH-WSD test set, which roughly correlates with our baseline system (0.62). Our added advantage is that modification of the weighting strategy allows us to iron out imbalance, or to reduce the influence of those portions of the graph that do not appear to encourage a spreading signal. By focusing on signal amplification and decay, rather than modifying graph semantics, we can change the relevance of particular portions of the ontology without losing any of the original semantic detail. Such modifications are sensitive to performance in the NLP task but, critically, do not require the assistance of domain experts.

We initially pursued a cautious approach to modifying the activation strategy, by only decreasing the starting weight of semantic categories associated with the affected nodes. This weight was decreased by a factor equivalent to the number of overlapping semantic types on the same node. Following this, we measured the accuracy of the approach against the MSH-WSD test corpus for WSD, testing blind, that is by only considering the overall accuracy. Upon close examination of the instance graph, for types of

[9] http://opennlp.apache.org/

structure or characteristics of nodes that may be hindering or over-amplifying the spreading signal (see section 4.1), we further modified the activation strategy to negate the potential influence that certain obviously problematic nodes may have. Modifying our spreading activation strategy in this way, after static graph analysis alone, produced much more accurate output (see table 5, experiment 3).

We then decided to split the test set in the ratio of 4:1, in order to more closely inspect the accuracy of particular cases of WSD, and attempt to correct this specific imbalance in the graph, while still performing some independent validation of the output. The random nature of the split was to choose every fifth example in the data, from the subset for each term. After performing WSD using this 80%, or train set, we discovered that it was possible to distinguish groups of high and low performing nodes in the graph, with respect to the set of static graph metrics, described in the following section.

3.5 Static Graph Analysis (SGA)

As shown in the simple example in 2.4, assessment of ontology semantics can be done up front, before the graph is used. Certain node characteristics may be examined in the graph using a set of graph theoretical metrics, and portions of the graph that are not conducive to spreading activation may be identified. This analysis allows us to make educated modifications to the weighting strategy for spreading activation, as described previously. The set of graph metrics we used is presented in table 4 below.

Metric	**Evaluation**
In Degree	# of inward semantic links
Out Degree	# of outward semantic links
Total Degree	(indegree + outdegree)
Inward Edge Type Variation (ETV)	# of inward edge types
Outward ETV	# of outward edge types
Total ETV	(Inward ETV + Outward ETV)

Table 4. Static Graph Metrics derived from Diestel (2010).

Following the use of these metrics, and the gathering of associated statistics, we categorised particular groups of node in order to apply a common weighting strategy that should maximise performance of the spreading activation algorithm. There were several common patterns that we identified, and chose to target for re-weight. Examples of those nodes that might negatively affect spreading activation are:

- Isolated Nodes, where Total Degree is 0
- Unbalanced Nodes, where inDegree and outDegree are significantly different
- Nodes with few variations in link type, or low Total ETV
- 'Black Hole' nodes, where there is a high Degree to ETV ratio (see section 4.1)

For isolated nodes, we examined the set of associated semantic categories, and boosted their starting weight. For unbalanced nodes, where the indegree was significantly higher or lower than the outdegree, we increased or decreased the decay factor accordingly, to reduce the imbalance of the spreading signal. For nodes with low ETV but high Degree, we increased the decay factor, in order to reduce the potential influence of a single over-used semantic link. For overly promiscuous (Norvig, 1986) or 'Black Hole' nodes, we reduced the starting weight applied by the associated semantic categories, and increased the rate of decay. In certain cases, the intended modifications were incompatible, and resulted in conflicting changes to the graph and weighting strategy. Where certain nodes might require a boost from one category, the starting weight for the same category may need to be reduced, due to an overly-connected node elsewhere. We decided to inhibit the negatively connected nodes only, in light of the increase in system accuracy from reducing noise compared to the gain from improvement of individual nodes.

4 Results and Discussion

The baseline activation strategy was promising. The introduction of a POS filter to ignore invalid instances (see section 3.3) had a strong effect on recall, due to reduced noise in the activation of the graph. Recall significantly improved upon the modification of starting weights after analysis of static graph metrics, although precision fell slightly. This result (0.82) constitutes state of the art in graph-based WSD for biomedical text. The fall in precision was not unexpected, since the graph was no longer so

biased toward specific word senses. We also present a further experiment that incorporates a fall-back mechanism for test cases where the spreading activation did not produce a disambiguated output. This result (0.89) constitutes state of the art in overall unsupervised biomedical WSD. This allows our method to assign a single word-sense for every ambiguous word or surface-form. This fall-back alone achieves accuracy of 59%, comparing favourably with a default-sense approach (54.5%: McInnes et al, 2011).

Finally, by identifying bias in the graph toward specific senses in the test corpus, using an 80% subset of the MSH-WSD data set for training, and then modifying the rate of decay for problematic nodes, we achieved a significant boost to recall, and consequently to overall accuracy. We draw a distinction between this and other results since the testing was not blind, but was using the gold-standard corpus directly, to examine the portions of the graph that did not perform well in testing. We envisage that this may still be of practical use in real-world applications, by firstly developing an appropriate gold-standard, which in conjunction with analysis of the ontology instance graph, will result in optimal output.

The current results reflect the scope of spreading activation being set to the whole document. Only one sense of a word is recognised within that context, and documents containing multiple interpretations of the same word will not be correctly disambiguated. However, by configuring the scope to a sentence or paragraph we may reduce the potential accuracy of the output by decreasing the available instances for activation. Prior research into the "One sense per discourse" hypothesis suggests that the existing approach should be appropriate in up to 98% of cases (Gale et al, 1992).

Experiment Description	Precision	Recall	FMeasure	Accuracy
1. Baseline system	0.935	0.659	0.6639	0.62
2. Baseline + POS Filter	0.901	0.721	0.7872	0.74
3. As in 2, with SGA re-weight	0.841	0.822	0.8317	0.82
4. As in 3, confidence fallback	0.912	0.887	0.8995	0.89
5. SGA+WSD (20% test set)	0.986	0.942	0.9635	0.93
McInnes et al, 2011				0.72
J-Yepes & Aronson, 2012				0.87

Table 5. Comparison of WSD Results.

4.1 Identifying and Resolving Graph Bias or Imbalance

In experiment 5, having already identified specific cases that remained unbalanced, we attempted to rectify this by examining the graph in parallel with the WSD metric data. If a graph displays characteristics indicating imbalance or bias, for example where a node is unreachable (isolated in the graph), or node degree and node edge-type variation are relatively low (see section 3.5), it is less likely that the spreading activation will reflect the meaning of the text. We made discoveries similar to the following:

- 80% of nodes with Total ETV >15 had WSD precision of over 90%
- 60% of nodes with Total ETV <5 had precision of less than 10%

We also discovered cases in the graph where a node had very high Degree (> 100), and relatively low ETV. In terms of spreading activation, these nodes would be especially problematic. We have coined the term 'Black Hole Node' to describe this phenomenon. In psycholinguistic terms, this may be comparable to the notion of a Freudian slip, where a node in the graph which is not immediately relevant to the context of the document, has become over-stimulated by its connectivity, or as Norvig (1986) would suggest, its "promiscuity". The signal will gravitate towards such an over-connected node during the process of spreading activation, affecting the relevance of other nodes in that context. An example black hole node is the UMLS CUI C0035298, representing a retina in a human eye, with 1636 edges and 19 edge types. The extra noise in activating such a node can skew the signal across the entire graph. Word senses that compete for relevance with this or related nodes will have poorer accuracy. We modified the activation strategy to reflect this by increasing the rate of decay on such nodes from 0.5 to 0.99.

By ensuring that only the graph weighting strategy is modified, we can keep all word-senses present in the graph, resolving the issue identified by Norvig (1986) where such graph content had to be removed. Using the WSD metric output, we also modified the activation strategy to cope with bias toward particular senses in the test corpus. We reduced the starting weight for semantic categories for the high-scoring sense, in order to potentially increase the relative semantic importance of the alternative senses. Table 6 demonstrates some of the improvements achieved with regard to specific ambiguous terms.

Term	F-Measure Before	F-Measure After
Murine Sarcoma Virus	0	0.47
Gamma-Interferon	0.013	0.28
RA	0.021	0.59
CCD	0.033	1
AA	0.899	0.99

Table 6. Examples of term-specific improvement using re-weighting strategy.

4.2 MSH-WSD Data Set

In working with the MSH-WSD data set, we came across many issues that Navigli (2009) previously identified. The number of ambiguous senses (423) in the context of the full UMLS set of almost 3 million, reduces the validity of this corpus for measuring real-world viability and accuracy. Further to this, our results with lexical analysis optimisation demonstrate that the test corpus ignored surrounding context for potentially overlapping terms, such as “bat” and “fruit bat”. In such cases, it would have been more accurate to use the CUI for “fruit bat” as the specific type of “bat”, but the test corpus does not reflect this. Our algorithm is sensitive to contextual semantics, so ensuring that all lexical matches of any length remain present, potentially reduces the accuracy of the algorithm’s output, as well as the real-world utility of the approach. In spite of the various data transformation techniques applied, our recall maximised at 96.4%. Critically, when we normalize our overall accuracy (0.89) to take this into account, we reach accuracy of 0.92, a significant achievement in unsupervised WSD. We are currently examining what may be required to achieve maximum recall of 100%. While such a result is not guaranteed, without full coverage of the test set, we have not yet measured the full potential of this method.

4.3 Identifying Focus Areas for Ontology Improvement

One of the primary outcomes of this research is a method for the identification of specific ontology portions that require further development. As we have seen in section 4.1, there are several candidates which stand out. Other issues pointing to required enhancements in the ontology were around the notion of isolated nodes in the graph. An example of this is "ADA", the American Dental Association. It is surprising to discover that although this term’s associated CUI (C0002456) is listed in 7 source ontologies of the UMLS SN, there are no semantic relationships in the source between this CUI and any others. Of the 203 ambiguous terms in the MSH-WSD data set, 5 of those terms had associated nodes that were similarly isolated in the graph. Without any semantic relationship to other concepts, it is reasonable to suggest that the ontology would benefit from focused development of these nodes’ surrounding context.

In terms of the variation of connectivity, we quickly discovered using our simple graph metrics that the “SIB” or sibling relationship was extremely common. Consider the concept C0325089 representing the *felidae family* or the animal *cat*, which has 8 connections, but for which SIB is the only available link type. Hard-wiring siblings in this fashion, with no other link, is unhelpful since spreading activation can already identify siblings from common parent nodes. We contend that such concepts are not as well connected as they may first appear, and are therefore strong candidates for further development. This will not be apparent from the Degree metric alone, but by combining Degree and Edge Type Variation with node-specific accuracy in an NLP task, it becomes a straightforward process. Following this discovery, we also suggest that an empirical analysis of link quality would be beneficial, although this would not be a trivial task given the size of the data set (~3 million senses and ~700 link types).

5 Summary and Future Research

We have presented a new method for evaluating ontology semantics which has several advantages over existing approaches. We have shown how the application of graph theoretical analysis to semantic structures like ontologies is a valid means by which to assess their semantic quality, while enabling the recommendation of specific focus areas for further development. We have additionally demonstrated that a graph-metric based weighting strategy for spreading activation can overcome an ontology’s inherent semantic inconsistencies, facilitating the optimisation of the ontology for a given NLP task.

In the case of our UMLS prototype, we made significant improvements using this technique, achieving state of the art in unsupervised knowledge based WSD (0.82), as well as achieving state of the art in overall unsupervised WSD, with the use of a fall-back probability score (0.89). An additional semi-

supervised approach, leveraging gold-standard data from a training portion of the MSH-WSD data set, had very promising performance (0.93). The amount of required input data to this method is relatively small when compared with fully supervised approaches, as a single gold-standard annotation in each target context is sufficient to evaluate the graph using our spreading activation algorithm.

In future we would like to apply this technique to other ontologies, and associated test sets, for other domains in NLP. Merging of domain-specific ontologies with more general semantic resources like Yago or Wordnet may help to facilitate the activation of otherwise poorly connected or isolated nodes in the graph. We would like to investigate the automatic learning of an optimal spreading activation weighting strategy. An empirical study comparing data from human ontology reviewers with this spreading activation technique, would also be helpful.

We would like to expand the set of metrics used, by adapting other existing graph theoretical metrics to suit the requirements of NLP. Some promising examples are "Centrality" and "Betweenness" outlined by Brandes and Erlebach (2005), which determine the relative importance of a node within a graph. In the case of UMLS, we can perform a comprehensive static analysis of all ambiguous CUIs within the data set, identifying competing senses which do not have sufficient separation in the graph. These senses could then be targeted in the configuration of the spreading activation strategy.

As interest grows in the use of graph theoretical methods for the analysis of cognitive processes (Van Dijk et al, 2010; Bullmore and Sporns, 2009; Sporns, 2003), exploring the relationship between spreading activation in a graph representing ontology semantics, as performed in this research, and in neural activity during psycholinguistic experimentation (Fang and Evermann, 2010), becomes an exciting prospect that may lead to a better understanding of semantic processing in the human brain.

Acknowledgements

Sincere thanks to Mikhail Sogrin (IBM), the talented developer of both 'Galaxy' and the lexicon expansion framework used here, without whom this research would not have been possible.

References

Agirre, E., Martínez, D., de Lacalle, O. L., & Soroa, A. (2006, July). Two graph-based algorithms for state-of-the-art WSD. In Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (pp. 585-593). Association for Computational Linguistics.

Agirre, E., & Soroa, A. (2009, March). Personalizing PageRank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics.* ACL

Agirre, E., Soroa, A., & Stevenson, M. (2010). Graph-based Word Sense Disambiguation of biomedical documents. *Bioinformatics*, *26*(22), 2889-2896.

Brandes, U., & Erlebach, T. (Eds.). (2005). Network analysis: methodological foundations (Vol. 3418). Springer.

Bullmore, E., & Sporns, O. (2009). Complex brain networks: graph theoretical analysis of structural and functional systems. Nature Reviews Neuroscience, 10 (3), 186-198.

Collins, A. M., & Quillian, M. R. (1969). Retrieval time from semantic memory. Journal of verbal learning and verbal behavior, 8(2), 240-247.

Collins, A. M., & Loftus, E. F. (1975). A spreading-activation theory of semantic processing. *Psychological review*, *82*(6), 407.

Crestani, F. (1997). Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, *11*(6), 453-482.

Diestel, R. (2005), *Graph Theory* (3rd ed.), Berlin, New York: Springer-Verlag, ISBN 978-3-540-26183-4.

El-Rab, W. G., Zaïane, O. R., & El-Hajj, M. (2013, August). Biomedical text disambiguation using UMLS. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (pp. 943-947). ACM.

Evermann, J., & Fang, J. (2010). Evaluating ontologies: Towards a cognitive measure of quality. *Information Systems*, *35*(4), 391-403.

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., & Welty, C. (2010). Building Watson: An overview of the DeepQA project. AI magazine, 31(3), 59-79.

Gale, W. A., Church, K. W., & Yarowsky, D. (1992, February). One sense per discourse. In Proceedings of the workshop on Speech and Natural Language (pp. 233-237). Association for Computational Linguistics.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, *5*(2)

Guarino, N., Oberle, D., & Staab, S. (2009). What is an Ontology? In *Handbook on ontologies* (pp. 1-17). Springer

Jiménez-Ruiz, E., Grau, B. C., & Horrocks, I. (2012). Exploiting the UMLS Metathesaurus in the Ontology Alignment Evaluation Initiative. In *E-LKR Workshop* (pp. 1-6).

Jiménez Ruiz, E., Grau, B. C., Horrocks, I., & Berlanga, R. (2011). Supporting concurrent ontology development: Framework, algorithms and tool. *Data & Knowledge Engineering*, *70*(1), 146-164.

Jimeno Yepes, A., & Aronson, A. R. (2012, January). Knowledge-based and knowledge-lean methods combined in unsupervised word sense disambiguation. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium* (pp. 733-736). ACM.

Liang, T., & Lin, Y. H. (2005). Anaphora resolution for biomedical literature by exploiting multiple resources. In Natural Language Processing–IJCNLP 2005(pp. 742-753). Springer Berlin Heidelberg

Ma, Y., Jin, B., Liu, X., Liu, L., & Lu, K. (2013). A Graph Derivation Based Approach for Measuring and Comparing Structural Semantics of Ontologies. *IEEE Transactions on Knowledge and Data Engineering*, 1.

McInnes, B. T., Pedersen, T., Liu, Y., Melton, G. B., & Pakhomov, S. V. (2011). Knowledge-based method for determining the meaning of ambiguous biomedical terms using information content measures of similarity. In *AMIA Annual Symposium Proceedings* (Vol. 2011, p. 895). American Medical Informatics Association.

McInnes, B. T., & Pedersen, T. (2013). Evaluating measures of semantic similarity and relatedness to disambiguate terms in biomedical text. Journal of biomedical informatics, 46(6), 1116-1124.

National Library of Medicine. 2013. *Unified Medical Language System*, version 2013AA. NLM

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, *41*(2), 10.

Norvig, P. (1986). Unified theory of inference for text understanding. CALIFORNIA UNIV BERKELEY GRADUATE DIV.

Noy, N. F. (2004). Tools for mapping and merging ontologies. In *Handbook on ontologies* (pp. 365-384). Springer

Pace-Sigge, M. (2013). Lexical Priming in Spoken English Usage. Palgrave Macmillan.

Pisanelli, D. M., Gangemi, A., & Steve, G. (1998). An ontological analysis of the UMLS Metathesaurus. In *Proceedings of the AMIA symposium* (p. 810). American Medical Informatics Association.

Plaza, L., Jimeno-Yepes, A. J., Díaz, A., & Aronson, A. R. (2011). Studying the correlation between different word sense disambiguation methods and summarization effectiveness in biomedical texts. *BMC bioinformatics*

Quillian, M.R. (1966). Semantic Memory. Unpublished doctoral dissertation, Carnegie Institute of Technology (Re-printed in part in M. Minsky (1968). Semantic Information Processing. Cambridge, Mass. MIT Press).

Sicilia, M. A., Rodríguez, D., García-Barriocanal, E., & Sánchez-Alonso, S. (2012). Empirical findings on ontology metrics. *Expert Systems with Applications*, *39*(8), 6706-6711.

Sporns, O. (2003). Graph theory methods for the analysis of neural connectivity patterns. In *Neuroscience Databases* (pp. 171-185). Springer US.

Tartir, S., Arpinar, I. B., Moore, M., Sheth, A. P., & Aleman-Meza, B. (2005, November). OntoQA: Metric-based ontology quality analysis. In *IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources* (Vol. 9).

Troussov, A., Sogrin, M., Judge, J., & Botvich, D. (2008). Mining socio-semantic networks using spreading activation technique. In *Proc. International Workshop on Knowledge Acquisition from the Social Web*

Tsatsaronis, G., Vazirgiannis, M., & Androutsopoulos, I. (2007, January). Word Sense Disambiguation with Spreading Activation Networks Generated from Thesauri. In *IJCAI* (Vol. 7, pp. 1725-1730).

Van Dijk, K. R., Hedden, T., Venkataraman, A., Evans, K. C., Lazar, S. W., & Buckner, R. L. (2010). Intrinsic functional connectivity as a tool for human connectomics: theory, properties, and optimization. Journal of neurophysiology, 103(1), 297.

Vrandečić, D., & Sure, Y. (2007). How to design better ontology metrics. In *The Semantic Web: Research and Applications* (pp. 311-325). Springer Berlin Heidelberg.

Learning to Distinguish Hypernyms and Co-Hyponyms

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir and Bill Keller
Department of Informatics,
University of Sussex,
Brighton, UK
juliewe,D.Clarke,J.P.Reffin,davidw,billk@sussex.ac.uk

Abstract

This work is concerned with distinguishing different semantic relations which exist between distributionally similar words. We compare a novel approach based on training a linear Support Vector Machine on pairs of feature vectors with state-of-the-art methods based on distributional similarity. We show that the new supervised approach does better even when there is minimal information about the target words in the training data, giving a 15% reduction in error rate over unsupervised approaches.

1 Introduction

Over recent years there has been much interest in the field of distributional semantics, drawing on the distributional hypothesis: words that occur in similar contexts tend to have similar meanings (Harris, 1954). There is a large body of work on the use of different similarity measures (Lee, 1999; Weeds and Weir, 2003; Curran, 2004) and many researchers have built thesauri (i.e. lists of "nearest neighbours") automatically and applied them in a variety of applications, generally with a good deal of success.

In early research there was much interest in how these automatically generated thesauri compare with human-constructed gold standards such as WordNet and Roget (Lin, 1998; Kilgarriff and Yallop, 2000). More recently, the focus has tended to shift to building thesauri to alleviate the sparse-data problem. Distributional thesauri have been used in a wide variety of areas including sentiment classification (Bollegala et al., 2011), WSD (Miller et al., 2012; Khapra et al., 2010), textual entailment (Berant et al., 2010), predicting semantic compositionality (Bergsma et al., 2010), acquisition of semantic lexicons (McIntosh, 2010), conversation entailment (Zhang and Chai, 2010), lexical substitution (Szarvas et al., 2013), taxonomy induction (Fountain and Lapata, 2012), and parser lexicalisation (Rei and Briscoe, 2013).

A primary focus of distributional semantics has been on identifying words which are similar to each other. However, semantic similarity encompasses a variety of different lexico-semantic and topical relations. Even if we just consider nouns, an automatically generated thesaurus will tend to return a mix of synonyms, antonyms, hyponyms, hypernyms, co-hyponyms, meronyms and other topically related words. A central problem here is that whilst most measures of distributional similarity are symmetric, some of the important semantic relations are not. The hyponymy relation (and converse hypernymy) which forms the ISA backbone of taxonomies and ontologies such as WordNet (Fellbaum, 1989), and determines lexical entailment (Geffet and Dagan, 2005), is asymmetric. On the other hand, the co-hyponymy relation which relates two words unrelated by hyponymy but sharing a (close) hypernym, is symmetric, as are synonymy and antonymy. Table 1 shows the distributionally nearest neighbours of the words `cat`, `animal` and `dog`. In the list for `cat` we can see 2 hypernyms and 13 co-hyponyms[1].

[1]We read `cat` in the sense `domestic cat` rather than `big cat`, hence `tiger` is a co-hyponym rather than hyponym of `cat`.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2249–2259, Dublin, Ireland, August 23-29 2014.

cat	dog 0.32, animal 0.29, rabbit 0.27, bird 0.26, bear 0.26, monkey 0.26, mouse 0.25, pig 0.25, snake 0.24, horse 0.24, rat 0.24, elephant 0.23, tiger 0.23, deer 0.23, creature 0.23
animal	bird 0.36, fish 0.34, creature 0.33, dog 0.31, horse 0.30, insect 0.30, species 0.29, cat 0.29, human 0.28, mammal, 0.28, cattle 0.27, snake 0.27, pig 0.26, rabbit 0.26, elephant 0.25
dog	cat 0.32, animal 0.31, horse 0.29, bird 0.26, rabbit 0.26, pig 0.25, bear 0.26, man 0.25, fish 0.24, boy 0.24, creature 0.24, monkey 0.24, snake 0.24, mouse 0.24, rat 0.23

Table 1: Top 15 neighbours of `cat`, `animal` and `dog` generated using Lin's similarity measure (Lin, 1998) considering all words and dependency features occurring 100 or more times in Wikipedia.

Distributional similarity is being deployed (e.g., Dinu and Thater (2012)) in situations where it can be useful to be able to distinguish between these different relationships. Consider the following two sentences.

The cat ran across the road. (1)

The animal ran across the road. (2)

Sentence 1 textually entails sentence 2, but sentence 2 does not textually entail sentence 1. The ability to determine whether entailment holds between the sentences, and in which direction, depends on the ability to identify hyponymy. Given a similarity score of 0.29 between `cat` and `animal`, how do we know which is the hyponym and which is the hypernym?

In applying distributional semantics to the problem of textual entailment, there is a need to generalise lexical entailment to phrases and sentences. Thus, the ability to distinguish different semantic relations is crucial if approaches to the composition of distributional representations of meaning that are currently receiving considerable interest (Widdows, 2008; Mitchell and Lapata, 2008; Baroni and Zamparelli, 2010; Grefenstette et al., 2011; Socher et al., 2012; Weeds et al., 2014) are to be applied to the textual entailment problem.

We formulate the challenge as follows: Consider a set of pairs of similar words $\langle A, B \rangle$ where one of three relationships hold between A and B: A lexically entails B, B lexically entails A or A and B are related by co-hyponymy. Given such a set, how can we determine which relationship holds? In Section 2, we discuss existing attempts to address this problem through the use of various *directional* measures of distributional similarity.

This paper considers the effectiveness of various supervised approaches, and makes the following contributions. First, we show that a SVM can distinguish the entailment and co-hyponymy relations, achieving a significant reduction in error rate in comparison to existing state-of-the-art methods based on the notion of distributional generality. Second, by comparing two different data sets, one built from BLESS (Baroni and Lenci, 2011) and the other from WordNet (Fellbaum, 1989), we derive important insights into the requirements of a valid evaluation of supervised approaches, and provide a data set for further research in this area. Third, we show that when learning how to determine an ontological relationship between a pair of similar words by means of the word's distributional vectors, quite different vector operations are useful when identifying different ontological relationships. In particular, using the difference between the vectors for pairs of words is appropriate for the entailment task, whereas adding the vectors works well for the co-hyponym task.

2 Related Work

Lee (1999) noted that the substitutability of one word for another was asymmetric and proposed the alpha-skew divergence measure, an asymmetric version of the Kullback-Leibler divergence measure. She found that this measure improved results in language modelling, when a word's distribution is smoothed using the distributions of its nearest neighbours.

Weeds et al. (2004) proposed a notion of distributional generality, observing that more general words tend to occur in a larger variety of contexts than more specific words. For example, we would expect to be able to replace any occurrence of `cat` with `animal` and so all of the contexts of `cat` must be plausible

contexts for `animal`. However, not all of the contexts of `animal` would be plausible for `cat`, e.g., "the monstrous animal barked at the intruder". Weeds et al. (2004) attempt to capture this asymmetry by framing word similarity in terms of co-occurrence retrieval (Weeds and Weir, 2003), where precision and recall are defined as:

$$P_{ww}(u,v) = \frac{\Sigma_{f \in F(u) \cap F(v)} I(u,f)}{\Sigma_{f \in F(u)} I(u,f)} \text{ and } R_{ww}(u,v) = \frac{\Sigma_{f \in F(u) \cap F(v)} I(v,f)}{\Sigma_{f \in F(v)} I(v,f)}$$

where $I(n,f)$ is the pointwise mutual information (PMI) between noun n and feature f and F(n) is the set of all features f for which $I(n,f) > 0$.

By comparing the precision and recall of one word's retrieval of another word's contexts, they were able to successfully identify the direction of an entailment relation in 71% of pairs drawn from WordNet. However, this was not significantly better than a baseline which proposed that the most frequent word was the most general.

Clarke (2009) formalised the idea of distributional generality using a partially ordered vector space. He also argued for using a variation of co-occurrence retrieval where precision and recall are defined as:

$$P_{cl}(u,v) = \frac{\Sigma_{f \in F(u) \cap F(v)} min(I(u,f), I(v,f))}{\Sigma_{f \in F(u)} I(u,f)} \text{ and } R_{cl}(u,v) = \frac{\Sigma_{f \in F(u) \cap F(v)} min(I(u,f), I(v,f))}{\Sigma_{f \in F(v)} I(v,f)}$$

Lenci and Benotto (2012) took the notion further and hypothesised that more general terms should have high recall and low precision, which would thus make it possible to distinguish them from other related terms such as synonyms and co-hyponyms. They proposed a variant of the Clarke (2009) measure to identify hypernyms:

$$invCL(u,v) = \sqrt[2]{P_{cl}(u,v) * (1 - R_{cl}(u,v))}$$

Evaluation on the BLESS data set (Baroni and Lenci, 2011), showed that this measure is better at distinguishing hypernyms from other relations than the measures of Weeds et al. (2004) and Clarke (2009).

Geffet and Dagan (2005) proposed an approach based on *feature inclusion*, which extends the rationale of Weeds et al. (2004) to lexical entailment. Using data from the web they demonstrated a strong correlation between complete inclusion of prominent features and lexical entailment. However, they were unable to assess this using an off-line corpus due to data sparseness.

Szpektor and Dagan (2008) found that the P_{ww} measure tends to promote relationships between infrequent words with narrow vectors (i.e. those with relatively few distinct context features). They proposed using the geometric average of P_{ww} and the symmetric similarity measure of Lin (1998) in order to penalise low frequency words.

Kotlerman et al. (2010) apply the IR evaluation method of *Average Precision* to the problem of identifying lexical inference and use the balancing approach of Szpektor and Dagan (2008) to demote similarities for narrow feature vectors; their measure is called *balAPinc*. They show that all of the asymmetric similarity measures previously proposed perform much better than symmetric similarity measures on a directionality detection experiment, and that their method and that of Clarke (2009) outperform the others with statistical significance. They also show that their measure is superior when used for term expansion in an event detection task.

Baroni et al. (2012) investigate the relation between phrasal and lexical entailment, and demonstrate that support vector machines can generalise entailment relations between quantifier phrases to entailment involving unseen quantifiers. They compare the performance of their system with the *balAPinc* measure.

The Stanford WordNet project (Snow et al., 2004) expands the WordNet taxonomy by analysing large corpora to find patterns that are indicative of hyponymy. For example, the pattern "NP_X and other NP_Y" is an indication that NP_X is a NP_Y, i.e. that NP_X is a hyponym of NP_Y. They use machine learning to identify other such patterns from known hyponym-hypernym pairs, and then use these patterns to find new relations in the corpus. The transitivity relation of the taxonomy is enforced by searching only over valid taxonomies and evaluating the likelihood of each taxonomy given the available evidence (Snow

et al., 2006). The approach is similar to ours in providing a supervised method of learning semantic relations, but relies on having features for occurrences of pairs of terms rather than just vectors for terms themselves. Our approach is therefore more generally applicable to systems which compose distributional representations of meaning.

Most recently, Rei and Briscoe (2013) note that hyponyms are well suited for lexical substitution. In their experiments with smoothing edge scores for parser lexicalisation, they find that a directional similarity measure, *WeightedCosine*[2], performs best. Also of note, Mikolov et al. (2013) propose a vector offset method to capture syntactic and semantic regularities between word representations learnt by a recurrent neural network language model. Yih et al. (2012) present a method for distinguishing synonyms and antonyms by inducing polarity in a document-term matrix before applying Latent Semantic Analysis. Santus et al. (2014) propose identifying hypernyms using a new measure based on entropy, SLQS, which is based on the hypothesis that the most typical linguistic contexts of a hypernym are less informative than the most typical linguistic contexts of its hyponyms. Evaluated on pairs extracted from the BLESS dataset (Baroni and Lenci, 2011), this measure outperforms P_{ww} at both discriminating hypernym test pairs from other types of relation and at determining the direction of the entailment relation.

3 Methodology

The code used to perform our experiments has been open sourced, and is available online.[3]

3.1 Vector Representations

Distributional information was collected for all of the nouns from Wikipedia provided they had occurred 100 or more times. We used a Wikimedia dump of Wikipedia from June 2011 and extracted text using wp2txt[4]. This was part-of-speech tagged, lemmatised and dependency parsed using the Malt Parser (Nivre, 2004). All major grammatical dependency relations involving open class parts of speech (`nsubj, dobj, iobj, conj, amod, nnmod`) and also occurring 100 or more times were extracted as features of the POS-tagged and lemmatised nouns. The value of each feature is the positive point wise mutual information (PPMI) (Church and Hanks, 1989) between the noun and the feature. The total number of noun vectors which can be harvested from Wikipedia with these parameters is $124,345$.

Our goal is to build classifiers that establish whether or not a given semantic relation, *rel*, holds between two similar words A and B. Support vector machines (SVMs), which are effective across a variety of classification scenarios, learn a boundary between two classes from a set of positive and negative example vectors. The two classes correspond to the relation *rel* holding or not holding. Here, however, we do not start with a single vector, but with two distributional vectors v_A and v_B for the words A and B, respectively. These vectors must be combined in some way to produce the SVM's input, and a number of ways were considered, defined in Table 2. Of these operations, the vector difference (used by *svmDIFF* and *knnDIFF*) and direct sum (used by *svmCAT*) are asymmetric, whereas the sum and pointwise multiplication (used by *svmADD* and *svmMULT*) are symmetric.

We now motivate the use of each of these operations. First, we note that pointwise multiplication (*svmMULT*) is intersective. Similar vectors will have a large intersection and it might be possible to learn the features that nouns occurring in different semantic relations should share. However, it does not retain any information about non-shared features and it is symmetric so it is difficult to see how it would be possible to use it to distinguish hypernyms from hyponyms. Pointwise addition (*svmADD*) effectively performs the union of the features, giving emphasis to the shared features. Whilst it does retain information about the non-shared features, it is also symmetric, making it difficult again to see how it would be useful in determining the direction of an entailment relation

Vector difference (as used in *svmDIFF* and *knnDIFF*), on the other hand, is asymmetric. Further, we might expect a small difference vector (containing many zeroes) to be indicative of similar nouns. Further, considering the majority sign of features in this difference vector might indicate the direction of

[2]The details of this measure are unpublished.

[3]`https://github.com/SussexCompSem/learninghypernyms`

[4]`https://github.com/yohasebe/wp2txt`

entailment. Using an SVM, we might expect to be able to effectively learn which of these features should be ignored and which should be combined, to decide the correct direction of entailment in the majority number of cases in our training data. However, note that if one uses vector difference it is impossible to distinguish between the case where a feature occurred with both nouns (to the same extent) and the case where a feature occurs with neither noun. Accordingly, a small difference vector may indicate that both nouns do not occur in many distinct contexts. A possible solution to this problem is to use the direct sum of the vectors (i.e., the concatenation of the two vectors) which retains all of the information from the original vectors. Finally, we consider the use of the single vector corresponding to the second word (*svmSING*) as a baseline. High performance by this operation would indicate that we can learn features of words which tend to be hypernyms (or co-hyponyms) without any regard to the other word in the putative relationship.

We also note that the behaviour of these methods may differ depending on the weighting used for vectors. For example, PMI is the log of a ratio of probabilities and therefore one might expect vector addition where vectors are weighted using PMI to correspond to multiplication where vectors are weighted using frequency or probability. However, the use of *positive* PMI (where negative PMI scores are regarded equal to zero), which is consistent with other work in this area, means that this correspondence is lost.

Because of the nature of our datasets, we were concerned that systems could learn information about the taxonomy from the relations in the training data, without making use of information in the vectors themselves. To investigate this, we constructed random vectors to be used in place of the vectors derived from Wikipedia. The dimensionality of the random vectors was chosen to be 1000 since this substantially exceeds the average number (398) of non-zero features in the Wikipedia vectors.

3.2 Classifiers

We constructed linear SVMs for each of the vector operations outlined in Section 3.1. We used linear SVMs for speed and simplicity, since the point is to compare the different vector representations of the pairings. For comparison, we also constructed a number of supervised, unsupervised, and weakly supervised classifiers. These are listed in Table 2. For the linear SVMs and kNN classifier, we used the scikit-learn implementations with default settings. For k nearest neighbours, we performed a parameter search, using nested cross-validation, varying k between 1 and 50.

For weakly supervised approaches, we evaluated the measure on the training set, then found the best threshold p on the training set that best divides the two classes using that measure. When classifying, we determine that the relation holds if the value of the measure exceeds p.

svmDIFF	A linear SVM trained on the vector difference $v_B - v_A$
svmMULT	A linear SVM trained on the pointwise product vector $v_B * v_A$
svmADD	A linear SVM trained on the vector sum $v_B + v_A$
svmCAT	A linear SVM trained on the vector concatenation $v_B \oplus v_A$
svmSING	A linear SVM trained on the vector v_B
knnDIFF	k nearest neighbours (knn) trained on the vector difference $v_B - v_A . 1 < k < 50$
widthdiff	$\textit{width}(B) > \textit{width}(A) \rightarrow \textit{rel}(A, B)$ where $\textit{width}(A)$ is number of non-zero features in A
singlewidth	$\textit{width}(B) > p \rightarrow \textit{rel}(A, B)$
cosineP	$\textit{sim}_{cos}(A, B) > p \rightarrow \textit{rel}(A, B)$ where $\textit{sim}_{cos}(A, B)$ is cosine similarity using PPMI
linP	$\textit{sim}_{lin}(A, B) > p \rightarrow \textit{rel}(A, B)$ (Lin, 1998)
CRdiff	$P_{ww}(A, B) > R_{ww}(A, B) \rightarrow \textit{rel}(A, B)$ (Weeds et al., 2004)
clarkediff	$P_{cl}(A, B) > R_{cl}(A, B) \rightarrow \textit{rel}(A, B)$ (Clarke, 2009)
invCLP	$\textit{invCL}(A, B) > p \rightarrow \textit{rel}(A, B)$ (Lenci and Benotto, 2012)
balAPincP	$\textit{balAPinc}(A, B) > p \rightarrow \textit{rel}(A, B)$ (Kotlerman et al., 2010)
most freq	The most frequent label in the training data is assigned to every test point.

Table 2: Implemented classifiers

3.3 Data Sets

One of key the challenges of this work has been to construct a data set which accurately and validly tests our hypotheses. All four of our datasets detailed below are available online [5].

In order to test our hypotheses, a data set needs to be balanced in many respects in order to prevent the supervised classifiers making use of artefacts of the data. This would not only make it unfair to compare the supervised approaches with the unsupervised approaches, but also make it unlikely that our results would be generalisable to other data. Here, we outline the requirements for the data sets, the importance of which is demonstrated by our initial results for a data set which does not satisfy all of them.

There should be an equal number of positive and negative examples of a semantic relation. Thus, random guessing or labelling with the most frequently seen label in the training data will yield 50% accuracy and precision. An advantage of incorporating this requirement means that evaluation can be in terms of simple accuracy (or error rate).

It should not be possible to do well simply by considering the distributional similarity of the terms. Hence, the negative examples need to be pairs of equally similar words, but where the relationship under consideration does not hold.

It should not be possible to do well by pre-supposing an entailment relation and guessing the direction. For example, it has been shown (Weeds et al., 2004) that given a pair of entailing words selected from WordNet, over 70% of the time the more frequent word is also the entailed word.

It should not be possible to do well using ontological information learnt about one or both of the words from the training data that is not generalisable to their distributional representations. For example, it should not be possible for the classifier simply to learn directly from the training pairs $\langle$`cat` ISA `mammal`$\rangle$ and $\langle$`mammal` ISA `animal`$\rangle$ that $\langle$`cat` ISA `animal`$\rangle$. Furthermore, we must ensure that a classifier cannot learn that a particular word is near the top of the ontological hierarchy, and, as a result, do well by guessing that a particular pairing probably has an entailment relation. For example, given many pairs such as $\langle$`cat` ISA `animal`$\rangle$, $\langle$`dog` ISA `animal`$\rangle$, a system which guessed $\langle$`rabbit` ISA `animal`$\rangle$ but not $\langle$`animal` ISA `rabbit`$\rangle$ would do better than random guessing. Whilst both of these types of information could be useful in a hybrid system, they do not require any distributional information and therefore we would not be learning anything about the distributional features of `animal` which make it likely to be a hypernym.

3.3.1 BLESS

We have constructed two data sets from BLESS (Baroni and Lenci, 2011) which is a collection of examples of hypernyms, co-hyponyms, meronyms and random unrelated words for each of 200 concrete, largely monosemous nouns. We will refer to these 200 nouns as the BLESS concepts.

$hyponym_{\text{BLESS}}$ is a set of 1976 labelled pairs of nouns. For each BLESS concept, 80% of the hypernyms were randomly selected to provide positive examples of entailment. The remaining hypernyms for the given concept were reversed and taken with the same number of co-hyponyms, meronyms and random words to form negative examples of entailment. A filter was applied to ensure that duplicate pairs were not included (e.g., if $\langle$`cat`, `animal`$\rangle$ is a positive pair then $\langle$`animal`, `cat`$\rangle$ cannot be a negative pair).

$cohyponym_{\text{BLESS}}$ is a set of 5835 labelled pairs of nouns. For each BLESS concept, the co-hyponyms were taken as positive examples of this relation. The same total number of (and split evenly between) hypernyms, meronyms and random words was taken to form the negative examples. The order of 50% of the pairs was reversed and again duplicate pairs were disallowed.

In both cases the pairs are labelled as positive or negative for the specified semantic relation and in both cases there are equal (± 1) numbers of positive and negative examples. For 99% of the generated BLESS pairs, both nouns had associated vectors harvested from Wikipedia. If a noun does not have an associated vector, the classifiers use a zero vector.

[5] `https://github.com/SussexCompSem/learninghypernyms`

3.3.2 WordNet

We constructed two data sets using WordNet. Whilst these data sets are similar in size to the BLESS data sets they more adequately satisfy the requirements laid out above[6]. We constructed a list of all non-rare, largely monosemous, single word terms in WordNet. To be considered non-rare, a word needed to have occurred in SemCor at least once (i.e. frequency information is provided about it in the WordNet package) and to have occurred in Wikipedia at least 100 times. To be considered largely monosemous, the predominant sense of the word needed to account for over 50% of the occurrences in the SemCor frequency information provided with WordNet. This led to a list of 7613 nouns.

hyponym$_{\text{WN}}$ is a set of 2564 labelled pairs of nouns constructed in the following way. Pairs $\langle A, B \rangle$ were found in the list of nouns where B is an ancestor of A (i.e., A lexically entails B). Each found pair is added either as a positive or a negative in the ratio 2:1 provided that the reverse pairing has not already been added and provided that each word has not previously been used in that position. Co-hyponym pairs (i.e., words which share a direct hypernym) were also found within the list of nouns. Each found pair is added to the data set (as a negative) provided the reverse pairing has not already been added, and provided that neither word has already been seen in that position in a pairing (either in the entailment pairs or the co-hyponym pairs). The same number of co-hyponym pairs as hypernym-hyponym negatives is selected. This provides a balanced data set where half of the pairs are positive examples of entailment and the other half are semantically similar but not entailing.

cohyponym$_{\text{WN}}$ is a set of 3771 labelled pairs of nouns. It was constructed in the same way as *hyponym*$_{\text{WN}}$ except the same number of co-hyponym pairs were selected as the total number of entailment pairs (in either direction). These co-hyponym pairs were labelled as positive and the entailment pairs were labelled as negative. Thus, this provides a balanced data set where half of the pairs are positive examples of co-hyponyms and the other half, the negative examples, are entailment pairs (with direction unspecified)

In both these sets, the average path distance between entailment pairs is 1.64, whereas path distance between co-hyponym pairs is 2.

3.4 Experimental Setup

Most of our experiments were carried out using an implementation of five-fold cross-validation using each combination of data set, vector set and classifier. In this setup, the pairs are randomly partitioned into five subsets, one subset is held out for testing whilst the classifiers are trained on the remaining four, and this process is repeated using each subset as the test set.

In initial experiments with the BLESS datasets, the SVM classifiers were able to achieve classification accuracy of over 95% for *hyponym*$_{\text{BLESS}}$ and over 90% for *cohyponym*$_{\text{BLESS}}$. However, the results using random vectors were not significantly different from using the distributional vectors harvested from Wikipedia. This indicated that the classifiers were learning ontological information implicit in the training data. In order to address this, when using the BLESS datasets, we removed any pair from the training data if either word was present in the test data. In order to preserve a reasonable amount of training data, we implemented this approach with ten-fold cross-validation. In all subsequent experiments, across all datasets and classifiers, we found performance by the random vectors was no higher than 52%. This indicates that the performance seen in Table 3 is due to learning from distributional features rather than any ontological information implicit in the training set.

4 Results

In Table 3, we compare average accuracy for a number of different classifiers on each of two tasks, distinguishing hyponyms and distinguishing co-hyponyms, on each of the two datasets.

Looking at the results for the *hyponym*$_{\text{BLESS}}$ data set, we can see that the SVM methods do generally outperform the unsupervised methods. However, the best performing model is svmSING, suggesting that, for this data set, it is best to try to learn the distributional features of more general terms, rather than comparing the vector representations of the two terms under consideration.

[6]Note that imposing these requirements on the BLESS data sets would lead to very small data sets, since information is only provided for 200 nouns.

dataset	svmDIFF	svmMULT	svmADD	svmCAT	svmSING	knnDIFF		
$hyponym_{\text{BLESS}}$	**0.74**	0.56	0.66	0.68	**0.75**	0.54		
$cohyponym_{\text{BLESS}}$	0.62	0.39	0.41	0.40	0.40	0.58		
$hyponym_{\text{WN}}$	**0.75**	0.45	0.37	**0.74**	0.69	0.50		
$cohyponym_{\text{WN}}$	0.37	0.60	**0.68**	**0.64**	0.58	0.50		
dataset	most freq	cosineP	linP	widthdiff	singlewidth	CRdiff	invCLP	balAPincP
$hyponym_{\text{BLESS}}$	0.54	0.53	0.54	0.56	0.58	0.52	0.54	0.54
$cohyponym_{\text{BLESS}}$	0.61	**0.79**	**0.78**	-	-	-	-	-
$hyponym_{\text{WN}}$	0.50	0.53	0.52	0.70	0.65	0.70	0.66	0.53
$cohyponym_{\text{WN}}$	0.50	0.50	0.55	-	-	-	-	-

Table 3: Accuracy Figures for the data sets generated from BLESS and WordNet (standard errors $<$ 0.02). For cohyponyms, results for measures designed to detect hyponymy have been omitted. We also omit results of clarkediff as these were consistently the same or less than CRdiff.

On the corresponding co-hyponym task, using the $cohyponym_{\text{BLESS}}$ data set, we see the best performing classifier is the cosine measure. The cosine measure is able to perform relatively well here because a substantial proportion of the negative examples (25%) are random unrelated words which will have low cosine scores. It is also consistent with earlier work (e.g., (Lenci and Benotto, 2012)) which suggests that measures such as the cosine measure "prefer" words in symmetric semantic relationships such as co-hyponymy. The poor performance of the SVM methods here can perhaps be explained by the paucity of the training data in this experimental set up with this data set. If, for example, our test concept is `robin`, our approach requires that we will not have any training pairs containing `robin`, or any training pairs containing any of the words to which `robin` is related in the test set. In a dataset as small as BLESS, this requirement effectively removes all knowledge of the distributional features of words in the target domain. Hence, the need for a larger dataset as we have extracted from WordNet.

Looking at the results for the $hyponym_{\text{WN}}$ data set, the directional SVM methods (*svmDIFF* and *svmCAT*) substantially outperform the symmetric SVM methods, and their performance is significantly better (at the 0.01% level) than the unsupervised methods. Also of note is the substantial difference between *svmDIFF* and *knnDIFF*. Both of these methods are trained on the differences of vectors. However, the linear SVM outperforms kNN by 19–25%. This may suggest that the shape of the vector space inhabited by the positive entailment pairs is particularly conducive for learning a linear SVM. Positive and negative pairs are close together (as evidenced by the poor performance of *kNN*), but generally linearly separable.

Looking at the results for the $cohyponym_{\text{WN}}$ data set, it is clear that the unsupervised methods cannot distinguish the co-hyponym pairs from the entailing pairs. The supervised SVM methods do substantially better, with the best performance achieved by *svmADD* and *svmCAT*. Both of these methods essentially retain information about all of the features of both words. *svmMULT* does much better than *svmDIFF*, which suggests that the shared features are more indicative than the non-shared features for this task.

The reasonably high performance of *svmSING* on both data sets suggests that words which have co-hyponyms in the data set tend to inhabit a somewhat different part of the feature space to words which are included as entailed words in the data set. We hypothesise that there are specific features which more general words tend to share (regardless of their topic) which makes it possible to identify more general words from more specific words. This is completely consistent with very recent results using SLQS, a new entropy-based measure (Santus et al., 2014). Here, the authors hypothesise that the most typical contexts of a hypernym are less informative than the most typical linguistic contexts of its hyponyms, with some promising results. It would be plausible to hypothesise that *svmSING* is learning which nouns typically have less informative contexts and are therefore likely to by hypernyms.

Given prior work, the performance of the *balAPincP* measure is lower than expected on the $hyponym_{\text{WN}}$ dataset. Our task is slightly different to that of (Kotlerman et al., 2010), since we are determining the existence (or not) of hyponymy, rather than the direction of entailment for pairs where it is known that a relationship exists. It could be that the measure is particularly suited to the latter task.

5 Conclusions and Further Work

We have shown that it is possible to predict to a large extent whether or not there is a specific semantic relation between two words given their distributional vectors, using a supervised approach based on linear SVMs. The increase in accuracy over unsupervised methods is significant at the 0.01% level and corresponds to a substantial absolute reduction in error rate (over 15%).

We have also shown that the choice of vector operation is significant. Whilst concatenating the vectors, and therefore retaining all of the information from both vectors including direction, generally performs well, we have also shown that different vector operations are useful in establishing different relationships. In particular, the vector difference operation, which loses information about the original vectors, achieved performance indistinguishable from concatenation on the entailment task, where the classifier is required to distinguish hyponyms from other semantically related words including hypernyms. On the other hand, the addition operation, which also loses information, outperformed concatenation by 4% (which is statistically significant at the 0.01% level) on the coordinate task, where the classifier is required to distinguish co-hyponyms from hyponyms and hypernyms. Hence the nature of the relationship one is trying to establish between words determines the nature of the operation one should perform on their associated vectors.

We have also shown that it is possible to outperform state-of-the-art unsupervised methods even when a data set has been constructed without ontological information, and when target words have not previously been seen in that position of a relationship in the training data. Hence, we believe the supervised methods are learning characteristics of the underlying feature space which are generalisable to new words (inhabiting the same feature space).

In future work, we intend to apply this approach to the problem of labelling the distributional neighbours found for a given word with specific semantic relations. We also plan to investigate the use of bag-of-words (windowed) vectors instead of grammatical relations for this task.

Finally, we believe that the data sets constructed from WordNet, which we publish alongside this paper, can be used as a useful benchmark in evaluating future advances in this area, both for supervised and unsupervised methods.

Acknowledgements

This work was funded by UK EPSRC project EP/IO37458/1 “A Unified Model of Compositional and Distributional Compositional Semantics: Theory and Applications”.

References

Marco Baroni and Alessandro Lenci. 2011. How we BLESSed distributional semantic evaluation. In *Proceedings of the GEMS 2011 workshop on Geometric Models of Natural Language Semantics, EMNLP 2011*.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32. Association for Computational Linguistics.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2010. Global learning of focused entailment graphs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1220–1229, Uppsala, Sweden, July. Association for Computational Linguistics.

Shane Bergsma, Aditya Bhargava, Hua He, and Grzegorz Kondrak. 2010. Predicting the semantic compositionality of prefix verbs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 293–303, Cambridge, MA, October. Association for Computational Linguistics.

Danushka Bollegala, David Weir, and John Carroll. 2011. Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*.

Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th Annual Meeting on Association for Computational Linguistics*, ACL '89, pages 76–83, Stroudsburg, PA, USA. Association for Computational Linguistics.

Daoud Clarke. 2009. Context-theoretic semantics for natural language: an overview. In *Proceedings of the Workshop of Geometric Models for Natural Language Semantics*.

James Curran. 2004. *From Distributional to Semantic Similarity*. Ph.D. thesis, University of Edinburgh.

Georgiana Dinu and Stefan Thater. 2012. Saarland: Vector-based models of semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*.

Christaine Fellbaum, editor. 1989. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, Massachusetts.

Trevor Fountain and Mirella Lapata. 2012. Taxonomy induction using hierarchical random graphs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 466–476, Montréal, Canada, June.

Maayan Geffet and Ido Dagan. 2005. Lexical entailment and the distributional inclusion hypothesis. In *Proceedings of the 43rd meeting of the Association for Computational Liuguistics (ACL)*, pages 107–114.

Edward Grefenstette, Mehrnoosh Sadrzadeh, Stephen Clark, Bob Coecke, and Stephen Pulman. 2011. Concrete sentence spaces for compositional distributional models of meaning. *Proceedings of the 9th International Conference on Computational Semantics (IWCS 2011)*, pages 125–134.

Zelig Harris. 1954. Distributional structure. *Word*, 10:146–162.

Mitesh Khapra, Anup Kulkarni, Saurabh Sohoney, and Pushpak Bhattacharyya. 2010. All words domain adapted WSD: Finding a middle ground between supervision and unsupervision. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1532–1541, Uppsala, Sweden, July.

Adam Kilgarriff and Colin Yallop. 2000. What's in a thesaurus? In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC2000)*.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Special Issue of Natural Language Engineering on Distributional Lexical Semantics*, 4(16):359–389.

Lillian Lee. 1999. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, College Park, Maryland, USA, June.

Alessandro Lenci and Giulia Benotto. 2012. Identifying hypernyms in distributional semantic spaces. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*Sem)*.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING 1998)*.

Tara McIntosh. 2010. Unsupervised discovery of negative categories in lexicon bootstrapping. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 356–365, Cambridge, MA, October. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June.

Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. 2012. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. In *Proceedings of COLING 2012*, pages 1781–1796, Mumbai, India, December.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June. Association for Computational Linguistics.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the ACL workshop on Incremental Parsing*, pages 50–57.

Marek Rei and Ted Briscoe. 2013. Parser lexicalisation through self-learning. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 391–400, Atlanta, Georgia, June. Association for Computational Linguistics.

Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte Im Walde. 2014. Chasing hypernyms in vector spaces with entropy. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 38–42, Gothenburg, Sweden, April.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17*.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 801–808. Association for Computational Linguistics.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211.

György Szarvas, Chris Biemann, and Iryna Gurevych. 2013. Supervised all-words lexical substitution using delexicalized features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1131–1141, Atlanta, Georgia, June.

Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 849–856, Manchester, UK, August.

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In Michael Collins and Mark Steedman, editors, *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 81–88.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of Coling 2004*, pages 1015–1021, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Julie Weeds, David Weir, and Jeremy Reffin. 2014. Distributional composition using higher-order dependency vectors. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality, EACL 2014*, Gothenburg, Sweden, April.

Dominic Widdows. 2008. Semantic vector products: Some initial investigations. In *Proceedings of the Second Symposium on Quantum Interaction, Oxford, UK*, pages 1–8.

Wen-tau Yih, Geoffrey Zweig, and John Platt. 2012. Polarity inducing latent semantic analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1212–1222, Jeju Island, Korea, July. Association for Computational Linguistics.

Chen Zhang and Joyce Chai. 2010. Towards conversation entailment: An empirical investigation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 756–766, Cambridge, MA, October. Association for Computational Linguistics.

"One Entity per Discourse" and "One Entity per Collocation" Improve Named-Entity Disambiguation

Ander Barrena*, Eneko Agirre*, Bernardo Cabaleiro, Anselmo Peñas**, Aitor Soroa***

*IXA NLP Group / University of the Basque Country, Basque Country
abarrena014@ikasle.ehu.es, e.agirre@ehu.es, a.soroa@ehu.es

**UNED NLP & IR Group, Madrid
anselmo@lsi.uned.es, bcabaleiro@lsi.uned.es

Abstract

The "one sense per discourse" (OSPD) and "one sense per collocation" (OSPC) hypotheses have been very influential in Word Sense Disambiguation. The goal of this paper is twofold: (i) to explore whether these hypotheses hold for entities, that is, whether several mentions in the same discourse (or the same collocation) tend to refer to the same entity or not, and (ii) test their impact in Named-Entity Disambiguation (NED). Our experiments show consistent results on different collections and three state-of-the-art NED system. OSPD hypothesis holds in around 96%-98% of documents whereas OSPC hypothesis holds in 91%-98% of collocations. Furthermore, a simple NED post-processing in which the majority entity is promoted, produces a gain in performance in all cases, reaching up to 8 absolute points of improvement in F-measure. These results show that NED systems would benefit of considering these hypotheses into their implementation.

1 Introduction

The "one sense per discourse" (OSPD) hypothesis was introduced by Gale et al. (1992), and stated that a word tends to preserve its meaning when occurring multiple times in a discourse. They estimated that the probability of two occurrences of the same polysemous noun drawn from one document having the same sense to be around 94% for documents from Grolier encyclopedia, and 96% for documents from Brown, based on word senses from the Oxford Advanced Learner's Dictionary and a handful of examples. A few years later, Krovetz (1998) reported 66% on larger corpora (SemCor and DSO) annotated with WordNet senses by third parties, but, unfortunately, he only reported how many polysemous nouns occurred with a single sense in **all** documents, not in each document. In the context of statistical machine translation, Carpuat (2009) reported that, 80% of the time, words occurring multiple times in a source document are translated into a single word in the target language.

In the case of entities, OSPD is closely related to coreference, where the task is to find whether two different mentions (perhaps using different surface strings like "John" and "he") in a document refer to the same entity or not. For instance, the coreference system presented by (Lee et al., 2013), uses a heuristic which links mentions in a document that share the same surface string: "This sieve [heuristic] accounts for approximately 16 CoNLL F1 points improvement, which proves that a signicant percentage of mentions in text are indeed repetitions of previously seen concepts". Our paper actually quantifies the amount of those repetitions for entities, providing additional evidence for the heuristic.

The "one sense per collocation" (OSPC) hypothesis was introduced by Yarowsky (1993), stating that a word tends to preserve its meaning when occurring with the same collocate. Yarowsky tested his hypothesis for several definitions of collocate, including positional collocates (word to left or right) and syntactic collocations (governing verb of object, governing verb of subject, modifying adjective). He reported entropy on train data, as well as disambiguation performance on unseen data, with the precision ranging between 90% and 99% for a handful of words with two distinct homograph senses, like, e.g. "bass" or "colon". In larger-scale research, Martinez and Agirre (2000) measured the precision

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2260–2269, Dublin, Ireland, August 23-29 2014.

Abbott Beefs Up Litigation Reserves NORTH CHICAGO, Ill. (AP) **Abbott** Laboratories Inc., bracing for a costly settlement in a federal investigation involving the prostate-cancer drug Lupron, said Friday it was increasing litigation reserves by $344 million. As part of the announcement, **Abbott** said it had restated its quarterly results and is now reporting a loss of $319.9 million for the first three months of this year rather than a profit. The move comes amid long-running negotiations between the U.S. Department of Justice and TAP Pharmaceutical Products, the 50-50 joint venture between **Abbott** and Takeda Chemical Industries of Japan that made Lupron. **Abbott** said in January ...

Figure 1: Example of OSPD for entities. All occurrences of "**Abbott**" refer to "Abbott Laboratories".

of similar collocations on corpora (Semcor and DSO) annotated by third parties with finer-grained senses from WordNet, reporting lower figures around 70%.

In this paper, we take a collocation to be a word (or multiword term) that co-occurs with the target named-entity more often than would be expected by chance. In our case we use syntactic dependencies to extract co-occurring terms.

These two hypotheses have been very influential, and have inspired multiple heuristics and methods in Word Sense Disambiguation research (Agirre and Edmonds, 2007, Chapters 5,7,10,11). In this work we are going to show that both hypotheses hold for named-entities as well, and that the hypotheses can be used to post-process the output of any Named-Entity Disambiguation system (NED) to improve its performance. NED, also known as Entity Linking, takes as input a named-entity mention in context and assigns it a specific entity from a given entity repository (Hachey et al., 2012; Daiber et al., 2013).

In the first part of this work we are going to test whether the two hypotheses hold for entity mentions with respect to a repository of entities extracted from Wikipedia. For instance, do all occurrences of mention "Abbott" in a document refer to the same entity? Do all occurrences of mention "CPI" as subject of verb "rise" refer to the same entity? Do all occurrences of "CDU" in relation to "Merkel" refer to the same entity? The examples in Figures 1 and 2 show evidence that this is indeed the case. The experiments aim at quantifying in which degree OSPD and OSPC hypotheses hold for entities[1].

In the second part of the paper, we will explore a simple method to incorporate OSPD and OSPC hypotheses to any existing NED system, showing their potential. After running the NED system, we take its output and observe, for each mention string, which is the entity returned most often for a given document (or collocation), assigning to all occurrences the majority entity. We tested the improvements with a freely available NED system (Daiber et al., 2013), a reimplementation of a strong Bayesian NED system (Han and Sun, 2011) and an in-house graph-based system. We got statistically significant improvements for all systems and "one sense" hypotheses that we tested, with a couple exceptions.

In order to check the OSPD and OSPC hypotheses for entities, we first looked into existing datasets. AIDA (Hoffart et al., 2011)[2] is a publicly available hand-tagged corpus based on the CoNLL named-entity recognition and disambiguation task dataset. AIDA contains links of all entity mentions in full documents, so it is a natural fit for OSPD. We estimated OSPD based on more than 4,000 mentions that occur multiple times in a document. For completeness, we also estimated OSPD at the collection level.

OSPD and OSPC are independent of each other, as one is applied at the document level and the other at the corpus level, focusing on the entities that occur with a specific collocation. Multiple occurrences of a target string in a document usually occur with different collocations, and conversely, multiple occurrences of a target string with a specific collocation typically occur in different documents. Note also that singletons (entities that are only mentioned once in a document) are not affected by OSPD, but could be affected by OSPC.

In order to estimate OSPC, no available corpus existed, so we decided to base our dataset on the TAC KBP 2009 Entity Linking dataset[3] (TAC2009 for short) (Ji et al., 2010). The TAC2009 dataset involves 138 mention strings, which have been annotated in several documents drawn primarily from Gigaword[4].

[1] For the sake of clarity we will also refer to OSPD and OSPC for entities as OSPD and OSPC.

[2] http://www.mpi-inf.mpg.de/yago-naga/aida/downloads.html

[3] http://www.nist.gov/tac/2013/KBP/EntityLinking/index.html

[4] http://catalog.ldc.upenn.edu/LDC2003T05

CPI subject-of rise:
China's consumer price index, or **CPI**, rose 2.8 percent last December.
In the 10 months to October, the **CPI** rose 1.35 percent, the core price index grew 1.13 percent ...
Measured on a month-on-month basis, March **CPI** rose 2.3 percent from February, ...
... still lower than in China, Hong Kong and Singapore, whose **CPI**s have rised 8.0 percent, ...
The core **CPI** rose 0.2 percent, in line with Wall Street expectations.
Angela Merkel has **CDU**:
... who share power with Merkel's **CDU** nationally in an uneasy " grand coalition " ...
Economy Minister Michael Glos, also from the CSU, the sister party to Merkel's **CDU** ...
In the past Merkel's **CDU** had been able to rely on the CSU's strength in Bavaria ...
... but while her conservative **CDU** wanted new legal tools to do so, ...
The new development has put a further strain on Merkel's **CDU** ...

Figure 2: Examples of OSPC for entities, showing five examples for a syntactic collocation (top row) and fie examples for a more specific proposition (bottom row). "**CPI**" might refer to "Comunist Party of India" or "Consumer Price Index", among others, but refers to the second in all cases. "**CDU**" can refer to the German "Christian Democratic Union" or "Catholic Distance University", among others, but refers to the first in all cases.

We extracted several syntactic collocations for those 138 mention strings from Gigaword, and hand-annotated them, yielding an estimate for the OSPC. Note that TAC2009 only provides the annotation for a specific mention in a document, so we had to annotate by hand the rest of occurrences in the documents. For instance, we analyzed examples of "CPI" as subject of the verb "rise" (cf. Figure 2). Some of the syntactic collocations like the subjects of verb "has" seemed very uninformative, so we decided to also check the OSPC hypothesis on more specific collocations, involving more complete argument structures. For instance, we checked "ABC" occurring as subject of "has" with object "radio". We call this more specific collocations *propositions* (Peñas and Hovy, 2010).

The paper is structured as follows. We will first present the resources used in this study. Section 3 presents the results of OSPD. Section 3.1 extends OSPD when, instead of documents, we take the complete collection. Section 4 presents the study of OSPC both for syntactic dependencies and propositions. Section 5 presents the experiments where OSPD and OSPC are used to improve the performance of existing systems. Finally, we draw the conclusions and future work.

2 Resources used

AIDA is based on the corpus used in the CONLL named-entity recognition and classification task, where all entities in full documents had been linked to the referred Wikipedia articles (using the 2010 Wikipedia dump). We use the full AIDA dataset, with 1,393 documents, 34,140 disambiguated entity mentions, where 27,240 are linked to a Wikipedia article. All in all there are 6,877 distinct mention strings (types) which are linked at least once to a Wikipedia article. The rest refer to articles not in Wikipedia (NIL instances), and were discarded. This corpus covers news from a sample of a few days spanning from 1996-05-28 to 1996-12-07.

In order to prepare our dataset for OSPC, we chose the dataset of the TAC KBP 2009 Entity Linking competition, as this dataset have been extensively used in Entity Linking evaluation. In addition, the corpus used in the task was very large, allowing us to mine relevant collocations (see below). We manually annotated the occurrences in the extracted collocations, producing two datasets, one for each kind of collocation (cf. Section 4). Note that the TAC KBP organizers only annotated one specific mention in each target document. For completeness, we also tagged the rest of the occurrences of the target mentions in the documents, thus allowing us to provide OSPD estimated based on TAC2009 data as well. This is the third dataset that we annotated by hand. The hand-annotation was performed by a single person, and later reviewed by the rest of the authors. The three annotation datasets are publicly available[5]. Hand-

[5] http://ixa2.si.ehu.es/OEPDC

NHasN	”U.S. dollar”
NPN	”condition of anonymity”
NVN	”official tells AFP”
NVNPN	”article maintains interest within layout”
NVPN	”others steal from input”
VNPN	”includes link to website”

Table 1: List of the six patterns used to extract propositions, with some examples.

tagging is costly, so we tagged around 250 examples of syntactic collocations and around 250 examples of propositions.

Note that both AIDA and TAC2009 contain mentions that were not linked to a Wikipedia article because the mention referred to an entity which was not listed in the entity inventory. We ignored all those cases (called NIL cases), as we would need to investigate, for each NIL, which actual entity they refer to.

The collocations were extracted from the TAC KBP collection (Ji et al., 2010), comprising 1.7 million documents, 1.3 millions from newswire and 0.5 millions from the web. We have parsed them with the Stanford CoreNLP software (Klein and Manning, 2003), obtaining around 650 million dependencies (De Marneffe and Manning, 2008). We selected subject, object, prepositional complements and adjectival modifiers as the source for syntactic collocations. In order to provide more specific collocations, we implemented the syntactic patterns proposed in (Peñas and Hovy, 2010), which produce so-called propositions. The result is a database with 16 million distinct propositions. Table 1 shows the six patterns used in this work, together with some examples.

In order to know whether a mention is ambiguous, we built a dictionary based on Wikipedia which lists, for each string mention, which entities it can refer to. We followed the construction method of (Spitkovsky and Chang, 2012), which checked article titles, redirects, disambiguation pages and hyperlinks to find mention strings that can be used to refer to entities. Contrary to them, we could not access hyperlinks in the web, so we could use only those in Wikipedia. According to our dictionary, the ambiguity of the mentions that we are studying is very high, 26.4 entities on average for the mentions in AIDA, and 62.6 entities on average for the mentions in TAC2009.

3 One entity per discourse

In order to estimate OSPD we divided the number of times a mention string referred to different entities in the document with the number of times a mention string occurred multiple times in the document. In the denominator and numerator we count each mention-document pair once.

Regarding AIDA, we found 12,084 occurrences of mentions which occurred more than once in a document, making 4,265 unique mention-document pairs[6] (cf. Table 2). In the vast majority of the cases those mentions refer to a single entity in the document, and only in 170 cases the mentions in the document refer to several entities. The last row in Table 2 shows the ratio between those values, 96.01%, showing that OSPD is strong in this dataset.

We also checked OSPD in the TAC2009 dataset. Out of the 138 distinct mention strings used in the task, we discarded those only linked to NIL (that is, no corresponding Wikipedia article existed) and those which were not ambiguous (that is, they had only one entity in the dictionary, cf. Section 2). That leaves 105 mention strings, occurring 1,776 times in 918 different documents, which we annotated by hand. The 105 strings occurred 1,776 times in 918 documents. Removing the cases where the mention occurred only once, we were left with 1,173 occurrences, which make 334 unique mention-document pairs, of which only 6 occurred with more than one sense (rightmost row in Table 2). This yields an estimate for OSPD of 98.2%.

[6]By unique mention-document pairs we mean that we only count once for a mention occurring multiple times in a document. For instance if mention *Smith* occurs 10 times in the whole corpus, 8 times in document *A* and 2 times in document *B*, we count two unique mention-document pairs.

	AIDA	TAC2009
Mention-document pairs	4,265	334
Ambiguous pairs	170	6
OSPD	**96.0%**	**98.2%**

Table 2: One entity per discourse: per document statistics in AIDA and TAC2009 datasets. Pairs stand for the number of unique mention-document pairs. The 4,265 pairs in AIDA correspond to 12,084 occurrences of mentions, and the 334 pairs in TAC2009 correspond to 1,173 occurrences.

	All mentions		First mention	
	AIDA	TAC2009	AIDA	TAC2009
Mention types	3,363	105	2,731	105
Ambiguous types	475	26	454	25
OSPD (collections)	**85.9%**	**75.2%**	**83.4%**	**76.2%**

Table 3: One entity per collection: statistics in AIDA and TAC2009. In the first two columns ("All mentions") we consider all mention types ($3,363$ types in AIDA correspond to $23,726$ occurrences of mentions, and 105 types in TAC2009 correspond to $1,776$ occurrences). In the second two columns ("First mention") we leave only the first mention of each document (in this case, there are $2,731$ mention types in AIDA which correspond to $15,275$ occurrences, and 105 types in TAC2009 corresponding to 941 occurrences).

Finally, we also thought about measuring OSPD on the Wikipedia articles, where many mentions have been manually linked to their respective article. Unfortunately, we noted that Wikipedia guidelines explicitly prevent authors linking a mention multiple times: *Generally, a link should appear only once in an article, but if helpful for readers, links may be repeated in infoboxes, tables, image captions, footnotes, and at the first occurrence after the lead*[7]. The fact that Wikipedia editors did not explicitly state exceptions to the above rule (e.g. for cases where the word or phrase is used to refer to two different articles, thus breaking the OSPD hypothesis) is remarkable, and might indicate that Wikipedia editors had not felt the need to challenge the OSPD hypothesis.

3.1 One entity per collection

We took the opportunity to also explore "one entity per collection", which gives an idea of what is the spread of entities for whole document collections. In this case, there is no need to count mention-document pairs, as there is one single document, the collection, so we estimate the hypothesis according to mention types. The first two columns in table 3 shows that, overall, mentions which occurred more than once in the collection tend to refer to the same entity 85.9% of the time in AIDA, and 75.2% of the time in TAC2009.

As we know that multiple mentions in a document tend to refer to one entity, the second two columns in table 3 offers the statistics when factoring out multiple occurrences of mention in a document, that is, leaving the first mention in each document. The statistics are very similar, with minor variations.

We think that the lower estimate for TAC2009 is an artifact of how the TAC KBP organizers set up the dataset, as they were explicitly looking for cases where the target string would refer to different entities, making the task more challenging for NED systems. This fact does not affect OSPD for documents, as those strings still tend to refer to a single entity per document, but given the need to find occurrences for different entities, the organizers (Ji et al., 2010) did focus on strings occurring with different entities across the document collection. This is in contrast with AIDA, where they tagged all named-entities occurring in the target documents. Had the organizers of TAC2009 focused on a random choice of strings and documents, the one entity per collection would also hold to the high degree exhibited in AIDA, as the genre of most of the documents is also news (as in AIDA).

[7] http://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Linking#What_generally_should_be_linked

	Syn. coll.	Propositions
Mention-collocation pairs	58	61
Ambiguous pairs	5	1
OSPC	**91.4%**	**98.4%**

Table 4: One entity per collocation: statistics for syntactic collocations and propositions. The 58 mention-collocation pairs correspond to 262 occurrences, and the 61 mention-proposition pairs to 279.

4 One entity per collocation

In order to estimate OSPC for **syntactic collocations**, we manually annotated several occurrences of the 138 mention strings of the TAC2009 dataset. Hand-tagging mention entities is a costly process, so we chose (at random) one syntactic dependency relation for each of the 138 mention strings that occurred more than five times in the corpus. We then hand-tagged at random five occurrences of each collocation (cf. Figure 2). This method would provide a maximum of 5 examples for each of the 138 mentions, but after checking the minimum frequency of the collocations, the quality of the context, repeated sentences, mentions that are not ambiguous in the dictionary, and whether the mention could be attached to an entity in the database, the actual number was lower. All in all we found 58 mention-collocation pairs (262 occurrences) for syntactic collocations (cf. middle column in Table 4). Only 5 mentions referred to more than one entity per collocation, yielding that OSPC for syntactic collocation is around 91.4%.

To gather the dataset for **propositions**, we followed the same method as for the syntactic collocations, that is, we chose (at random) one propositions involving one of the 138 mention strings that occurred more than five times in the corpus, and hand-tagged at random five occurrences of each proposition (cf. Figure 2. As with syntactic collocations, we also found a limited number of mentions filling the desired properties. That left 61 mention-collocation pairs (279 occurrences) for propositions (cf. right column in Table 4). Only 1 mention referred to more than one entity per proposition, yielding OSPC for propositions around 98.4%. This shows that the more specific the context is, the stronger is the link between mention and entity.

5 Improving performance

In order to check whether any of the "one sense" hypothesis above could improve the performance of a NED system, we followed a simple procedure: After running the NED system, we take its output and observe, for each mention string, which is the entity returned most often for a given document (or collocation), assigning to all occurrences the majority entity. In case of ties, we return the entity with the highest support from the NED system. We tested the improvements on three NED systems: the freely available DBpedia Spotlight, a reimplementation of a strong Bayesian NED system and a graph-based system.

DBpedia Spotlight is a freely available NED system (Daiber et al., 2013), based on a generative probabilistic model (Han and Sun, 2011). Nowadays it is one of the most widely used NED systems and attains performances close to state-of-the-art (Daiber et al., 2013)We used the default values of the parameters for all the experiments in this paper.

We also tested an in-house reimplementation of the generative probabilistic model presented in (Han and Sun, 2011). This is a state-of-the-art system which got the same accuracy as the best participant (72.0) when evaluated in the non-NIL subset of TAC2013.

UKB is a freely-available system for performing Word Sense Disambiguation and Similarity based on random walks on graphs (Agirre and Soroa, 2009). Instead of using it on WordNet, we represented Wikipedia as a graph, where vertices are the wikipedia articles and edges represents bidirectional hyperlinks among Wikipedia pages, effectively implementing a NED system. We used a Wikipedia dump from 2013 in our experiments. UKB is a competitive, state-of-the-art system which attained a score of 69.0 when evaluated in the non-NIL subset of the TAC2013 dataset.

The input of the systems is the context of each mention to be disambiguated, in the form of a 100 token window centered in the target mention. In NED, the identification of the correct mention to be

Mention in context	Entity
Abbott Beefs Up Litigation ...	Abbot_Kinney
Abbott Laboratories Inc., bracing ...	Abbott_Laboratories
Abbott said it had restated ...	Abbott_Laboratories
venture between **Abbott** and Takeda ...	Abbott_Laboratories
Abbott said in January ...	Abbott_Laboratories

Figure 3: Applying OSPD: Each of the five occurrences of Abbott in the document in Figure 1 has been tagged independently by a NED systems, which return the correct entity in all but one case (precision 80%). Applying OSPD would return the correct entity (Abbott_Laboratories) in all cases, improving precision to 100%.

	AIDA			TAC 2009		
	Prec.	**Recall**	**F1**	**Prec.**	**Recall**	**F1**
Spotlight	83.24	63.90	72.30	64.48	46.44	53.99
+ OSPD Discourse	**84.17**	70.01	76.44	**64.65**	**48.50**	**55.42**
+ OSPD Collection	84.02	**74.64**	**79.05**	56.24	47.98	51.78
UKB	70.09	69.03	69.55	67.70	67.64	67.67
+ OSPD Discourse	71.30	70.23	70.76	**70.21**	**70.21**	**70.21**
+ OSPD Collection	**75.79**	**74.64**	**75.21**	68.84	68.84	68.84
(Han and Sun, 2011)	65.71	65.11	65.41	65.49	65.49	65.49
+ OSPD Discourse	67.77	67.37	67.57	66.27	66.27	66.27
+ OSPD Collection	**74.29**	**73.89**	**74.09**	**68.24**	**68.24**	**68.24**

Table 5: Applying OSPD: NED performance on AIDA and TAC2009 OSPD datasets, including each of the three NED systems, and the results after applying OSPD at the document and collections levels. Bold marks best result for each system.

disambiguated is part of the problem. AIDA does provide gold mentions, but TAC2009 only provides a query string which might be just a substring of the real mention in the document. We treated both corpus in the same way. In the case of DBpedia Spotlight we use the built-in mention spotter. In the case of our in-house implementations, we use the longest string that matches a valid entity mention in the system, as given by the dictionary (cf. Section 3).

Some of the NED systems do not return an entity for all mentions, so we evaluate precision, recall and the harmonic mean (F1 measure). Statistical significance has been estimated using Wilcoxon. We reused the same corpora as in the previous sections for the evaluation, and also removed all NIL mentions (i.e. mentions which refer to an entity not in Wikipedia).

5.1 One entity per discourse

We report the improvements using OSPD for both **document** and **collection** levels. At the document level, we relabel mentions that occur multiple times in a document using the entity returned most times by the NED system in that document. Figure 3 illustrates the idea for a NED system on the same sample document as in Figure 1. At the collection level, we relabel mentions using the entity returned most times by the NED systems in the whole collection.

Table 5 reports the results of the performance as evaluated on mentions occurring multiple times in the AIDA and TAC2009 datasets. The numbers in the left part of the table correspond to the performance as evaluated on mentions occurring multiple times in AIDA documents. Note that the number of occurrences where OSPD at the collection level can be applied is larger (a superset of those for OSPD at the document level), as, for instance, a mention string occurring once in three different documents won't be affected by OSPD at the document level, but it could be relabeled at the collection level. We were especially interested in making the numbers between OSPD at the document and collection levels

CPI subject-of rise	Angela Merkel has CDU:
Consumer_price_index	Christian_Democratic_Union_(Germany)
Consumer_price_index	Catholic_Distance_University
Communist_Party_of_India	Christian_Democratic_Union_(Germany)
Communist_Party_of_India	Christian_Democratic_Union_(Germany)
Consumer_price_index	Christian_Democratic_Union_(Germany)

Figure 4: Applying OSPC: A NED system system tagged each example in Figure 2 independently. For CPI, the precision is 60%, but after relabeling with OSPC it would be 100%. For CDU, the improvement is from 80% to 100%.

	Syntactic collocations			Propositions		
	prec.	**recall**	**F1**	**prec.**	**recall**	**F1**
Spotlight	82.46	66.41	73.57	74.67	60.22	66.67
+ OSPC	**82.63**	**67.18**	**74.11**	**74.79**	**62.72**	**68.23**
UKB	75.86	75.57	75.72	67.87	67.38	67.63
+ OSPC	**78.54**	**78.24**	**78.39**	**68.59**	**68.10**	**68.35**
(Han and Sun, 2011)	75.57	75.57	75.57	71.33	71.33	71.33
+ OSPC	**78.24**	**78.24**	**78.24**	**73.12**	**73.12**	**73.12**

Table 6: Applying OSPC: NED performance on TAC2009, including each of the three NED systems, and the results after applying OSPC for syntactic collocations and propositions. Bold is used for best results for each system.

directly comparable, and therefore report the results on the same occurrences, that is, the occurrences where OSPD at the document level can be applied.

The results show a small but consistent improvement for OSPD at the document level in precision, recall and F1 for the three NED systems, around 1 or 2 absolute points. The improvements when applying OSPD at the collection level are also consistent, but remarkably larger, between 5 and 9 absolute points. All improvements are statistically significant (p-value below 0.01).

Table 5 also reports the results after applying OSPD to TAC2009 instances which occurred more than once in a document. Results for OSPD at document level and collection level follow the same methodology as for AIDA. The improvement at the collection level is not so consistent, with a loss in performance for Spotlight, a small improvement for UKB, and a larger improvement for (Han and Sun, 2011). All differences across the table are statistically significant (p-value below 0.01).

While the OSPD at the document level is strong in both corpora, Section 3.1 showed that the OSPD at the collection level is only strong in AIDA, with a much lower estimate in TAC2009. This fact would explain why the improvement with OSPD at the collection level is not consistent. Following the rationale in Section 3.1, we think that had the organizers of the task chosen strings and documents at random, the improvement in TAC 2009 at the collection level would be also as high as in AIDA. The high improvement in AIDA at the collection level compared to the more modest improvement at the document level, despite having a lower OSPD estimate (cf. Section 3.1), could be caused by the fact that there are more occurrences and evidence in favor of the majority entity.

5.2 One entity per collocation

Figure 4 shows the application of OSPC to the output of a NED system to two sample collocations in our dataset. In this case, the application of OSPC would increase precision to 100%. The actual result on the datasets produced in Section 4 for syntactic collocations and propositions is reported on table 6.

Regarding syntactic collocations, table 6 shows that the improvement is small but consistent for the three systems on precision, recall and F1, ranging from 0.5 to 2.5 absolute points in F1 score. The results for propositions also show the same trend, with consistent improvements across the table. All differences

in the two tables are statistically significant (p-value < 0.01), except for UKB.

6 Conclusions and future work

Our study shows that OSPD holds for 96%-98% (in the AIDA and TAC2009 datasets, respectively) of the mentions that occur multiple times in documents. We also measured OSPD at the collection level (86% and 75%, respectively). OSPC holds for 91% of the mentions that occur multiple times in the syntactic collocations that we studied, and 98% of the mentions that occur multiple times in more specific collocations. We reused the publicly available AIDA dataset for estimating OSPD. In addition, we created a dataset to study OSPC based on the TAC KBP Entity Linking 2009 task dataset, which is publicly available[8].

We carefully chose to estimate both OPSD and OSPC on TAC2009, in order to make the numbers between OSPD and OSPC comparable. The OSPD numbers for AIDA are very similar to those obtained on TAC2009, providing complementary evidence. Although the high estimate of OSPD for entities was somehow expected, the high estimate of OSPC for the syntactic collocations, especially the propositions, was somehow unexpected, given the high ambiguity rate of the discussed strings, and the fact that the ambiguity included similar entities, like for instance "ABC" which can refer, among other 190 entities, to the American Broadcasting Company or the Australian Broadcasting Corporation.

Our results also show that a simple application of the OSPD and OSPC hypotheses to the output of three different NED systems improves the results in all cases. Remarkably, the highest performance gain, 8 absolute points, was for OSPD at the collection level in the AIDA corpus.

The results presented here could be largely dependent on the domain and genre of the documents, as well as the definition of collocation. Our work is a strong basis for claiming that OSPD and OSPC hold for entities, but the evidence could be further extended exploring alternative operationalization of collocations and a larger breadth of genres and domains.

For the future we would like to check whether these hypotheses can be further used to improve current NED systems. The OSPD hypothesis can be used to jointly disambiguate all occurrences of a mention in a document. The OSPC hypothesis could be used to acquire important disambiguation features, or to perform large-scale joint entity linking. The OSPD for whole collections could be useful for documents on specific domains, and for domain adaptation scenarios.

Acknowledgements

This work was partially funded by MINECO (CHIST-ERA READERS project – PCIN-2013-002- C02-01) and the European Commission (QTLEAP – FP7-ICT-2013.4.1-610516, OPENER – FP7-ICT-2011-SME-DCL-296451). Ander Barrena is supported by a PhD grant from the University of the Basque Country.

References

Eneko Agirre and Philip Edmonds. 2007. *Word Sense Disambiguation: Algorithms and Applications*. Springer Publishing Company, Incorporated, 1st edition.

Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 33–41.

Marine Carpuat. 2009. One translation per discourse. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, DEW '09, pages 19–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. 2013. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems*, I-SEMANTICS '13, pages 121–124, New York, NY, USA. ACM.

[8] http://ixa2.si.ehu.es/OEPDC

Marie-Catherine De Marneffe and Christopher D. Manning. 2008. Stanford typed dependencies manual. *URL http://nlp. stanford. edu/software/dependencies manual. pdf*.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. One sense per discourse. In *Proceedings of the workshop on Speech and Natural Language*, HLT '91, page 233237, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R. Curran. 2012. Evaluating Entity Linking with Wikipedia. *Artif. Intell.*, 194:130–150, January.

Xianpei Han and Le Sun. 2011. A Generative Entity-mention Model for Linking Entities with Knowledge Base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 945–954.

Johannes Hoffart, Mohamed A. Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stephan Thater, and Gerdhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Conference on Empirical Methods in Natural Language Processing, Edinburgh, Scotland, United Kingdom 2011*, pages 782–792.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the tac 2010 knowledge base population track. In *Third Text Analysis Conference (TAC 2010)*.

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

Robert Krovetz. 1998. More than one sense per discourse. In *NEC Princeton NJ Labs., Research Memorandum*.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4).

David Martinez and Eneko Agirre. 2000. One sense per collocation and genre/topic variations. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, page 207215, Stroudsburg, PA, USA. Association for Computational Linguistics.

Anselmo Peñas and Eduard Hovy. 2010. Filling knowledge gaps in text for machine reading. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 979–987. Association for Computational Linguistics.

Valentin I. Spitkovsky and Angel X. Chang. 2012. A Cross-lingual Dictionary for English Wikipedia Concepts. *Eighth International Conference on Language Resources and Evaluation (LREC 2012)*.

David Yarowsky. 1993. One sense per collocation. In *Proceedings of the workshop on Human Language Technology*, HLT '93, page 266271, Stroudsburg, PA, USA. Association for Computational Linguistics.

Comparable Study of Event Extraction in Newswire and Biomedical Domains

Makoto Miwa[†,‡] **Paul Thompson**[†] **Ioannis Korkontzelos**[†] **Sophia Ananiadou**[†]
[†]National Centre for Text Mining and School of Computer Science,
University of Manchester, United Kingdom
[‡]Graduate School of Engineering, Toyota Technological Institute, Japan
{makoto.miwa, paul.thompson, ioannis.korkontzelos, sophia.ananiadou}@manchester.ac.uk

Abstract

Event extraction is a popular research topic in natural language processing. Several event extraction tasks have been defined for both the newswire and biomedical domains. In general, different systems have been developed for the two domains, despite the fact that the tasks in both domains share a number of characteristics. In this paper, we analyse the commonalities and differences between the tasks in the two domains. Based on this analysis, we demonstrate how an event extraction method originally designed for the biomedical domain can be adapted for application to the newswire domain. The performance is state-of-the-art for both domains, with F-scores of 52.7% for the biomedical domain and 52.1% for the newswire domain in terms of their primary evaluation metrics.

1 Introduction

Research into event extraction was initially focussed on the general language domain, largely driven by the Message Understanding Conferences (MUC) series (e.g., Chinchor (1998)) and the Automated Content Extraction (ACE) evaluations[1]. More recently, the focus of research has been widened to the biomedical domain, motivated by the ongoing series of biomedical natural language processing (BioNLP) shared tasks (STs) (e.g., Kim et al. (2013)).

Although the textual characteristics and the types of relevant events to be extracted can vary considerably between domains, the same general features of events normally hold across domains. An event usually consists of a trigger and arguments (see Figures 1 and 2.) A trigger is typically a verb or a nominalised verb that denotes the presence of the event in the text, while the arguments are usually entities. In general, arguments are assigned semantic roles that characterise their contribution towards the event description.

Until now, however, there has been little, if any, effort by researchers working on event extraction in different domains to share ideas and techniques, unlike syntactic tasks (e.g., (Miyao and Tsujii, 2008)) and other information extraction tasks, such as named entity recognition (e.g., (Giuliano et al., 2006)) and relation extraction (e.g., (Qian and Zhou, 2012)). This means that the potential to exploit cross-domain features of events to develop more adaptable event extraction systems is an under-studied area. Consequently, although there is a large number of published studies on event extraction, proposing many different methods, no work has previously been reported that aims to adapt an event extraction method developed for one domain to a new domain.

In response to the above, we have investigated the feasibility of adapting an event extraction method developed for the biomedical domain to the newswire domain. To facilitate this, we firstly carry out a detailed static analysis of the differences that hold between event extraction tasks in the newswire and biomedical domains. Specifically, we consider the ACE 2005 event extraction task (Walker et al., 2006) for the newswire domain and the Genia Event Extraction task (GENIA) in BioNLP ST 2013 (Kim et al., 2013) for the biomedical domain. Based on the results of this analysis, we adapt the biomedical event

[1] itl.nist.gov/iad/mig/tests/ace

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2270–2279, Dublin, Ireland, August 23-29 2014.

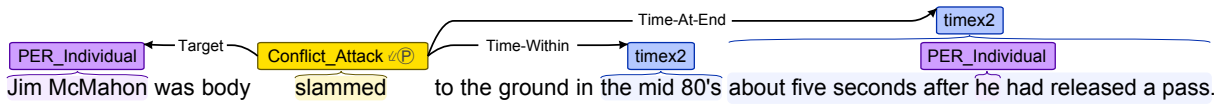


Figure 1: ACE 2005 event example (ID: MARKBACKER_20041220.0919)

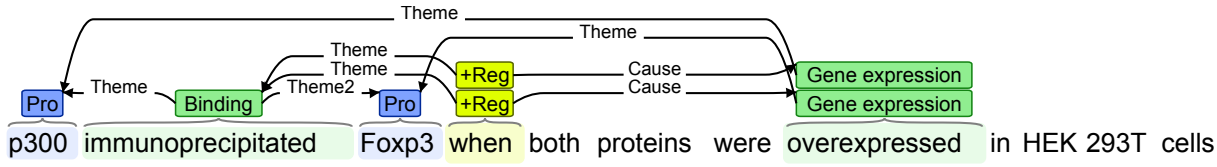


Figure 2: GENIA event example (ID: PMC-1447668-08-Results)

extraction method to the task of extracting events in the newswire domain, according to the specification of the ACE 2005 event extraction task. The original method consists of a classification pipeline that has previously been applied to extract events according to task descriptions that are similar to GENIA. In order to address the differences between this task and the ACE task, we have made a number of changes to the original method, including modifications to the classification labels assigned, the pipeline itself and the features used. We retrained the model of the adapted system on the ACE task, compared the performance, and empirically analysed the differences between the two tasks in terms of entity-related information. We demonstrate that the resulting system achieves state-of-the-art performance for tasks in both domains.

2 Related Work

In this section, we introduce the two domain specific event extraction tasks on which we will focus, i.e., the ACE 2005 event extraction task, which concerns events in the newswire domain, and the GENIA event task from the BioNLP ST 2013, which deals with biomedical event extraction. We also examine state-of-the-art systems that have been developed to address each task.

2.1 Newswire Event Extraction

The extraction of events from news-related texts has been widely researched, largely due to motivation from the various MUC and ACE shared tasks. Whilst MUC focussed on filling a single event template on a single topic by gathering information from different parts of a document, ACE defined a more comprehensive task, involving the recognition of multiple fine-grained and diverse types of entities and associated intra-sentential events within each document.

A common approach to tackling the MUC template filling task has involved the employment of pattern-based methods, e.g., Riloff (1996). In contrast, supervised learning approaches have constituted a more popular means of approaching the ACE tasks[2]. In this paper, we choose to focus on adapting our biomedical-focussed event extraction method to the ACE 2005 task. Our choice is based on the task definition for ACE 2005 having more in common with the BioNLP 2013 GENIA ST definition than the MUC event template task definition.

In terms of the characteristics of state-of-the-art event extraction systems designed according to the ACE 2005 model, pipeline-based approaches have been popular (Grishman et al., 2005; Ahn, 2006). Grishman et al. (2005) proposed a method that sequentially identifies textual spans of arguments, role types, and event triggers. This pipeline approach has been further extended in several subsequent studies. For example, Liao et al. (2010) investigated document-level cross-event consistency using co-occurrence of events and event arguments, while Hong et al. (2011) exploited information gathered from the web to ensure cross-entity consistency.

[2]Note that there are also approaches using few or no training data (e.g., (Ji and Grishman, 2008; Lu and Roth, 2012)) for the ACE 2005 task, but they are not so many and we will focus on the supervised learning approaches in this paper.

Li et al. (2013) recently proposed a joint detection method to detect both triggers and arguments (together with their role types) using a structured perceptron model. The system outperformed the best results reported for the ACE 2005 task in the literature, without the use of any external resources.

2.2 Biomedical Event Extraction

The task of event extraction has received a large amount of attention from BioNLP researchers in recent years. Interest in this task was largely initiated by the BioNLP 2009 ST, and has been sustained through the organisation of further STs in 2011 and 2013. The STs consist of a number of different sub-tasks, the majority of which concern the extraction of events from biomedical papers from the PubMed database. Events generally concern interactions between biomedical entities, such as proteins, cells and chemicals.

Similarly to newswire event extraction systems, pipeline-based methods have constituted a popular approach to extracting events in the biomedical domain (Björne and Salakoski, 2013; Miwa et al., 2012). The pipeline developed by Miwa et al. (2012) consists of a number of modules, which sequentially detect event triggers, event arguments, event structures and hedges (i.e., speculations and negations). The system has been applied to several event extraction tasks, and has achieved the best performance on most of these, in comparison to other systems. It should be noted that the ordering of the components in biomedical event extraction pipelines often differs from pipelines designed for news event extraction, e.g., Grishman et al. (2005), which was described above.

As in newswire event detection, some joint (non pipeline-based) approaches have also been proposed for biomedical event extraction. For example, McClosky et al. (2012) used a stacking model to combine the results of applying two different methods to event extraction. The first method is a joint method, similar to Li et al. (2013), that detects triggers, arguments and their roles. However, in contrast to the structured perceptron employed in Li et al. (2013), McClosky et al. (2012) use a dual-decomposition approach for the detection. The second method is based on dependency parsing and treats event structures as dependency trees.

3 Adaptation of Biomedical Event Extraction to Newswire Event Extraction

In this section, we firstly analyse the differences between the domain-specific ACE 2005 and GENIA event extraction tasks. Based on our findings, we propose an approach to adapting an existing event extraction method, originally developed for biomedical event extraction, to the ACE 2005 task, by resolving the observed differences between the two task definitions.

3.1 Differences in event extraction tasks

Both the ACE 2005 and GENIA tasks concern the task of event extraction, i.e., the identification of relationships between entities. For both tasks, the requirement is to extract events from text that conform to the general event description introduced earlier, i.e., a trigger and its arguments, each of which is assigned a semantic role. Despite this high-level similarity between the tasks, their finer-grained details diverge in a number of ways. Apart from the different textual domain, the tasks adopt varying annotation schemes. The exact kinds of annotations provided at training time are also different, as are the evaluation settings.

Several variants of the official task setting for the ACE 2005 corpus have been defined. This is partly due to the demanding nature of the official task definition, which requires the detection of events from scratch, including the recognition of named entities participating in events, together with the resolution of coreferences. Alternative task settings (such as Ji and Grishman (2008); Liao and Grishman (2010))) generally simplify the official task definition, e.g., by omitting the requirement to perform coreference resolution. A further issue is that the test data sets for the official task setting have not been made publicly available. As a result of the multiple existing variations of the ACE 2005 task definition that have been employed by different research efforts, direct comparison of our results with those obtained by other state-of-the art systems is problematic. The solution we have chosen is to adopt the same ACE 2005 event extraction task specification that has been adopted in recent research, by Hong et al. (2011) and Li et al. (2013). For GENIA, we follow the specification of the original GENIA event extraction task.

	ACE 2005	GENIA
# of entity types	13 (type) / 53 (subtype)	2
Argument	Entity/Nominal/Value/Time	Entity
# of event types	8 (type) / 33 (subtype)	13
# of argument role types	35	7
Max # of arguments for an event	11	4
Nested events	None	Possible
Overlaps of events	None	Possible
Correspondences of arguments	None	Possible
Entity	Available (Given)	Available (Partially given)
Entity attributes	Available (Given)	Not available
Event attributes	Available (Not given)	Available (Not given)
Entity coreference	Available (Given)	Available (Not given)
Event coreference	Available (Not given)	Not available
Evaluation	Trigger/Role	Event

Table 1: Comparison of event definitions and event extraction tasks. *"Available annotations"* are annotations available in the corresponding corpus, while *"Given annotations"* are annotations provided during (training and) prediction. *"Given annotations"* do not need to be predicted during event extraction.

Event annotation examples for ACE 2005 and GENIA are shown in Figures 1 and 2, respectively. Table 1 summarises the following comparison between the two event extraction tasks.

Semantic types There are more event, role and entity types and a greater potential number of arguments in ACE 2005 events than in GENIA events. There is also a hierarchy of event types and entity types in ACE 2005. For example, the *Life* event type has *Be-Born*, *Marry*, *Divorce*, *Injure*, *Die* event subtypes. Some GENIA event types can also be arranged to have a hierarchy but they are limited. Events in ACE 2005 can take non-entity arguments, e.g., *Time*.

Nested events/Overlapping events Event structures are flat in ACE 2005, but they can be nested in GENIA, i.e., an event can take other events as its arguments. Events in GENIA can also be overlapping, in the sense that a particular word or phrase can be a trigger for multiple events. Figure 2 illustrates both nesting and overlapping in GENIA events. These properties of GENIA events are not addressed by methods developed for event extraction according to the ACE 2005 specification, making direct application of these methods to the GENIA task impossible.

Links amongst arguments A specific feature of the GENIA event extraction task, which is completely absent from the ACE 2005 task, is that links amongst arguments sometimes have to be identified. For example, the *Binding* event type in the GENIA task can take the following argument role types: *Theme*, *Theme2*, *Site* and *Site2*. The number *2* is attached to differentiate specific linkages between arguments: *Site* is the location of *Theme*, while *Site2* is the location of *Theme2*.

Entities, events and their attributes Entities in ACE 2005 have rich attributes associated with them. For example, the *Time* entity type has an attribute to store a normalised temporal format (e.g., *2003-03-04* for entities "20030304", "March 4" and "Tuesday") while the *GPE (Geo-Political Entity)* type has attributes such as subtypes (e.g., *Nation*), mention type (proper name, common noun or pronoun), roles (location of a group or person) and style (*literal* or *metonymic*). In contrast, GENIA entities have no attributes[3]. In ACE 2005, all entities are provided (gold) in the training and test data and they do not need to be predicted. In GENIA, some named entities (i.e., *Protein*s) are also provided, but other types of named or non-named entities that can constitute event arguments, such as locations and sites of proteins, are not provided in the test data and thus need to be predicted as part of the extraction process. Events in both corpora also have associated attributes: modality,

[3]Types are not counted as attributes in this paper.

polarity, genericity and tense in ACE 2005 and negation and speculation in GENIA. The GENIA task definition requires event attributes to be predicted, but the ACE 2005 task definition does not.

Coreference Both entity and event coreference are annotated in ACE 2005, but only entity coreference is annotated in GENIA. Events in ACE 2005 can take non-entity mentions, such as pronouns, as their arguments. However, events in GENIA can take only entity mentions as arguments. Thus, instead of non-entity mentions, coreferent entity mentions that are the closest to triggers are annotated as arguments in GENIA. For example, in Figure 2, "*p300*" and "*Foxp3*" are annotated as *Theme*s of *Gene_expression* events instead of "*both proteins*".

Evaluation In ACE 2005, the accuracy of extracted events is evaluated at the level of individual arguments and their roles. Completeness of events is not taken into consideration (Li et al., 2013), presumably because each event can take many arguments. Evaluation is performed by taking into account the 33 event subtypes, rather than the 8 coarser-grained event types. In contrast, evaluation of events according to the GENIA specification considers only the correctness of *complete* events, after nested events have been broken down.

In summary, the ACE 2005 task is in some respects more complex than the GENIA task, because it concerns a greater number event types, whose arguments may constitute a greater range of entity types, and whose semantic roles are drawn from a larger set, some of which are specific to particular event types and entities. In other respects, the task is more straightforward than the GENIA task, because of the simpler nature of the event structures in ACE 2005, i.e., there are no nested or overlapping event structures.

3.2 Adaptation of event extraction method

Since event structures are simpler in ACE 2005 than GENIA, we choose to adapt a biomedical event extraction method to the ACE 2005 task rather than the other way around. The inverse adaptation, starting from a newswire event extraction method, is considered more complex, since we would need to extend the method to capture the more complex event structures required in the GENIA task. It would additionally be inappropriate to employ domain adaptation methods (Daumé III and Marcu, 2006; Pan and Yang, 2010) to allow GENIA-trained models to be applied to the ACE 2005 tasks. This is because such methods require that there is at least a certain degree of overlap between the target information types, which is not the case in this scenario.

We employ the biomedical event extraction pipeline method described in Miwa et al. (2012) as our starting point. Our motivation is that, due to their modular nature, pipeline approaches are often easier to adapt to other task settings than joint approaches, e.g., (McClosky et al., 2012; Li et al., 2013). In addition, the method has previously been shown to achieve state-of-the-art performance in several biomedical event extraction tasks (Miwa et al., 2012).

The pipeline consists of four detectors, i.e., trigger/entity, event role, event structure, and hedge detectors. The trigger/entity detector finds triggers and entities in text. The event role detector determines which triggers/entities constitute arguments of events, links them to the appropriate event trigger and assigns semantic roles to the arguments. The event structure detector merges trigger-argument pairs into all possible complete event structures, and determines which of these structures constitute actual events. The same detector determines links between arguments, such as *Theme2* and *Site2*. The hedge detector finds negation and speculation information associated with events. Each detector solves multi-label multi-class classification problems using lexical and syntactic features obtained from multiple parsers. These features include character n-grams, word n-grams, and shortest paths between triggers and participants within parse structures. More detailed information can be found in Miwa et al. (2012).

We have updated the original method by simplifying the format of the classification labels used by both the event role detector and event structure detector modules. We refer to this method as *BioEE*, which we have applied to the GENIA task. We use only the role types (e.g., *Theme*) as classification labels for instances in the event role detector, instead of the more complex labels used in the original version of the module, which combined event types, roles and semantic entity types of arguments (e.g.,

Binding:Theme-Protein). Similarly, in the event structure detector, we use only two labels ("EVENT" or "NOT-EVENT"), instead of the previously used composite labels, which consisted of the event type, together with the roles and semantic entity types of all arguments of the event (e.g., *Regulation:Cause-Protein:Theme-Protein*.) We employed the simplified labels, since they increase the number of training instances for each label. The use of such labels, compared to the more complex ones, could reduce the potential of carrying out detailed modelling of specific aspects of the task. However, this was found not to be an issue, since the use of the simplified labels improved the performance of the pipeline in detecting events within the GENIA development data set (about 1% improvement in F-score). The simplification of the set of classification labels was also vital to ensure the tractability of the classification problems within the context of the ACE 2005 task. For example, using the same conventions to formulate classification labels as in the original system would result in 345 possible labels (compared to 91 in GENIA) to be predicted by the event role detector (and an even greater number of labels for the event structure detector), based on event-role-semantic type combinations found in the ACE training/development sets.

In order to adapt the system to extract events according to the ACE 2005 specification, we modified BioEE in several ways, making changes to both the pipeline itself and the features employed by the different modules. We refer to this method as *Adapted BioEE*, and we applied this method to the ACE 2005 task. These changes were made in an attempt to address the two major differences between the GENIA and ACE 2005 tasks, i.e., the simpler event structures and the availability of entity attribute and coreference information in ACE.

The pipeline-based modifications consisted of removing certain modules from the original pipeline, such that only two modules remained, i.e., the trigger/entity and event role detectors. The other two modules of the original pipeline, i.e., the event structure and hedge detectors, were designed to deal with problems that do not exist in the ACE 2005 extraction task, and thus their usage would be redundant. Instead of using the event structure detector to piece the different elements of an event, we simply aggregate all the arguments of the same trigger into a single event structure, after the event role detector has been applied.

As mentioned above, the ACE 2005 task definition includes rich information about entities, including attributes and coreference information. Existing systems developed to address this task have exploited this information to generate rich feature sets for classification (Liao and Grishman, 2010; Li et al., 2013). Based on the demonstrated utility of this information within the context of event extraction, we also choose to use it, by adding binary feature that indicate the presence of base forms, entity subtypes, and attributes of the entities and their coreferent entities to features in both detectors above. We choose to use base forms, since surface forms of entities are not used by most biomedical event extraction systems, including BioEE. We also add the features for Brown clusters (Brown et al., 1992) following Li et al. (2013). Further details can be found in Li et al. (2013).

4 Evaluation

4.1 Evaluation settings

To assess the performance of Adapted BioEE on the ACE 2005 task, we followed the evaluation process and settings used in previously reported studies (Hong et al., 2011; Li et al., 2013). ACE 2005 consists of 599 documents. In order to facilitate direct comparison with other systems trained on the same data, we conducted a blind test on the same 40 newswire documents that were used for evaluation in (Ji and Grishman, 2008; Li et al., 2013), and used the remaining documents as training/development sets. We use precision (P), recall (R) and F-score (F) to report the performance of the adapted system in classifying triggers and argument roles. We use the latter F-score as our primary metric for comparing our system with other systems, since this score better reflects the performance of the extraction of event structures.

GENIA consists of 34 full paper articles (Kim et al., 2013). To evaluate the performance of BioEE on the GENIA task, we followed the task setting in BioNLP ST 2013 and used the official evaluation systems provided by the organisers. We also used the same partitioning of data that was employed in the official BioNLP ST 2013 evaluation, with 20 articles being used as the training/development set, and the remaining 14 articles being held back as the test set. For brevity, we show the only the primary P,

	Arg. Role Decomposition			Event Detection		
	P	R	F	P	R	F (%)
BioEE	71.76	47.44	57.12	64.36	44.62	52.71
BioEE (+Entity)	69.47	46.94	56.02	61.81	44.11	51.48
EVEX	64.30	48.51	55.30	58.03	45.44	50.97
TEES-2.1	62.69	49.40	55.26	56.32	46.17	50.74

Table 2: Overall performance of BioEE on the GENIA data set

	Trigger Classification			Arg. Role Classification			Event Detection		
	P	R	F	P	R	F	P	R	F (%)
Adapted BioEE	59.9	72.6	65.7	54.2	50.2	52.1	20.7	21.7	21.2
Adapted BioEE (-Entity)	57.9	71.5	64.0	51.0	48.1	49.5	19.7	19.3	19.5
Li et al. (2013)	73.7	62.3	67.5	64.7	44.4	52.7	-	-	-
Hong et al. (2011)	72.9	64.3	68.3	51.6	45.5	48.4	-	-	-

Table 3: Overall performance of Adapted BioEE on the ACE 2005 data set

R and F scores in the shared task, i.e., the EVENT TOTAL results obtained using the approximate span & recursive evaluation method, as recommended by the organisers. The method individually evaluates each complete *core* event, i.e., event triggers with their *Theme* and/or *Cause* role arguments, with relaxed span matching, after nested events have been broken down as explained in Section 3.1. Note that the scores do not count the non-named entities, hedges, and links between arguments, since only core events are considered in the official evaluation.

We applied both a deep parser, Enju (Miyao and Tsujii, 2008) and a dependency parser, ksdep (Sagae and Tsujii, 2007) to generate features for the ACE 2005 task, and their bio-adapted versions for the GENIA task. We also employed the GENIA sentence splitter (Sætre et al., 2007) for sentence splitting, and the snowball (Porter2) stemmer[4] for stemming. We did not make use of any other external resources, such as dictionaries, since this would hinder direct comparison of the two versions of the system.

4.2 Evaluation on GENIA

The "Event Detection" column in Table 2 shows evaluation results of BioEE on GENIA. The effects on performance by including entity-related features, i.e., entity base forms and Brown clustering, as introduced in Section 3.2, are shown as "BioEE (+Entity)". The inclusion of these features slightly degrades the performance.

For completeness, we also show in Table 2 the best and second best performing systems that took part in the official BioNLP 2013 ST evaluation: EVEX (Hakala et al., 2013) and TEES-2.1 (Björne and Salakoski, 2013). TEES-2.1 consists of a modular pipeline similar to BioEE, but it uses a different set of features. EVEX enhances the output of TEES-2.1, by using information obtained from the results of large-scale event extraction. The comparison shows that BioEE achieves state-of-the-art event extraction performance on the GENIA task.

4.3 Evaluation on ACE 2005

The "Trigger Classification" and "Arg. Role Classification" columns of Table 3 summarise the evaluation results of the Adapted BioEE system (as described in Section 3.2) on the ACE 2005 task.

We analysed the effects of incorporating features based on entity-related information into the extraction process, by repeating the experiments with such features omitted (-Entity). As can be observed in Table 3, the removal of entity-related features led to 3% performance decrease in F-score.

For completeness, Table 3 also illustrates the results of state-of-the-art systems that were specifically developed for ACE 2005: the system based on a joint approach (Li et al., 2013) and the pipeline-based system enhanced with web-gathered information (Hong et al., 2011). The difference between the

[4] snowball.tartarus.org

Adapted BioEE and the best system is small and insignificant and the Adapted BioEE achieved performance that is comparable to or better than these other systems, in terms of the F-scores in argument role classification.

5 Discussion

To further investigate the differences in performance of the BioEE and Adapted BioEE systems on the two tasks, we evaluate the scores achieved for each task using the evaluation criteria originally designed for the other task. Specifically, we apply the ACE 2005 argument role classification criteria to the output of GENIA task, and we apply the complete event-based evaluation, originally used to evaluate the GENIA task, to the events extracted for the ACE 2005 task. The "Arg. Role Decomposition" column of Table 2 depicts the former evaluation, while the "Event Detection" column of Table 3 shows the latter.

Table 2 also shows the performance of the other biomedical event extraction systems introduced above in carrying out argument role classification, since such information was provided as "Decomposition" within the results of the original task evaluation[5]. Although the results shown for "Arg. Role Decomposition" in Table 2 are not directly comparable to those shown for "Arg. Role Classification" in Table 3 (given the different characteristics of GENIA and ACE 2005 tasks), the scores are broadly comparable. This demonstrates that the task of argument role classifications is equally challenging for both tasks.

The "Event Detection" column of Table 3 illustrates event-based evaluation scores on ACE 2005. The event structure detector was added to the pipeline to facilitate comparison of the results of the two different tasks in a similar setting, and performance was evaluated according to the GENIA evaluation criteria. Evaluation scores on ACE 2005 are unexpectedly low compared to those in Table 2. Considering that the performance of argument role classification is similar in both tasks, this low performance is likely to be due to the large number of potential event arguments in ACE 2005. This means that, in comparison to GENIA events, which have a small number of possible argument types, there is a greater chance that some arguments of more complex ACE 2005 events will fail to be detected. According to the GENIA evaluation criteria, even if the majority of arguments has been correctly identified, the complete event structure will still be evaluated as incorrect. This helps to explain why such evaluation criteria may have been deemed inappropriate in the original ACE 2005 evaluations.

Subsequently, we analysed the effects of utilising entity-related features. We show the results obtained by adding entity information (+Entity) in Table 2 and the results obtained by removing entity information (-Entity) in Table 3. The positive or negative effect on performance of adding or removing these features is consistent across all subtask evaluations shown in the two tables, although the exact level of performance improvement or degradation depends on the subtask under evaluation. Overall, the inclusion of the features degraded the performance of BioEE on the GENIA task, but improved the performance of Adapted BioEE on the ACE 2005 task. These differences may be due to the increased richness of entity information in the ACE 2005 corpus, suggesting that enriching entities in the GENIA corpus with attribute information could be a possible way to further improve the performance of the system on this task.

6 Conclusions and Future Work

In this paper, we have described our adaptation of a biomedical event extraction method to the newswire domain. We firstly evaluated the method on a biomedical event extraction task (GENIA), and showed that its performance was superior to other state-of-the-art systems designed for the task. We then adapted the method to a newswire event extraction task (ACE 2005), by addressing the major differences between the tasks. With only a small number of adaptations, the resulting system was also able to achieve state-of-the-art performance on the newswire extraction task. These results show that there is no need to develop separate systems for event extraction tasks in different domains, as long as the types of tasks being addressed exhibit domain-independent features. However, further discussion and evaluation is needed to better understand how different potential methods for adapting such tools from one domain to another can be used and/or combined effectively.

[5] bionlp-st.dbcls.jp/GE/2013/results

As future work, we intend to further investigate the adaptation of alternative methods proposed for use in one domain to another domain. Several interesting approaches have been described, such as the utilisation of contextual information beyond the boundaries of individual sentences in the newswire domain (Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011) and joint approaches in the biomedical domain (McClosky et al., 2012), but their adaptability to other domains has not yet been investigated. We also intend to investigate the possibility of discovering and utilising shared information between the two domains (Goldwasser and Roth, 2013). Encouraging greater levels of communication between researchers working on NLP tasks in different domains will help to stimulate such new directions of research, both for event extraction and for other related information extraction tasks, such as relation extraction and coreference resolution.

Acknowledgements

This work was supported by the Arts and Humanities Research Council (AHRC) [grant number AH/L00982X/1], the Medical Research Council [grant number MR/L01078X/1], the European Community's Seventh Program (FP7/2007-2013) [grant number 318736 (OSSMETER)], and the JSPS Grant-in-Aid for Young Scientists (B) [grant number 25730129].

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia, July. ACL.

Jari Björne and Tapio Salakoski. 2013. Tees 2.1: Automated annotation scheme learning in the bionlp 2013 shared task. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 16–25, Sofia, Bulgaria, August. ACL.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Nancy A. Chinchor. 1998. Overview of MUC-7/MET-2. In *Proceedings of the 7th Message Understanding Conference (MUC-7/MET-2)*.

Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126.

Claudio Giuliano, Alberto Lavelli, and Lorenza Romano. 2006. Simple information extraction (sie): A portable and effective ie system. In *Proceedings of the Workshop on Adaptive Text Extraction and Mining (ATEM 2006)*, pages 9–16, Trento, Italy, April. Association for Computational Linguistics.

Dan Goldwasser and Dan Roth. 2013. Leveraging domain-independent information in semantic parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 462–466, Sofia, Bulgaria, August. Association for Computational Linguistics.

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. NYU's english ACE 2005 system description. In *Proceedings of ACE 2005 Evaluation Workshop*, Washington, US.

Kai Hakala, Sofie Van Landeghem, Tapio Salakoski, Yves Van de Peer, and Filip Ginter. 2013. Evex in st'13: Application of a large-scale text mining resource to event extraction and network construction. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 26–34, Sofia, Bulgaria, August. ACL.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th ACL-HLT*, pages 1127–1136, Portland, Oregon, USA, June. ACL.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio, June. ACL.

Jin-Dong Kim, Yue Wang, and Yamamoto Yasunori. 2013. The genia event extraction shared task, 2013 edition - overview. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 8–15, Sofia, Bulgaria, August. ACL.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st ACL*, pages 73–82, Sofia, Bulgaria, August. ACL.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th ACL*, pages 789–797, Uppsala, Sweden, July. ACL.

Wei Lu and Dan Roth. 2012. Automatic event extraction with structured preference modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 835–844, Jeju Island, Korea, July. Association for Computational Linguistics.

David McClosky, Sebastian Riedel, Mihai Surdeanu, Andrew McCallum, and Christopher Manning. 2012. Combining joint models for biomedical event extraction. *BMC Bioinformatics*, 13(Suppl 11):S9.

Makoto Miwa, Paul Thompson, and Sophia Ananiadou. 2012. Boosting automatic event extraction from the literature using domain adaptation and coreference resolution. *Bioinformatics*, 28(13):1759–1765.

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature forest models for probabilistic HPSG parsing. *Computational Linguistics*, 34(1):35–80, March.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Longhua Qian and Guodong Zhou. 2012. Tree kernel-based protein–protein interaction extraction from biomedical literature. *Journal of biomedical informatics*, 45(3):535–543.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049.

Rune Sætre, Kazuhiro Yoshida, Akane Yakushiji, Yusuke Miyao, Yuichiro Matsubayashi, and Tomoko Ohta. 2007. AKANE System: Protein-protein interaction pairs in BioCreAtIvE2 Challenge, PPI-IPS subtask. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pages 209–212, CNIO, Madrid, Spain, April.

Kenji Sagae and Jun'ichi Tsujii. 2007. Dependency parsing and domain adaptation with LR models and parser ensembles. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1044–1050, Prague, Czech Republic, June. ACL.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. *Linguistic Data Consortium*.

A Probabilistic Co-Bootstrapping Method for Entity Set Expansion

Bei Shi, Zhengzhong Zhang
Institute of Software,
Chinese Academy of Sciences,
Beijing, China

Le Sun, Xianpei Han
State Key Laboratory of Computer Science,
Institute of Software,
Chinese Academy of Sciences,
Beijing, China

{shibei, zhenzhong, sunle, xianpei}@nfs.iscas.ac.cn

Abstract

Entity Set Expansion (ESE) aims at automatically acquiring instances of a specific target category. Unfortunately, traditional ESE methods usually have the expansion boundary problem and the semantic drift problem. To resolve the above two problems, this paper proposes a probabilistic Co-Bootstrapping method, which can accurately determine the expansion boundary using both the positive and the discriminant negative instances, and resolve the semantic drift problem by effectively maintaining and refining the expansion boundary during bootstrapping iterations. Experimental results show that our method can achieve a competitive performance.

1 Introduction

Entity Set Expansion (ESE) aims at automatically acquiring instances of a specific target category from text corpus or Web. For example, given the capital seeds {*Rome*, *Beijing*, *Paris*}, an ESE system should extract all other capitals from Web, such as *Ottawa*, *Moscow* and *London*. ESE system has been used in many applications, e.g., dictionary construction (Cohen and Sarawagi, 2004), word sense disambiguation (Pantel and Lin, 2002), query refinement (Hu et al., 2009), and query suggestion (Cao et al., 2008).

Due to the limited supervision provided by ESE (in most cases only 3-5 seeds are given), traditional ESE systems usually employ bootstrapping methods (Cucchiarelli and Velardi, 2001; Etzioni et al., 2005; Pasca, 2007; Riloff and Jones, 1999; Wang and Cohen, 2008). That is, the entity set is iteratively expanded through a pattern generation step and an instance extraction step. Figure 1(a) demonstrates a simple bootstrapping process.

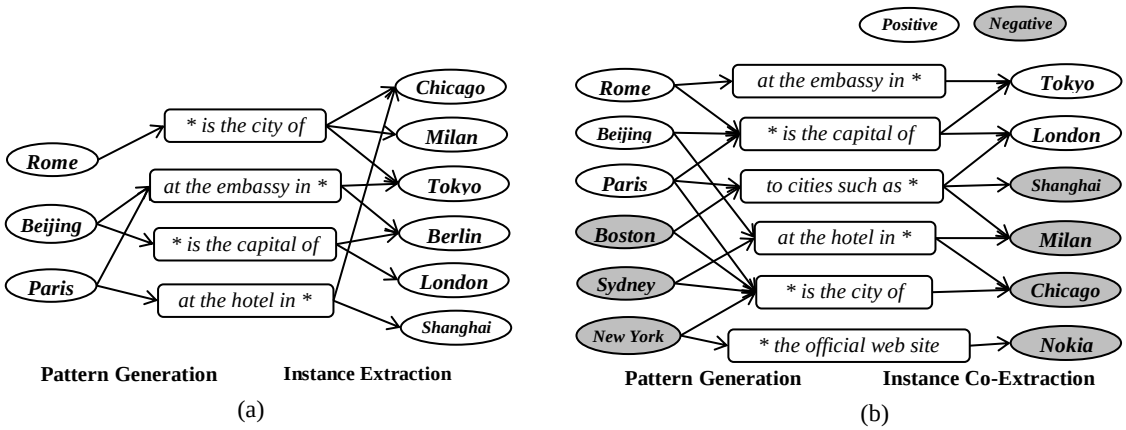


Figure 1: A demo of Bootstrapping (a) and Co-Bootstrapping (b)

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2280–2290, Dublin, Ireland, August 23-29 2014.

However, the traditional bootstrapping methods have two main drawbacks:

1) **The expansion boundary problem**. That is, using only positive seeds (i.e., some example entities from the category we want to expand), it is difficult to represent which entities we want to expand and which we don't want. For example, starting from positive seeds {*Rome, Beijing, Paris*}, we can expand entities at many different levels, e.g., all capitals, all cities, or even all locations. And all these explanations are reasonable.

2) **The semantic drift problem**. That is, the expansion category may change gradually when noisy instances/patterns are introduced during the bootstrapping iterations. For example, in Figure 1 (a), the instance *Rome* will introduce a pattern "* *is the city of*", which will introduce many noisy city instances such as *Milan* and *Chicago* for the expansion of *Capital*. And these noisy cities in turn will introduce more city patterns and instances, and finally will lead to a semantic drift from *Capital* to *City*.

In recent years, some methods (Curran et al, 2007; Pennacchiotti and Pantel, 2011) have exploited mutual exclusion constraint to resolve the semantic drift problem. These methods expand multiple categories simultaneously, and will determine the expansion boundary based on the mutually exclusive property of the pre-given categories. For instance, the exclusive categories *Fruit* and *Company* will be jointly expanded and the expansion boundary of {*Apple, Banana, Cherry*} will be limited by the expansion boundary of {*Google, Microsoft, Apple Inc.*}. These methods, however, still have the following two drawbacks:

1) These methods require that the expanded categories should be mutually exclusive. However, in many cases the mutually exclusive assumption does not hold. For example, many categories hold a hyponymy relation (e.g., the categories *City* and *Capital*, because the patterns for *Capital* are also the patterns for *City*) or a high semantic overlap (e.g., the categories *Movies* and *Novels*, because some movies are directly based on the novels of the same title.).

2) These methods require the manually determination of the mutually exclusive categories. Unfortunately, it is often very hard for even the experts to determine the categories which can define the expansion boundaries for each other. For example, in order to expand the category *Chemical Element*, it is difficult to predict its semantic drift towards *Color* caused by the ambiguous instances {*Silver, Gold*}.

In this paper, to resolve the above problems, we propose a probabilistic Co-Bootstrapping method. The first advantage of our method is that we propose a method to better define the expansion boundary using both the positive and the discriminant negative seeds, which can both be automatically populated during the bootstrapping process. For instance, in Figure 1(b), in order to expand *Capital*, the Co-Bootstrapping algorithm will populate both positive instances from the positive seeds {*Rome, Beijing, Paris*}, and negative instances from the negative seeds {*Boston, Sydney, New York*}. In this way the expansion boundary of *Capital* can be accurately determined.

The second advantage of our method is that we can maintain and refine the expansion boundary during bootstrapping iterations, so that the semantic drift problem can be effectively resolved. Specifically, we propose an effective scoring algorithm to estimate the probability that an extracted instance belongs to the target category. Based on this scoring algorithm, this paper can effectively select positive instances and discriminant negative instances. Therefore the expansion boundary can be maintained and refined through the above jointly expansion process.

We have evaluated our method on the expansion of thirteen categories of entities. The experimental results show that our method can achieve 6%~15% P@200 performance improvement over the baseline methods.

This paper is organized as follows. Section 2 briefly reviews related work. Section 3 defines the problem and proposes a probabilistic Co-Bootstrapping approach. Experiments are presented in Section 4. Finally, we conclude this paper and discuss some future work in Section 5.

2 Related Work

In recent years, ESE has received considerable attentions from both research (An et al., 2003; Cafarella et al., 2005; Pantel and Ravichandran, 2004; Pantel et al., 2009; Pasca, 2007; Wang and Cohen, 2008) and industry communities (e.g., *Google Sets*). Till now, most ESE systems employ bootstrapping methods, such as *DIPRE* (Brin, 1998), *Snowball* (Agichtein and Gravano, 2000), etc.

The main drawbacks of the traditional bootstrapping methods are the expansion boundary problem and the semantic drift problem. Currently, two strategies have been exploited to resolve the semantic drift problem. The first is the ranking based approaches (Pantel and Pennacchiotti, 2006; Talukdar et al., 2008), which select highly confident patterns and instances through a ranking algorithm, with the assumption that high-ranked instances will be more likely to be the instances of the target category. The second is the mutual exclusion constraint based methods (Curran et al., 2007; McIntosh and Curran, 2008; Pennacchiotti and Pantel, 2011; Thelen and Riloff, 2002; Yangarber et al., 2002), which expand multiple categories simultaneously and determine the expansion boundary based on the mutually exclusive property of the pre-given categories.

3 The Co-Bootstrapping Method

3.1 The Framework of Probabilistic Co-Bootstrapping

Given the initial positive seeds and negative seeds, the goal of our method is to extract instances of a specific target semantic category. For demonstration, we will describe our method through the running example shown in Figure 1(b).

Specifically, Figure 2 shows the framework of our method. The central tasks of our Co-Bootstrapping method are as follows:

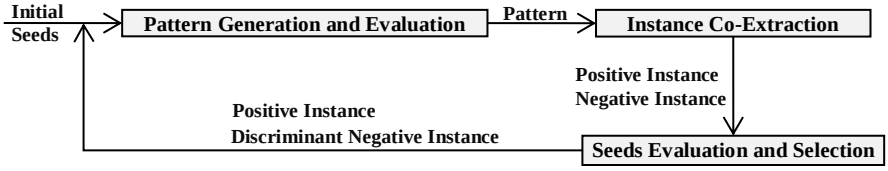


Figure 2: The framework of probabilistic Co-Bootstrapping

1)**Pattern Generation and Evaluation.** This step generates and evaluates patterns using the statistics of the positive and the negative instances. Specifically, we propose three measures of pattern quality: the Generality (*GE*), the Precision of Extracted Instances (*PE*) and the Precision of Not Extracted Instances (*PNE*).

2)**Instance Co-Extraction.** This step co-extracts the positive and the negative instances using highly confident patterns. Specifically, we propose an effective scoring algorithm to estimate the probability that an extracted instance belongs to the target category based on the statistics and the quality of the patterns which extract it.

3)**Seed Selection.** This step selects the high ranked positive instances and discriminant negative instances to refine the expansion boundary by measuring how well a new instance can be used to define the expansion boundary.

The above three steps will iterate until the number of extracted entities reaches a predefined threshold. We describe these steps as follows.

3.2 Pattern Generation and Evaluation

In this section, we describe the pattern generation and evaluation step. In this paper, each pattern is a 4-grams lexical context of an entity. We use the Google Web 1T corpus's (Brants and Franz, 2006) 5-grams for both the pattern generation and the instance co-extraction in ESE. Our method generates patterns through two steps: 1) Generate candidate patterns by matching seeds with the 5-grams. 2) Evaluate the quality of the patterns.

For the first step, we simply match each seed instance with all 5-grams, then we replace the matching instance with wildcard "*" to generate the pattern.

Count	**Positive**	**Negative**
Extracted	Extracted Positive (***ep***)	Extracted Negative (***en***)
Not Extracted	Not Extracted and Positive (***nep***)	Not Extracted and Negative (***nen***)

(a)

Extracted Positive (***ep***)	*London*
Extracted Negative (***en***)	*Shanghai, Milan*
Not Extracted Positive (***nep***)	*Tokyo*
Not Extracted Negative (***nen***)	*Chicago, Nokia*

(b)

Table 1: (a) shows the four classes of instances according to polarity and extraction. (b) shows the four classes of the instances given *"to cities such as *"*

For the second step, we propose three measures to evaluate the quality of a pattern, correspondingly the *Generality (GE)*, the *Precision of Extracted Instances (PE)*, and the *Precision of Not Extracted Instances (PNE)*. Specifically, given a pattern, we observed that all instances can be categorized into four classes, according to whether they belong to the target category and whether they can be extracted by the pattern (shown in Table 1(a)). For example, given the pattern *"to cities such as *"* in Figure 1(b), the instances under its four classes are shown in Table 1 (b).

The proposed three measures of the quality of a pattern can be computed as follows (In most cases, we cannot get the accurate number of *ep*, *en*, *nep* and *nen*. So this paper uses the corresponding known instances in the previous iteration to approximately compute *ep*, *en*, *nep* and *nen*):

1)**Generality** (*GE*). The Generality of a pattern measures how many entities can be extracted by it. A more general pattern will cover more entities than a more specific pattern. Specifically, the *GE* of a pattern is computed as:

$$GE = \frac{ep + en}{ep + en + nep + nen}$$

That is, the proportion of the instances which can be extracted by the pattern in the previous iteration.

2)**Precision of Extracted Instances** (*PE*). The *PE* measures how likely an instance extracted by a pattern will be positive. That is, a pattern with higher *PE* will be more likely to extract positive instances than a lower *PE* pattern. The *PE* is computed as:

$$PE = \frac{ep}{ep + en}$$

That is, the proportion of positive instances within all instances which can be extracted by the pattern in the previous iteration.

3)**Precision of Not Extracted Instances** (*PNE*). The *PNE* measures how likely a not extracted instance is positive. Instances not extracted by a high *PNE* pattern will be more likely to be positive. *PNE* is computed as:

$$PNE = \frac{nep/(ep + nep)}{nep/(ep + nep) + nen/(en + nen)}$$

Because the number of negative instances is usually much larger than the number of positive instances, we normalize the number of positive and negative instances in the formula.

Table 2 shows these measures of some selected patterns evaluated using the Google Web 1T corpus. We can see that the above measures can effectively evaluate the quality of patterns. For instance, *GE("* is the city of")=0.566* is larger than *GE*("at the embassy in **")=0.340*, which is consistent with our intuition that the pattern *"* is the city of"* is more general than *"at the embassy in *"*. *PE*(*"* is the capital of"*)*=0.928* is larger than *PE*(*"* is the city of"*)*=0.269*, which is consistent with our intuition that the instances extracted by *"* is the capital of"* are more likely *Capital* than by*"* is the city of"*.

	GE	***PE***	***PNE***
*at the embassy in **	*0.340*	*0.833*	*0.312*
** is the capital of*	*0.321*	*0.928*	*0.224*
*to cities such as **	*0.426*	*0.875*	*0.566*
*at the hotel in **	*0.333*	*0.192*	*0.571*
** is the city of*	*0.566*	*0.269*	*0.592*
** the official web site*	*0.218*	*0.230*	*0.607*

Table 2: The *GE*, *PE* and *PNE* of some selected patterns

3.3 Instance Co-Extraction

In this section, we describe how to co-extract positive instances and discriminant negative instances. Given the generated patterns, the central task of this step is to measure the likelihood of an instance to be positive. The higher the likelihood, the more likely the instance belongs to the target category. To resolve the task, we propose a probabilistic method which predicts the probability of an instance to be positive, i.e., the *Instance Positive Probability* and we denote it as *P+*. Generally, the ***P+*** is determined by both the statistics and the quality of patterns. We start with the observation that:

1) If an instance is extracted by a pattern with a high *PE*, the instance will have a high ***P+***.
2) If an instance is not extracted by a high *PNE* pattern, the instance will have a high ***P+***.
3) If an instance is extracted by many patterns with high *PE* and not extracted by many patterns with high *PNE*, the instance will have a high ***P+***, and vice versa.

Based on the above observations, the computation of ***P+*** is as follows:

The Situation of One Pattern

For the situation that only one pattern exists, the ***P+*** of an instance can be simply computed as:

$$P+(e) = \begin{cases} PE(p) & \text{when } p \text{ extracts } e \\ PNE(p) & \text{otherwise} \end{cases}$$

where *e* denotes an extracted instance and *p* denotes a pattern which extracts *e*. This formula means that if the instance is extracted by a pattern, the ***P+*** is determined by the *PE* of the pattern. For example, in Figure 3 (a), the instance *Tokyo* is only extracted by the pattern *"at the embassy in *"* and the ***P+*** is determined by the *PE* of *"at the embassy in *"*, i.e., ***P+***(*Tokyo*)=*PE*(*"at the embassy in *"*).

The above formula also means when the instance cannot be extracted by the only pattern, the ***P+*** will be determined by the *PNE* of the pattern. For example, in Figure 3 (b), the instance *Tokyo* is not extracted by the only pattern *"at the hotel in *"* and the ***P+*** is only determined by the *PNE* of *"at the hotel in *"*, that is, ***P+****(Tokyo)=PNE*(*"at the hotel in *"*).

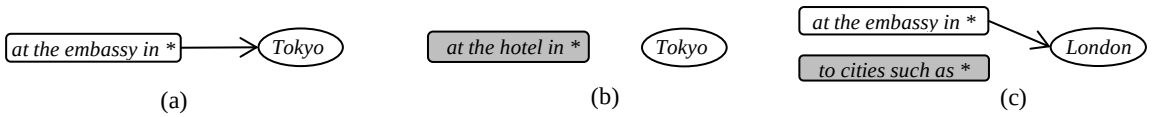


Figure 3: (a) *Tokyo* is extracted by *"at the embassy in *"*. (b) *Tokyo* is not extracted by *"at the hotel in *"*. (c) *London* is extracted by *"at the embassy in *"* and not extracted by *"to cities such as *"*.

The Situation of Multiple Patterns

In this section, we describe how to compute ***P+*** in the situation of multiple patterns. Specifically, we assume that an instance is extracted by different patterns independently. Therefore, given all the pattern-instance relations (i.e., whether a specific pattern extracts a specific instance), the likelihood for an instance *e* being positive is computed as:

$$PosLikelihood(e) \propto \prod_{p \in R^+} P(p \to e, e \in I^+) \prod_{p \in R^-} P(p \nrightarrow e, e \in I^+)$$

where R^+ is all the patterns which extract *e*, and R^- is all the patterns which do not extract *e*. I^+ is the set of all positive instances. $P(p \to e, e \in I^+)$ is the probability of the event *"pattern p extracts instance e and e is positive"*. Using Bayes rule, this probability can be computed as:

$$P(p \to e, e \in I^+) = P(p \to e)P(e \in I^+ | p \to e)$$

where $P(p \to e)$ is the probability of the event *"p extracts an instance e"*, its value is *GE(p)*; $P(e \in I^+ | p \to e)$ is the conditional probability that *e* is positive under the condition *"p extracts e"*, and its value is *PE(p)*. Finally $P(p \to e, e \in I^+)$ is computed as:

$$P(p \to e, e \in I^+) = GE(p)PE(p)$$

$P(p \nrightarrow e, e \in I^+)$ is the probability of the event "*p does not extract e and e is positive*", which can be computed as:

$$P(p \nrightarrow e, e \in I^+) = P(p \nrightarrow e)P(e \in I^+ | p \nrightarrow e)$$

$P(p \nrightarrow e)$ is the probability of *p* not extracting an instance *e*, and its value is 1-*GE(p)*. $P(e \in I^+ | p \nrightarrow e)$ is the conditional probability that *e* is positive under the condition *"p does not extract e"*, and its value is *PNE(p)*. Then $P(p \nrightarrow e, e \in I^+)$ is finally computed as:

$$P(p \nrightarrow e, e \in I^+) = (1 - GE(p))PNE(p)$$

For example, in Figure 3 (c), the instance *London* is extracted by the pattern *"at the embassy in *"* and not extracted by the pattern *"to cities such as *"*. In this situation, *PosLikelihood*(*London*)= [*GE*(*"at the embassy in *"*) × *PE*(*"at the embassy in"*)] × [(1-*GE*(*"to cities such as *"*)) × *PNE*(*"to cities such as *"*)].

Using the same intuition and the same method, the likelihood of an instance being negative is computed as:

$$NegLikelihood(e) \propto \prod_{p \in R^+} P(p \rightarrow e, e \notin I^+) \prod_{p \in R^-} P(p \nrightarrow e, e \notin I^+)$$

where $P(p \rightarrow e, e \notin I^+)$ is the probability of the event *"p extracts e and e is negative"*, which is computed as:

$$P(p \rightarrow e, e \notin I^+) = GE(p)(1 - PE(p))$$

$P(p \nrightarrow e, e \notin I^+)$ is the probability of the event *"p does not extract e and e is negative"*, which is computed as:

$$P(p \nrightarrow e, e \notin I^+) = (1 - GE(p))(1 - PNE(p))$$

For instance, in Figure 3 (c), *NegLikelihood*(*London*) = [*GE*("*at the embassy in **") × (1-*PE*("*at the embassy in*"))] × [(1-*GE*("*to cities such as **")) × (1-*PNE*("*to cities such as **"))].

Finally, the *Instance Positive Probability*, ***P+***, is computed as:

$$P+(e) = \frac{PosLikelihood(e)}{PosLikelihood(e) + NegLikelihood(e)}$$

3.4 Seed Selection

In this section, we describe how to select positive and discriminant negative instances at each iteration.

To determine whether an instance is positive, we use a threshold of ***P+*** to determine the polarity of instances, which can be empirically estimated from data. The instances which have much higher ***P+*** than the threshold will be added to the set of positive instances. For example, *London* and *Tokyo* in Figure 1 (b) are selected as positive instances.

To select discriminant negative instances, we observed that not all negative instances are the same useful for the expansion boundary determination. Intuitively, the discriminant negative instances are those negative instances which are highly overlapped with the positive instances. For instance, due to the lower overlap between categories *Fruit* and *Capital*, *Apple* is not a discriminant negative instance since it provides little information for the expansion boundary determination. Therefore, the instances near the threshold are used as the discriminant negative instances in the next iteration. (Notice that, the computation of *GE*, *PE* and *PNE* still uses all positive and negative instances, rather than only discriminant negative instances). For example, in Figure 1(b), *Shanghai*, *Milan* and *Chicago* are selected as discriminate negative instances, and *Nokia* will be neglected. Finally the boundary between *Capital* and *City* can be determined by the positive instances and the discriminant negative instances.

4 Experiments

4.1 Experimental Settings

Category	Description	Category	Description
CAP	Place: capital name	**FAC**	Facilities: names of man-made structures
ELE	chemical element	**ORG**	Organization: e.g. companies, governmental
FEM	Person: female first name	**GPE**	Place: Geo-political entities
MALE	Person: male first name	**LOC**	Locations other than GPEs
LAST	Person: last name	**DAT**	Reference to a date or period
TTL	Honorific title	**LANG**	Any named language
NORP	Nationality, Religion, Political(adjectival)		

Table 3: Target categories

Corpus: In our experiments, we used the Google Web 1T corpus (Brants and Franz, 2006) as our expansion corpus. Specifically, we use the open source package *LIT-Indexer* (Ceylan and Mihalcea, 2011) to support efficient wildcard querying for pattern generation and instance extraction.

Target Expansion Categories: We conduct our experiments on thirteen categories, which are shown in Table 3. Eleven of them are from Curran et al. (2007). Besides the eleven categories, to evaluate how well ESE systems can resolve the semantic drift problem, we use two additional categories (*Capital* and *Chemical Element*) which are high likely to drift into other categories.

Evaluation Criteria: Following Curran et al (2007), we use *precision at top n (P@N)* as the performance metrics, i.e., the percentage of correct entities in the top n ranked entities for a given category. In our experiments, we use *P@10, P@20, P@50, P@100* and *P@200*. Since the output is a ranked list of extracted entities, we also choose the average precision (AP) as the evaluation metric. In our experiments, the correctness of all extracted entities is manually judged. In our experiments, we present results to 3 annotators, and an instance will be considered as positive if 2 annotators label it as positive. We also provide annotators some supporting resources for better evaluation, e.g., the entity list of target type collected from Wikipedia.

4.2 Experimental Results

In this section, we analyze the effect of negative instances, categories boundaries, and seed selection strategies. We compare our method with the following two baseline methods: i) ***Only_Pos (POS)***: This is an entity set expansion system which uses only positive seeds. ii) ***Mutual_Exclusion (ME)***: This is a mutual exclusion bootstrapping based ESE method, whose expansion boundary is determined by the exclusion of the categories.

We implement our method using two different settings: i) ***Hum_Co-Bootstrapping (Hum_CB)***: This is the proposed Co-Bootstrapping method in which the initial negative seeds are manually given. Specifically, we randomly select five positive seeds from the list of the category's instances while the initial negative seeds are manually provided. ii) ***Feedback_Co-Bootstrapping (FB_CB)***: This is our proposed probabilistic Co-Bootstrapping method with two steps of selecting initial negative seeds: 1) Expand the entity set using only the positive seeds for only first iteration. Return the top ten instances. 2) Select the negative instances in the top ten results of the first iteration as negative seeds.

4.2.1. Overall Performance

Several papers have shown that the experimental performance may vary with different seed choices (Kozareva and Hovy, 2010; McIntosh and Curran, 2009; Vvas et al., 2009). Therefore, we input the ESE system with five different positive seed settings for each category. Finally we average the performance on the five settings so that the impact of seed selection can be reduced.

	P@10	**P@20**	**P@50**	**P@100**	**P@200**	**MAP**
POS	0.84	0.74	0.55	0.41	0.34	0.42
ME	0.83(0.90)	0.79(0.87)	0.68(0.78)	0.58(0.67)	0.51(0.59)	-
Hum_CB	0.97	0.95	0.83	0.71	0.57	0.78
FB_CB	**0.97**	**0.96**	**0.90**	**0.79**	**0.66**	**0.85**

Table 4: The overall experimental results

Table 4 shows the overall experimental results. The results in parentheses are the known results of eleven categories (without *CAP* and *ELE*) shown in (Curran et al., 2007). MAP of ME is missed because there are no available results in (Curran et al., 2007). From Table 4, we can see that:

1) Our method can achieve a significant performance improvement: Compared with the baseline POS, our method Hum_CB and FB_CB can respectively achieve a 23% and 32% improvement on P@200; Compared with the baseline ME, our method Hum_CB and FB_CB can respectively improve P@200 by 6% and 15%.

2) By explicitly representing the expansion boundary, the expansion performance can be increased: Compared with the baseline POS, ME can achieve a 17% improvement on P@200, and our method Hum_CB can achieve a 23% improvement on P@200.

3) The negative seeds can better determine the expansion boundary than mutually exclusive categories. Compared with ME, Hum_CB and FB_CB can respectively achieve a 6% and 15% improvement on P@200. We believe this is because using negative instances is a more accurate and more robust way for defining and maintaining the expansion boundary than mutually exclusive categories.

4) The system's feedback is useful for selecting negative instances: Compared with Hum_CB, FB_CB method can significantly improve the P@200 by 9.0%. We believe this is because that the system's feedback is a good indicator of the semantic drift direction. In contrast, it is usually difficult for human to determine which directions the bootstrapping will drift towards.

4.2.2. Detailed Analysis: Expansion Boundary

In Table 5, we show the top 20 positive and negative *Capital* instances (FB_CB setting). From Table 5, we can make the following observations: 1) Our method can effectively generate negative instances. In Table 5, the negative instances contain cities, states, countries and general terms, all of which have a high semantic overlap with *Capital* category. 2) The positive instances and negative instances generated by our Co-Bootstrapping method can discriminately determine the expansion boundary. For instance, the negative instances *Kyoto* can distinguish *Capital* from *City*; *Australia* and *China* can distinguish *Capital* from *Country*;

Positive Instances		***London***, ***Paris***, ***Moscow***, ***Beijing***, ***Madrid***, ***Amsterdam***, ***Washington***, ***Tokyo***, ***Berlin***, ***Rome***, ***Vienna***, ***Baghdad***, ***Athens***, ***Bangkok***, ***Cairo***, ***Dublin***, ***Brussels***, ***Prague***, *San*, ***Budapest***
Negative Instances (with categories)	City	*Kyoto*, *Kong*, *Newcastle*, *Zurich*, *Lincoln*, *Albany*, *Lyon*, *LA*, *Shanghai*
	Country	*China*, *Australia*
	General	*downtown*, *April*
	State	*Hawaii*, *Oklahoma*, *Manhattan*
	Other	*Hollywood*, *DC*, ***Tehran***, ***Charlotte***

Table 5: Top 20 positive instances and negative instances (True positive instances are in bold)

4.2.3. Detailed Analysis: Semantic Drift Problem

POS	***Stockholm***, *Tampa*, *East*, *West*, *Springfield*, *Newport*, *Cincinnati*, ***Dublin***, *Chattanooga*, *Savannah*, *Omaha*, *Cambridge*, *Memphis*, *Providence*, ***Panama***, *Miami*, ***Cape***, *Victoria*, *Milan*, ***Berlin***
ME	***London***, ***Prague***, *Newport*, ***Cape***, ***Dublin***, *Savannah*, *Chattanooga*, ***Beijing***, *Memphis*, ***Athens***, ***Berlin***, *Miami*, ***Plymouth***, *Victoria*, *Omaha*, ***Tokyo***, *Portland*, *Troy*, *Anchorage*, ***Bangkok***
Hum_CB	***London***, ***Rome***, ***Berlin***, ***Paris***, ***Athens***, ***Moscow***, ***Tokyo***, ***Beijing***, ***Prague***, ***Madrid***, ***Vienna***, ***Dublin***, ***Budapest***, ***Amsterdam***, ***Bangkok***, ***Brussels***, *Sydney*, ***Cairo***, ***Washington***, *Barcelona*
FB_CB	***London***, ***Paris***, ***Moscow***, ***Beijing***, ***Washington***, ***Tokyo***, ***Berlin***, ***Rome***, ***Vienna***, ***Baghdad***, ***Athens***, ***Bangkok***, ***Cairo***, ***Brussels***, ***Prague***, *San*, ***Budapest***, ***Amsterdam***, ***Dublin***, ***Madrid***

Table 6: Top 20 instances of all methods (True positive instances are in bold)

To analyze how our method can resolve the semantic drift problem, Table 6 shows the top 20 positive *Capital* instances of different methods. From Table 6, we can make the following observations: i) Different methods can resolve the semantic drift problem to different extent: ME is better than POS, with 50% instances being positive, and our method is better than ME, with 95% instances being positive. ii) The Co-Bootstrapping method can effectively resolve the semantic drift problem: 25% of POS's top 20 instances and 50% of ME's top 20 instances are positive. In contrast, 90% of Hum_CB's top 20 instances and 95% of FB_CB's top 20 instances are positive respectively. It proves that Co-Bootstrapping method can better resolve the semantic drift problem than POS and ME.

4.3 Parameter Optimization

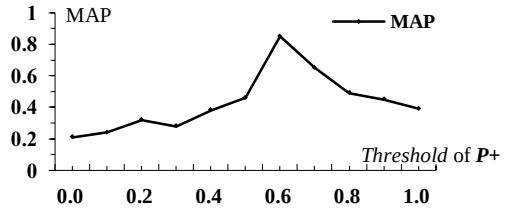


Figure 4: The MAP vs. *threshold* of ***P+***

Our method has only one parameter: *threshold* of ***P+***, which determines the instance's polarity. Intuitively, a larger *threshold* of ***P+*** will improve the precision of the positive instances but will regard some positive instances as negative instances mistakenly. As shown in Figure 4, our method can achieve the best *MAP* performance when the value of the *threshold* is 0.6.

4.4 Comparison with State-of-the-Art Systems

We also compare our method with three state-of-the-art systems: *Google Sets*[1]-- an ESE application provided by Google, *SEAL*[2] -- a state-of-the-art ESE method proposed by Wang and Cohen (2008), and *WMEB* -- a state-of-the-art mutual exclusion based system proposed in McIntosh and Curran (2008). To make a fair comparison, we directly use the results before the adjustment which miss P@10 and P@50 in their original paper (McIntosh and Curran, 2008) and compared the performance of these systems on nine categories in (McIntosh and Curran, 2008). For each system, we conduct the experiment five times to reduce the impact of seeds selection. The average P@10, P@50, P@100 and P@200 are shown in Figure 5.

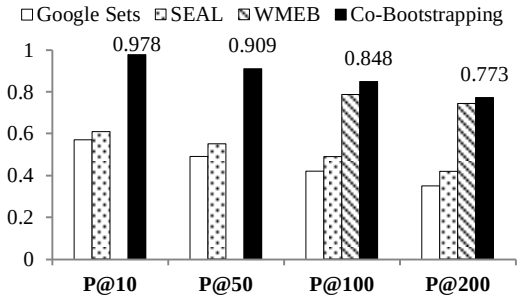


Figure 5: The results compared with three state-of-the-art systems

From the results shown in Figure 5, we can see that our probabilistic Co-Bootstrapping method can achieve state-of-the-art performance on all metrics: Compared with the well-known baseline *Google Sets*, our method can get a 42.0% improvement on P@200; Compared with the *SEAL* baseline, our method can get a 35.0% improvement on P@200; Compared with the WMEB method, our method can achieve a 6.2% improvement on P@100 and a 3.1% improvement on P@200.

5 Conclusion and Future Work

In this paper, we proposed a probabilistic Co-Bootstrapping method for entity set expansion. By introducing negative instances to define and refine the expansion boundary, our method can effectively resolve the expansion boundary problem and the semantic drift problem. Experimental results show that our method achieves significant performance improvement over the baselines, and outperforms three state-of-the-art ESE systems. Currently, our method did not take into account the long tail entity expansion, i.e., the instances which appear only a few times in the corpus, such as *Saipan, Roseau* and *Suva* for the *Capital* category. For future work, we will resolve the long tail entities in our Co-Bootstrapping method by taking the sparsity of instances/patterns into consideration.

6 Acknowledgements

We would like to thank three anonymous reviewers for invaluable comments and suggestions to improve our paper. This work is supported by the National Natural Science Foundation of China under Grants no. 61100152 and 61272324, and the National High Technology Development 863 Program of China under Grants no. 2013AA01A603.

References

Eugene Agichtein and Luis Gravano. 2000. *Snowball: Extracting Relations from Large Plain-Text Collections.* In: Proceedings of the fifth ACM conference on Digital libraries (DL-00), Pages 85-94.

Joohui An, Seungwoo Lee, and Gary Geunbae Lee. 2003. *Automatic acquisition of named entity tagged corpus from world wide web*. In: Proceedings of ACL-03, Pages 165-168, Volume 2.

Thorsten Brants and Alex Franz. 2006. *Web 1t-5gram version1*. http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2006T13

[1] https://docs.google.com/spreadsheet/
[2] http://www.boowa.com/

Sergey Brin. 1998. *Extracting patterns and relations from the World Wide Web.* In: Proceedings of the Workshop at the 6th International Conference on Extending Database Technology, Pages 172-183.

Michael J. Cafarella, Doug Downey, Stephen Soderland, and Oren Etzioni. 2005. *KnowItNow: Fast, Scalable Information Extraction from the Web*. In: Proceedings of EMNLP-05, Pages 563-570.

Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. *Context-aware query suggestion by mining click-through and session data*. In Proceedings of KDD-08, pages 875–883.

Hakan Ceylan and Rada Mihalcea. 2011. *An Efficient Indexer for Large N-Gram Corpora*. In: Proceedings of System Demonstrations of ACL-11, Pages 103-108.

William W. Cohen and Sunita Sarawagi. 2004. *Exploiting dictionaries in named entity extraction: combining semi-Markov extraction processes and data integration methods.* In: Proceedings of KDD-04, Pages 89-98.

Alessandro Cucchiarelli and Paola Velardi. 2001. *Unsupervised Named Entity Recognition Using Syntactic and Semantic Contextual Evidence*. In: Computational Linguistics, Pages 123-131, Volume 27.

James R. Curran, Tara Murphy, and Bernhard Scholz. 2007. *Minimising semantic drift with Mutual Exclusion Bootstrapping*. In: Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics, Pages 172–180.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. *Unsupervised Named-Entity Extraction from the Web: An Experimental Study*. In: Artificial Intelligence, Pages 91-134, Volume 165.

Jian Hu, Gang Wang, Fred Lochovsky, Jiantao Sun, and Zheng Chen. 2009. *Understanding user's query intent with Wikipedia*. In Proceedings of WWW-09, Pages 471–480.

Zornitsa Kozareva and Eduard Hovy. 2010. *Learning arguments and supertypes of semantic relations using recursive patterns.* In: Proceedings of ACL-10, Pages 1482–1491.

Tara McIntosh and James R. Curran. 2008. *Weighted mutual exclusion bootstrapping for domain independent lexicon and template acquisition*. In: Proceedings of the Australasian Language Technology Association Workshop, Pages 97-105.

Tara McIntosh and James R. Curran. 2009. *Reducing semantic drift with bagging and distributional similarity.* In: Proceedings of ACL-09, Pages 396-404.

Patrick Pantel and Dekang Lin. 2002. *Discovering word senses from text*. In: Proceedings of KDD-08, Pages 613-619.

Patrick Pantel and Deepak Ravichandran. 2004. *Automatically Labeling Semantic Classes.* In: Proceedings of HLT/NAACL, Pages 321-328, Volume 4.

Patrick Pantel and Marco Pennacchiotti. 2006. *Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations.* In: Proceedings of ACL-06, Pages 113–120.

Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu and Vishnu Vyas. 2009. *Web-Scale Distributional Similarity and Entity Set Expansion*. In: Proceedings of EMNLP-09, Pages 938-947.

Marius Pasca. 2007. *Weakly-supervised discovery of named entities using web search queries*. In: Proceedings of CIKM-07, Pages 683-690.

Marco Pennacchiotti, Patrick Pantel. 2011. *Automatically building training examples for entity extraction.* In: Proceedings of CoNLL-11, Pages 163-171.

Ellen Riloff and Rosie Jones. 1999. *Learning dictionaries for information extraction using multi-level bootstrapping*. In: Proceedings of AAAI-99, Pages 474-479.

Partha P. Talukdar, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. *Weakly-supervised acquisition of labeled class instances using graph random walks*. In: Proceedings of EMNLP-08, Pages 582-590.

Michael Thelen and Ellen Riloff. 2002. *A bootstrapping method for learning semantic lexicons using extraction pattern contexts*. In: Proceedings of ACL-02, Pages 214-221.

Richard C. Wang and William W. Cohen. 2008. *Iterative Set Expansion of Named Entities using the Web*. In: Proceedings of ICDM-08, Pages 1091-1096.

Richard C. Wang and William W. Cohen. 2009. *Automatic Set Instance Extraction using the Web*. In: Proceedings of ACL-09, Pages 441-449.

Vishnu Vvas, Patrick Pantel and Eric Crestan. 2009. *Helping editors choose better seed sets for entity set expansion*. In: Proceedings of CIKM-09, Pages 225-234

Roman Yangarber, Winston Lin and Ralph Grishman. 2002. *Unsupervised learning of generalized names*. In: Proceedings of COLING-02, Pages 1-7.

Separating Brands from Types: an Investigation of Different Features for the Food Domain

Michael Wiegand and **Dietrich Klakow**
Spoken Language Systems
Saarland University
D-66123 Saarbrücken, Germany
{Michael.Wiegand|Dietrich.Klakow}@lsv.uni-saarland.de

Abstract

We examine the task of separating types from brands in the food domain. Framing the problem as a ranking task, we convert simple textual features extracted from a domain-specific corpus into a ranker without the need of labeled training data. Such method should rank brands (e.g. *sprite*) higher than types (*e.g. lemonade*). Apart from that, we also exploit knowledge induced by semi-supervised graph-based clustering for two different purposes. On the one hand, we produce an auxiliary categorization of food items according to the Food Guide Pyramid, and assume that a food item is a type when it belongs to a category unlikely to contain brands. On the other hand, we directly model the task of brand detection using seeds provided by the output of the textual ranking features. We also harness Wikipedia articles as an additional knowledge source.

1 Introduction

Brands play a significant role in social life. They are the subject matter of many discussions in social media. Their automatic detection for information extraction tasks is a pressing problem since, despite their unique property to refer to commercial products of specific companies, in everyday language they often occur in similar contexts as common nouns. A typical domain where such behaviour can be observed is the food domain, where **food brands** (e.g. *nutella* or *sprite*) are often used synonymously with the **food type**[1] of which the brand is a prototypical instance (e.g. *chocolate spread* or *lemonade*). Such usage is illustrated in (1) and (2).

(1) In the evening, I eat a slice of bread with either **nutella** or marmalade.

(2) I prepare my pancakes with baking soda, water and a lacing of **sprite** instead of sugar.

This particular phenomenon of metonymy (Lakoff and Johnson, 1980), commonly referred to as *generic-ized trademarks*, of course, has consequences on automatic lexicon induction methods. If one automatically extracts food types, one also obtains food brands.

In this paper, we examine features to detect brands automatically. Solving the issue with the help of a manually-compiled list of brands neglects parts of the nature of brands. Brands come and go. Some products may be discontinued after a certain amount of time (e.g. due to limited popularity) while, on the other hand, new products constantly enter the market. For instance, popular food brands, such as *sierra mist* or *kazoozles*, did not exist a decade ago. Therefore, a list of brands that is manually created today may not reflect the predominant food brands that will be available in a decade.

The features we introduce to detect brands consider both the intrinsic properties of brands and their contextual environment. Even though in many contexts, brands are used as ordinary type expressions (1), there might be specific contexts that are only observed with brands. We also consider distributional properties: brands may co-occur with other brands. Moreover, they may be biased towards certain categories, e.g. sweets, beverages etc. For the latter, we actually exploit the usage of food brands to be

[1] We define *food type* as common nouns that denote a particular type of food, e.g. *apple*, *chocolate*, *cheese* etc.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2291–2302, Dublin, Ireland, August 23-29 2014.

Method	Corpus	Corpus Type	P@10	P@100	P@500
ranking by frequency	chefkoch.de	domain specific	0.00	22.00	25.60
induction based on coordination	Wikipedia	open domain	90.00	60.00	47.80
induction based on coordination	chefkoch.de	domain specific	100.00	98.00	92.00

Table 1: Precision at rank n (P@n) of different food induction methods.

Label	Items	Examples
Food Types	1745	apple, baguette, beer, corn flakes, crisps, basmati rice, broccoli, chocolate spread, gouda, orange juice, pork, potato, steak, sugar
Food Brands	221	activia, babybel, becel, butterfinger, kit kat, nutella, pepsi, philadelphia, smacks, smarties, sprite, ramazzotti, tuborg, volvic

Table 2: Gold standard of the food vocabulary.

used as genericized trademarks, allowing food categorization methods for types to be easily extended to brands. Moreover, we examine how external knowledge resources, such as Wikipedia, can be harnessed as a means to separate brands from types. Our task is lexicon construction rather than contextual entity classification, that is, we are interested in what a food item generally conveys and not what it conveys in a specific context.

We consider the food domain as a target domain since there are large, unlabeled domain-specific corpora available gathered from social media which are vital for the methods we explore. It is also a domain for which there has already been done research in the area of natural language processing (NLP), and there are common applications, such as virtual customer advice or product recommendation, that may exploit such NLP technology.

The methods we consider require no, or hardly any human supervision. Thus, we imagine that they can also be applied to other domains at a low cost. In particular, other life-style domains, such as fashion, cosmetics or electronics show parallels, since comparable textual web data from which to extract domain-specific knowledge are available.

Our experiments are carried out on German data, but our findings should carry over to other languages since the issues we address are (mostly) language universal. All examples are given as English translations. We use the term **food item** to refer to the union of food brands and food types. All food items will be written in lowercase reflecting the identical case spelling in German, i.e. types and brands are *both* written uppercase. In English, both types and brands can be written uppercase or lowercase[2], however, there is a tendency in user-generated content/social media to write mostly lowercase.

2 Motivation & Data

Previous research on lexicon induction proposed a widely applicable method based on *coordination* (Hatzivassiloglou and McKeown, 1997; Riloff and Shepherd, 1997; Roark and Charniak, 1998): First, a set of seed expressions that are typical of the categories one wants to induce are defined. Then, additional instances of those categories are obtained by extracting *conjuncts* of the seed expressions (i.e. all expressions that match *<seed> and/or <expression>* are extracted as new instances). A detailed study of such lexicon induction has recently been published by Ziering et al. (2013), who also point out the great semantic coherence of conjuncts.

This method can also be applied to the food domain. As a domain-specific dataset for all our experiments, we use a crawl of *chefkoch.de*[3] (Wiegand et al., 2012) consisting of $418,558$ webpages of forum entries. *chefkoch.de* is the largest German web portal for food-related issues. Table 1 shows the effectiveness of coordination as a means of extracting food items from our domain-specific corpus. Given a seed set of 10 frequent food items (we use: *water*, *salt*, *sugar*, *salad*, *bread*, *meat*, *cake*, *flour*, *butter* and

[2]There are plenty of food types that are written uppercase, e.g. *Jaffa Cakes*, *Beef Wellington*, *BLT*, *Hoppin' John* etc.

[3]`www.chefkoch.de`

Properties	Type of Property	Example	Brands	Types
nonwords	general	ebly, sprite, twix	41.63	-*NA*-
derived from proper noun	general	cheddar, evian, jim beam	31.22	2.29
foreign words	general	camembert, merci, wasabi	27.15	12.37
length	general	*average no. of characters*	7.97	10.53
word initial plosives	stylistic	p,t,k,b,d,g (*attract attention*)	31.22	35.81
assonance	stylistic	f**a**nt**a**, k**i**w**i** (fruit), p**a**p**a**y**a**	11.76	11.06
alliteration	stylistic	**b**a**b**y**b**el, **b**lue**b**erry, **t**ic **t**ac	6.79	3.78
onomatopoeia	stylistic	**crunch**ips, **pop**corn	2.71	0.52
rhyme	stylistic	j**elly** b**elly**, h**ubba** b**ubba**	1.35	0.06

Table 3: Comparison of intrinsic properties between brands and types; brands are always underlined; all numbers (except for *length*) are the proportion with the respective property.

potato), we compute all conjuncts and rank them according to frequency. We do this on our domain-specific corpus and on *Wikipedia*. As a baseline, we simply sort all nouns according to frequency in our domain-specific corpus. The table shows that ranking by frequency is no effective method. Conjuncts produce good results provided that they are extracted from a domain-specific corpus.

Even though coordination is a very reliable method to induce food items, it fails to distinguish between food types and food brands. We produced a labeled food vocabulary to be used for all our subsequent experiments consisting of food types and food brands (see Table 2). The food types exclusively comprise the food vocabulary from Wiegand et al. (2014). The food brands were manually selected with the help of the web. We only include food items that occur at least 5 times in our corpus. In our food vocabulary, 87% of our food brands occur as a conjunct of a food type. Therefore, the problem of confusing brands with types is inherent to induction based on coordination.

3 Intrinsic Properties

Table 3 provides some statistics on intrinsic properties of our food items giving some indication which feature types might be used for this task. We also include some stylistic properties of brands that have been addressed in previous marketing research and applied psychology. We focus on fairly straightforward features from *desirable brand name characteristics* (Robertson, 1989), since we assume that there is more general agreement on the underlying concepts than there is on the concepts underlying complex sound symbolism (Klink, 2000; Yorkston and Menon, 2004). For the statistics in Table 3, most properties (i.e. all except *length* and *word-initial plosives*) have been detected manually. The reason for this is that their automatic detection is not trivial (e.g. there is no established algorithm to detect onomatopoeia; even the detection of rhyme or assonance is not straightforward given the low grapheme-phoneme correspondence of English). We did not want the statistics for this exploratory experiment to be distorted by error-proneness of the detection methods.

Table 3 shows that a large part of brands are nonwords indicating that this task is hard to be solved with intrinsic features only. Since there is a high number of brands that are derived from some *existing* proper noun being either a person or a location, named-entity recognition might be applied to this task. Many brands are also foreign words. Unfortunately, applying language checking software on our food items turned out to perform poorly. (These tools are only effective on longer texts, e.g. sentences or entire documents, and do not work on isolated words, as in our problem setting.) We also noticed a difference in average word length between brands and types which is consistent with Robertson (1989) who claims that brand names should be *simple*. Most stylistic features seem to be less relevant to our task as they are either too infrequent or not discriminative. Therefore, we do *not* consider them as features for the detection of brands in our forthcoming experiments.

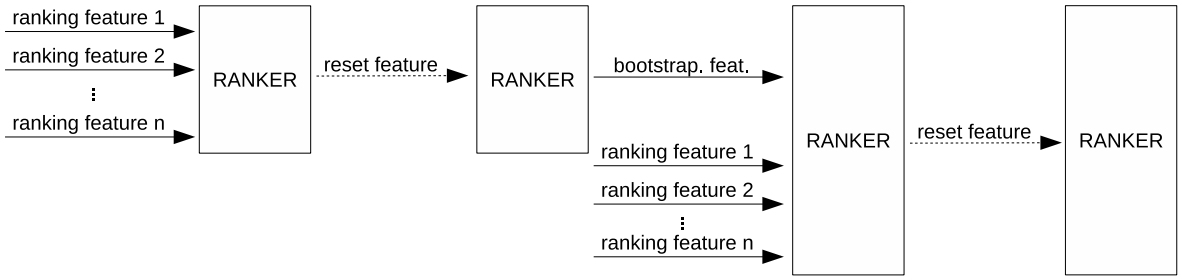


Figure 1: Processing Pipeline for ranking.

Feature Type	Features
ranking feature	LENGTH, COMMERCE, NER$_{target}$, NER$_{context}$, DIVERS, PAT$_{mod}$, PAT$_{pp}$
reset feature	GRAPH$_{pyramid}$
bootstrapping feature	GRAPH$_{brand}$, WIKI, VSM

Table 4: Feature classification.

4 Method

Our aim is to determine predictive features for the detection of brands. Rather than employing some supervised learner that requires manually labeled training data, we want to convert these features directly into a classifier without costly labeled data. We conceive this task as a ranking task. The reason for using a ranking is that our features can be translated into a ranking score in a very straightforward manner. For the evaluation, we do not have to determine some empirical threshold separating the category *brand* from the category *type*. Instead, the evaluation measures we employ for ranking implicitly assume highly ranked instances as brands and instances ranked at the bottom as types.

For the ranking task, we employ the processing pipeline as illustrated in Figure 1. Most of our features are designed in such a way that they assign a ranking score to each of our food items by counting how often a feature is observed with a food item; that is why we call these features **ranking features**. The resulting ranking should assign high scores to food brands and low scores to food types. If we want to combine several features into one ranking, we simply average for each food item the different ranking scores of the individual ranking features. This is possible since they have the same range $[0; 1]$. We obtain such range by normalizing the number of occurrences of a feature with a particular food item by the total number of occurrences of that food item. The combination by averaging is unbiased as it treats all features equally.

We also introduce a **reset feature** which is applied on top of an existing ranking provided by ranking features. A reset feature is a negative feature in the sense that it is usually a reliable cue that a food item is *not* a brand. If it fires for a particular food item, then its ranking score is reset to 0.

Finally, we add **bootstrapping features**. These features produce an output similar to the ranking features (i.e. another ranking). However, unlike the ranking features, the bootstrapping features produce their output based on a weakly-supervised method which requires some labeled input. Rather than manually providing that input, we derive it from the combined output that is provided by the ranking and reset features. We restrict ourselves to instances with a high-confidence prediction, which translates to the top and bottom end of a ranking. (Since the instances are not manually labeled, of course, not every label assignment will be correct. We hope, however, that by restricting to instances with a high-confidence prediction, we can reduce the amount of errors to a minimum.) The output of a bootstrapping feature is combined with the set of ranking features to a new ranking onto which again a reset feature is applied.

Table 4 shows which feature (each will be discussed below) belongs to which of the above feature types (i.e. ranking, reset or bootstrapping features). Most features (i.e. all except WIKI) are extracted from our domain-specific corpus introduced in §2.

4.1 Length

Since we established that brands tend to be shorter than types (§3), we add one feature that ranks each food item according to its number of characters.

4.2 Target Named-Entity Recognition (NER$_{target}$)

Brands can be considered a special kind of named entities. We apply a part-of-speech tagger to count how often a food item has been tagged as a proper noun. We decided against a named-entity recognizer as it usually only recognizes persons, locations and organizations, while part-of-speech taggers employ a general tag for all proper nouns (that may go well beyond the three afore-mentioned common types). We use a statistical tagger, i.e. *TreeTagger* (Schmid, 1994), that also employs features below the word level. As many of our food items will be unknown words, a character-level analysis may still be able to make useful predictions.

4.3 Contextual Named-Entity Recognition (NER$_{context}$)

We also count the number of other named entities that co-occur with the target food brand within the same sentence. We are only interested in organizations; an organization co-occurring with a brand is likely to be the company producing that brand (e.g. *He loves Kellogg's*$_{company}$ *frosties*$_{brand}$.) For this feature, we rely on the output of a named-entity recognizer for German (Chrupała and Klakow, 2010).

4.4 Diversification (DIVERS)

Once a product has established itself on the market for a substantial amount of time, many companies introduce variants of their brand to further consolidate their market position. The purpose of this diversification is to appeal to customers with special needs. A typical variant of food brands are *light* products. In many cases, the names of variants consist of the name of the original brand with some prefix or suffix indicating the particular type of variant (e.g. *mini babybel* or *philadelphia light*). We manually compiled 11 affixes and check for each food item how often it is accompanied by one of them.

4.5 Commerce Cues (COMMERCE)

Presumably, brands are more likely to be mentioned in the context of commercial transaction events than types. Therefore, we created a list of words that indicate these types of events. The list was created ad hoc. We used external resources, such as FrameNet (Baker et al., 1998) or GermaNet (Hamp and Feldweg, 1997) (the German version of WordNet (Miller et al., 1990)), and made no attempt to tune that list to our domain-specific food corpus. The final list (85 cues in total) comprises: verbs (and deverbal nouns) that convey the event of a commercial transaction (e.g. *buy*, *purchase* or *sell*), persons involved in a commercial transaction (e.g. *customer* or *shop assistant*), means of purchase (e.g. *money*, *credit card* or *bill*), places of purchase (e.g. *supermarket* or *shop*) and judgment of price (e.g. *cheap* or *expensive*).

4.6 Food Modifier (PAT$_{mod}$)

Even though many mentions of brands are similar to those of types, there exist some particular contexts that are mostly observed with brands. If the food item to be classified often occurs as a modifier of another food item, then the target item is likely to be some brand. This is due to the fact that many brands are often mentioned in combination with the food type that they represent, e.g. ***volvic*** *mineral water*, ***nutella*** *chocolate spread.*

4.7 Prepositional Phrase Embedding (PAT$_{pp}$)

Instead of appearing as a modifier (§4.6), a brand may also be embedded in some prepositional phrase that has a similar meaning, e.g. *We only buy the chocolate spread* $[$*by* ***nutella***$]_{PP}$.

4.8 Graph-based Methods (GRAPH)

We also employ some semi-supervised graph clustering method in order to assign semantic types to food items as introduced in Wiegand et al. (2014). The underlying data structure is a food graph that is generated automatically from our domain-specific corpus where nodes represent food items and edge weights

Category	Description	General	Brands
MEAT	meat and fish (products)	19.48	1.31
BEVERAGE	beverages (incl. alcoholic drinks)	17.19	**23.96**
SWEET	sweets, pastries and snack mixes	14.90	**25.60**
SPICE	spices and sauces	10.53	2.42
VEGE	vegetables (incl. salads)	10.38	0.00
STARCH	starch-based side dishes	9.21	4.42
MILK	milk products	6.71	**23.48**
FRUIT	fruits	4.48	1.14
GRAIN	grains, nuts and seeds	3.41	0.00
FAT	fat	2.54	**20.00**
EGG	eggs	0.92	0.00

Table 5: Proportion of categories in the entire food vocabulary (*General*) and among brands (*Brands*).

represent the similarity between different items. The weights are computed based on the frequency of co-occurrence within a similarity pattern (e.g. *X instead of Y*). Food items that cluster with each other in such a graph (i.e. food items that often co-occur in a similarity pattern) are most likely to belong to the same class. For the detection of brands, we examine two different types of food categorization. We always use the same clustering method (Wiegand et al., 2014) and the same graph. Depending on the specific type of categorization, we only change the seeds to fit the categories to be induced.

4.8.1 Categories of the Food Guide Pyramid (GRAPH$_{pyramid}$)

The first categorization we consider is the categorization of food items according to the *Food Guide Pyramid* (U.S. Department of Agriculture, 1992) as examined in Wiegand et al. (2014). We observed that food brands are not equally distributed throughout the entire range of food items. There is a notable bias of food brands towards beverages (mostly soft drinks and alcoholic drinks), sweets, snack mixes, dairy products and fat. Other categories, e.g. nuts, vegetables or meat, hardly contain brands.[4] The category inventory and the proportion among types and brands are displayed in Table 5.

We use the category information as a negative feature, that is, we re-set the ranking score to 0 if the category of the food item is either MEAT, SPICE, VEGE, STARCH, FRUIT, GRAIN or EGG. In order to obtain a category assignment to our food vocabulary, we re-run the best configuration from Wiegand et al. (2014) including the choice of category seeds. We just extend the graph that formerly only contained food types by nodes representing brands. We use no manually-compiled knowledge regarding food brands. Even though the seed food items are exclusively food types, we hope to be also able to make inferences regarding food brands. This is illustrated in Figure 2(a): The brand *mars* can be grouped with food types that are sweets, therefore, we conclude that *mars* is also some sweet. (Brands can be grouped with food types of their food category, since food brands are often used as if they were types (§1)). Since sweets are plausible candidates for brands (Table 5), *mars* is likely to be some brand.

We think that such bias of brands towards certain subcategories is also present in other domains. For example, in the electronic domain laptops will have a much larger variety of brands than network cables. Similarly, in the fashion domain there exist much more shoe brands than sock brands.

4.8.2 Direct Graph Clustering Separating Brands from Types (GRAPH$_{brand}$)

We also apply graph clustering directly for the separation of brands from types, i.e. we assign some brand and type seeds and then run graph-based clustering (Figure 2(b)). In order to combine the output of this clustering with that of the previous methods, we interpret the confidence of the output as a ranking score. As we pursue an unsupervised approach, we do not manually label the seeds but rely on the output of a ranker using a combination of above features (Figure 1). Instances at the top of the ranking are considered brand seeds, while instances at the bottom are considered type seeds.

[4]There may be companies which, among other things, also sell these food types, but we do not want to extract the names of *organizations* (as in traditional named-entity recognition), e.g. *Kraft Foods*, but specific product names, e.g. *philadelphia*.

(a) food type categorization (b) brand detection

Figure 2: Similarity graphs; **bold** items are seeds; line width of edges represents strength of similarity.

4.9 Wikipedia Bootstrapping (WIKI)

For many information extraction tasks, the usage of collaboratively-edited resources is increasingly becoming popular. One of the largest resources of that type is *Wikipedia*. For our vocabulary of food items, we could match 57% of the food brands and 53% of the food types with a Wikipedia article.

Even though Wikipedia may hold some useful information for the detection of brands, this information is not readily available in a structured format, such as *infoboxes*. This is illustrated by (3)-(5) which display the first sentence of three Wikipedia articles, where (3) and (4) are food brands and (5) is a food type. There is some thematic overlap across the two categories (e.g. (4) and (5) describe the ingredients of the food item). However, if one also considers the entire articles, some notable topical differences between brands and types become obvious. The articles of food brands typically focus on commercial aspects (i.e. market situation and product history) while articles of food types describe the actual food item (e.g. by distinguishing it from other food items or naming its origin). Therefore, a binary topic classification based on the *entire* document should be a suitable approach. In the light of the diversified language employed for articles on brands (cp. (3)-(4)), we consider a bag-of-words classifier more effective than applying some textual patterns on those texts.

(3) *BRAND:* **Twix** is a chocolate bar made by Mars, Inc.

(4) *BRAND:* **Smarties** is a brand under which Nestlé produces colour-varied sugar-coated chocolate lentils.

(5) *TYPE:* **Milk chocolate** is a type of chocolate made from cocoa produce (cocoa bean, cocoa butter), sugar, milk or dairy products.

Similar to GRAPH$_{brand}$ (§4.8.2), we harness Wikipedia via a bootstrapping method. We generate a labeled training set of Wikipedia articles representing brands and types using the combined output of the ranking features (+ reset feature). We then train a supervised classifier on these data and classify all articles representing food items of our food vocabulary. We use the output score of the classifier for the article of each food item (which amounts to some confidence score) and thus obtain a ranking score. For those food items for which no Wikipedia entry exists, we produce a score of 0.

4.10 Vector Space Model (VSM)

While GRAPH$_{brand}$ (§4.8.2) determines similar food items by means of highly weighted edges in a similarity graph (that represent the frequency of co-occurrences with a similarity pattern), we also examine whether distributional similarity can be harnessed for the same purpose. We represent each food item as a vector, where the vector components encode the frequency of words that co-occur with mentions of the food item in a fixed window of 5 words (in our domain-specific corpus). Similar to GRAPH$_{brand}$ (§4.8.2) and WIKI (§4.9), we consider the n highest and m lowest ranked food items provided by ranking features (+ reset feature) as labeled brand and type instances for a supervised classifier. For testing, we apply this classifier on each food item in our vocabulary, or more precisely, its vector representation. Thus we obtain another ranking score (again, the output amounts to some confidence score).

	Plain					+Graph$_{pyramid}$ *(reset feature)*				
Feature	**P@10**	**P@50**	**P@100**	**P@200**	**AP**	**P@10**	**P@50**	**P@100**	**P@200**	**AP**
RANDOM	10.00	18.00	14.00	14.00	0.119	20.00	22.00	22.00	21.50	0.167
LENGTH	10.00	20.00	22.00	21.50	0.163	10.00	32.00	41.00	40.00	0.230
DIVERS	60.00	46.00	37.00	25.00	0.207	60.00	50.00	39.00	30.50	0.240
COMMERCE	30.00	28.00	31.00	27.00	0.220	40.00	38.00	39.00	35.00	0.294
NER$_{context}$	70.00	72.00	52.00	43.50	0.401	80.00	72.00	51.00	46.50	0.425
PAT$_{pp}$	90.00	78.00	64.00	50.00	0.439	**100.00**	78.00	69.00	53.00	0.476
PAT$_{mod}$	60.00	68.00	**69.00**	58.00	0.460	90.00	76.00	**76.00**	58.00	0.507
NER$_{target}$	80.00	70.00	60.00	52.50	0.479	80.00	78.00	72.00	61.50	0.525
combined	**100.00**	**88.00**	66.00	**59.00**	**0.612**	**100.00**	**86.00**	**76.00**	**62.50**	**0.626**

Table 6: Precision at rank n (P@n) and average precision (AP) of the different ranking features.

Partition	**Prec**	**Rec**	**F**
Food Types	70.49	72.82	71.04
Food Brands	69.09	66.21	64.93

Table 7: Performance of food categorization according to the *Food Guide Pyramid* (auxiliary classification).

5 Experiments

In the following experiments, we mostly evaluate rankings. For that we employ *precision at rank n* and *average precision*. The former computes precision at a predefined rank n, whereas the latter provides an average of the precisions measured at every possible rank. While average precision provides a score that evaluates the ranking as a whole, precision at rank n typically focuses on the correctness of higher ranks.[5]

5.1 Evaluation of Ranking Features

Table 6 (left half) displays the results of the individual and combined ranking features. As a trivial baseline, we also include RANDOM which is randomized ranking of the food items. The table shows that all features except LENGTH produce a notably better ranking than RANDOM. Following the inspection of intrinsic properties of brands in §3, it does not come as a surprise that NER$_{target}$ is the strongest feature. However, also the contextual features NER$_{context}$, PAT$_{pp}$ and PAT$_{mod}$ produce reasonable results. If we combine all features (except the poorly performing LENGTH), we obtain a notable improvement over NER$_{target}$ which proves that those different features are complementary to a certain extent.

5.2 Evaluation of the Reset Feature

In Table 7, we examine the food categorization according to the Food Guide Pyramid as such. For this evaluation, we partition the output of automatic categorization into (actual) types and brands. Thus we can compare the performance between those two different types of food items, and can quantify the loss on the categorization on brands against the categorization on types. (Due to the fact that the seeds exclusively comprise types, we must assume that performance on brands will be lower.)[6] Even though there is a slight loss on brands (mostly recall), we still consider this categorization useful for our purposes.

[5]The manually labeled food vocabulary is available at:
`www.lsv.uni-saarland.de/personalPages/michael/relFood.html`

[6]Since the categories to indicate unlikely brands (§4.8.1) are extremely sparse (Table 5), we conflate them for this evaluation as one large category NEGATIVE. Because of this and due to the fact that the food type vocabulary is slightly smaller than the one used in Wiegand et al. (2014) (since we only consider food items mentioned at least 5 times in our corpus (§2)), the performance scores of food categorization in Table 7 and the one reported in Wiegand et al. (2014) differ accordingly.

Classifier		Acc	Prec	Rec	F
Baselines	Majority-Class Classifier	88.76	44.38	50.00	47.02
	seeds only: 50 top+150 bottom	9.51	91.00	13.85	23.47
	seeds only: 100 top+300 bottom	18.57	86.17	25.48	37.81
Bootstrap. Features	WIKI (*seeds: 50 top+150 bottom*)	43.95	**87.68**	43.33	57.91
	VSM (*seeds: 100 top+300 bottom*)	77.87	64.93	**81.61**	66.39
	GRAPH$_{brand}$ (*seeds: 100 top+300 bottom*)	82.91	81.36	67.27	**73.53**

Table 8: Bootstrapping features in isolation compared with baselines (i.e. reference classifiers).

Table 6 (right half) shows the performance of the corresponding reset feature on the brand detection task. We observe a systematic increase in performance when added on top of the ranking features.

5.3 Evaluation of Bootstrapping Features

Table 9 displays the performance of the bootstrapping features. For the labeled training data, we empirically determined the optimal class ratio (1:3) and the optimal number of seeds (the top 100 and bottom 300 items for VSM and GRAPH$_{brand}$, and top 50 and bottom 150 items for WIKI). As a supervised classifier for VSM and WIKI, we chose Support Vector Machines using *SVM^{light}* (Joachims, 1999).

The table shows that only GRAPH$_{brand}$ and WIKI improve the ranking, whereas WIKI is notably stronger. These results suggest that Wikipedia is a good resource from which to learn whether a food item is a brand or not. However, this task could not be completely solved by WIKI since not all food items are covered by Wikipedia (§4.9). To further prove this, we also evaluate an upper bound of Wikipedia, WIKI$_{oracle}$ (exclusively using that resource), in which we pretend to correctly interpret every Wikipedia page as an article for either a food brand or a food type. We rank all brands having a Wikipedia article highest. They are followed by those food items having no article (ordered randomly) and, finally, by the food types having a Wikipedia article. Table 9 shows that we are able to outperform WIKI$_{oracle}$.

Our pipeline (Figure 1) applies the reset feature at two stages. We also examine whether it is necessary to apply that feature for a second time. Presumably, the bootstrapping feature is so effective that we do not have to apply further type filtering. After all, the reset feature will also downweight some correct food items (Table 5). Table 9 confirms that when the reset feature is applied only once, we obtain a better performance (according to average precision) for all bootstrapping features (even for VSM).

Finally, Table 8 evaluates the bootstrapping features in isolation. Since, unlike the ranking features, the bootstrapping features provide a definite classification for each food item (in addition to a prediction score evaluated as a ranking score), we consider the output for a binary classification task. In this setting, we make use of the four evaluation measures *accuracy*, *precision*, *recall* and *F-score*. For the last three measures, we always compute the macro average score.

As a baseline, we also include a majority-class classifier that always predicts the class *food type*. Interestingly, in terms of F-score, GRAPH$_{brand}$ is the best method rather than WIKI, i.e. the best method from the previous evaluation in Table 9. The reason for this is that we evaluate in isolation rather than in combination with other features (i.e. parts of the additional benefit included in GRAPH$_{brand}$ may already be contained in ranking and reset features). Secondly, in a ranking task (Table 9), good performance is usually achieved by classifiers biased towards a high precision. Indeed, the best ranker in Table 9, i.e. WIKI, achieves the highest precision in Table 8.

6 Related Work

Ling and Weld (2012) examine named-entity recognition on data that also include brands, however, the class of brands is not explicitly discussed. Putthividhya and Hu (2011) explore brands in the context of product attribute extraction. Entities are extracted from eBay's clothing and shoe category. Nadeau et al. (2006) explicitly generate gazetteers of car brands obtained from corresponding websites. Those textual data are very restrictive in that they do not represent sentences but category listings or tables. In this paper, we consider as textual source a more general text type, i.e. forum entries, that comprise full

			-2nd reset	
Feature	**P@200**	**AP**	**P@200**	**AP**
WIKI$_{oracle}$	66.00	0.429	*-N/A-*	*-N/A-*
ranking+GRAPH$_{pyramid}$	62.50	0.626	*-N/A-*	*-N/A-*
ranking+GRAPH$_{pyramid}$+VSM	60.00	0.619	63.00	0.661
ranking+GRAPH$_{pyramid}$+GRAPH$_{brand}$	67.50	0.638	65.50	0.662
ranking+GRAPH$_{pyramid}$+WIKI	**70.00**	**0.688**	**73.00**	**0.718**

Table 9: Impact of bootstrapping; *-2nd reset:* does not apply reset feature for a second time (Figure 1).

sentences. Previous work also focuses on traditional (semi-)supervised algorithms. Hence, there are only few additional insights as to the specific properties of brand names. Min and Park (2012) examine the aspect of product instance distinction on the use case of product reviews on jeans from Amazon. Their work focuses on temporal features to identify distinct product instances (these may also include brand names).

The food domain has also recently received some attention. Different types of classification have been explored including ontology mapping (van Hage et al., 2005), part-whole relations (van Hage et al., 2006), recipe attributes (Druck, 2013), dish detection and the categorization of food types according to the Food Guide Pyramid (Wiegand et al., 2014). Relation extraction tasks have also been examined. While a strong focus is on food-health relations (Yang et al., 2011; Miao et al., 2012; Kang et al., 2013; Wiegand and Klakow, 2013), relations relevant to customer advice have also been addressed (Wiegand et al., 2012; Wiegand et al., 2014). Beyond that, Chahuneau et al. (2012) relate sentiment information to food prices with the help of a large corpus consisting of restaurant menus and reviews. Druck and Pang (2012) extract actionable recipe refinements. To the best of our knowledge, we present the first work that explicitly addresses the detection of brands in the food domain. While brands as such present an additional dimension to previously examined types of categorization, we also show that the categorization according to the Food Guide Pyramid helps to decide whether a food item is a brand or not.

7 Conclusion

We examined the task of separating types from brands in the food domain. Framing the problem as a ranking task, we directly converted predictive features extracted from a domain-specific corpus into a ranker without the need of labeled training data. Apart from those ranking features, we also exploited knowledge induced by semi-supervised graph-based clustering for two different purposes. On the one hand, we produced an auxiliary categorization of food items according to the Food Guide Pyramid, and assumed that a food item is a type when it belongs to a category that is unlikely to contain brands. On the other hand, we directly modelled the task of brand detection by using seeds provided by the output of the textual ranking features. We also learned additional high-precision knowledge from Wikipedia webpages using a similar bootstrapping scheme.

Acknowledgements

This work was performed in the context of the Software-Cluster project SINNODIUM. Michael Wiegand was funded by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IC10S01. The authors would like to thank Melanie Reiplinger for proofreading the paper.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of the International Conference on Computational Linguistics and Annual Meeting of the Association for Computational Linguistics (COLING/ACL)*, pages 86–90, Montréal, Quebec, Canada.

Victor Chahuneau, Kevin Gimpel, Bryan R. Routledge, Lily Scherlis, and Noah A. Smith. 2012. Word Salad: Relating Food Prices and Descriptions. In *Proceedings of the Joint Conference on Empirical Methods in Natural*

Language Processing and Computational Natural Language Learning (EMNLP/CoNLL), pages 1357–1367, Jeju Island, Korea.

Grzegorz Chrupała and Dietrich Klakow. 2010. A Named Entity Labeler for German: Exploiting Wikipedia and Distributional Clusters. In *Proceedings of the Conference on Language Resources and Evaluation (LREC)*, pages 552–556, La Valletta, Malta.

Gregory Druck and Bo Pang. 2012. Spice it up? Mining Refinements to Online Instructions from User Generated Content. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 545–553, Jeju, Republic of Korea.

Gregory Druck. 2013. Recipe Attribute Detection Using Review Text as Supervision. In *Proceedings of the IJCAI-Workshop on Cooking with Computers (CWC)*, Beijing, China.

Birgit Hamp and Helmut Feldweg. 1997. GermaNet - a Lexical-Semantic Net for German. In *Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15, Madrid, Spain.

Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the Semantic Orientation of Adjectives. In *Proceedings of the Conference on European Chapter of the Association for Computational Linguistics (EACL)*, pages 174–181, Madrid, Spain.

Thorsten Joachims. 1999. Making Large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press.

Jun Seok Kang, Polina Kuznetsova, Michael Luca, and Yejin Choi. 2013. Where Not to Eat? Improving Public Policy by Predicting Hygiene Inspections Using Online Reviews. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1443–1448, Seattle, WA, USA.

Richard R. Klink. 2000. Creating Brand Names with Meaning: The Use of Sound Symbolism. *Marketing Letters*, 11(1):5–20.

George Lakoff and Mark Johnson. 1980. *Metaphors We Live By*. University of Chicago Press.

Xiao Ling and Daniel S. Weld. 2012. Fine-Grained Entity Recognition. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 94–100, Toronto, Canada.

Qingliang Miao, Shu Zhang, Bo Zhang, Yao Meng, and Hao Yu. 2012. Extracting and Visualizing Semantic Relationships from Chinese Biomedical Text. In *Proceedings of the Pacific Asia Conference on Language, Information and Compuation (PACLIC)*, pages 99–107, Bali, Indonesia.

George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3:235–244.

Hye-Jin Min and Jong C. Park. 2012. Product Name Classification for Product Instance Distinction. In *Proceedings of the Pacific Asia Conference on Language, Information and Compuation (PACLIC)*, pages 289–298, Bali, Indonesia.

David Nadeau, Peter D. Turney, and Stan Matwin. 2006. Unsupervised Named-Entity Recognition: Generating Gazetteers and Resolving Ambiguity. In *Proceedings of the Canadian Conference on Artificial Intelligence*, pages 266–277, Québec City, Québec, Canada.

Duangmanee Putthividhya and Junling Hu. 2011. Bootstrapped Named Entity Recognition for Product Attribute Extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1557–1567, Edinburgh, Scotland, UK.

Ellen Riloff and Jessica Shepherd. 1997. A Corpus-Based Approach for Building Semantic Lexicons. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 117–124, Providence, RI, USA.

Brian Roark and Eugene Charniak. 1998. Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 1110–1116, Montreal, Quebec, Canada.

Kim Robertson. 1989. Strategically Desirable Brand Name Characteristics. *Journal of Comsumer Marketing*, 6(4):61–71.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49, Manchester, United Kingdom.

Human Nutrition Information Service U.S. Department of Agriculture. 1992. The Food Guide Pyramid. Home and Garden Bulletin 252, Washington, D.C., USA.

Willem Robert van Hage, Sophia Katrenko, and Guus Schreiber. 2005. A Method to Combine Linguistic Ontology-Mapping Techniques. In *Proceedings of International Semantic Web Conference (ISWC)*, pages 732 – 744, Galway, Ireland. Springer.

Willem Robert van Hage, Hap Kolb, and Guus Schreiber. 2006. A Method for Learning Part-Whole Relations. In *Proceedings of International Semantic Web Conference (ISWC)*, pages 723 – 735, Athens, GA, USA. Springer.

Michael Wiegand and Dietrich Klakow. 2013. Towards Contextual Healthiness Classification of Food Items – A Linguistic Approach. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, pages 19–27, Nagoya, Japan.

Michael Wiegand, Benjamin Roth, and Dietrich Klakow. 2012. Web-based Relation Extraction for the Food Domain. In *Proceedings of the International Conference on Applications of Natural Language Processing to Information Systems (NLDB)*, pages 222–227, Groningen, the Netherlands. Springer.

Michael Wiegand, Benjamin Roth, and Dietrich Klakow. 2014. Automatic Food Categorization from Large Unlabeled Corpora and Its Impact on Relation Extraction. In *Proceedings of the Conference on European Chapter of the Association for Computational Linguistics (EACL)*, pages 673–682, Gothenburg, Sweden.

Hui Yang, Rajesh Swaminathan, Abhishek Sharma, Vilas Ketkar, and Jason D'Silva, 2011. *Learning Structure and Schemas from Documents*, volume 375 of *Studies in Computational Intelligence*, chapter Mining Biomedical Text Towards Building a Quantitative Food-disease-gene Network, pages 205–225. Springer Berlin Heidelberg.

Eric Yorkston and Geeta Menon. 2004. A Sound Idea: Phonetic Effects of Brand Names on Consumer Judgments. *Journal of Consumer Research*, 31:43–51.

Patrick Ziering, Lonneke van der Plas, and Hinrich Schuetze. 2013. Bootstrapping Semantic Lexicons for Technical Domains. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, Nagoya, Japan, 1321–1329.

Unsupervised Instance-Based Part of Speech Induction Using Probable Substitutes

Mehmet Ali Yatbaz **Enis Rıfat Sert** **Deniz Yuret**
Koç University Artificial Intelligence Laboratory, İstanbul
{myatbaz,esert,dyuret}@ku.edu.tr

Abstract

We develop an instance (token) based extension of the state of the art word (type) based part-of-speech induction system introduced in (Yatbaz et al., 2012). Each word instance is represented by a feature vector that combines information from the target word and probable substitutes sampled from an n-gram model representing its context. Modeling ambiguity using an instance based model does not lead to significant gains in overall accuracy in part-of-speech tagging because most words in running text are used in their most frequent class (e.g. 93.69% in the Penn Treebank). However it is important to model ambiguity because most frequent words are ambiguous and not modeling them correctly may negatively affect upstream tasks. Our main contribution is to show that an instance based model can achieve significantly higher accuracy on ambiguous words at the cost of a slight degradation on unambiguous ones, maintaining a comparable overall accuracy. On the Penn Treebank, the overall many-to-one accuracy of the system is within 1% of the state-of-the-art (80%), while on highly ambiguous words it is up to 70% better. On multilingual experiments our results are significantly better than or comparable to the best published word or instance based systems on 15 out of 19 corpora in 15 languages. The vector representations for words used in our system are available for download for further experiments.

1 Introduction

Unsupervised part-of-speech (POS) induction aims to classify words into syntactic categories using unlabeled, plain text input. The problem of induction is important for studying under-resourced languages that lack labeled corpora and high quality dictionaries. It is also essential in modeling child language acquisition because every child manages to induce syntactic categories without access to labeled sentences, labeled prototypes, or dictionary constraints (Ambridge and Lieven, 2011). Categories induced from data may point to shortcomings or inconsistencies of hand-labeled categories as discussed in Section 4. Finally, the induced categories or the vector representations generated by the induction algorithms may improve natural language processing systems when used as additional features.

Word-based POS induction systems classify different instances of a word in a single category (which we will refer to as the *one-tag-per-word assumption*). Instance-based systems classify each occurence of a word separately and can handle ambiguous words.

Examples of word-based systems include ones that represent each word with the vector of neighboring words (context vectors) and cluster them (Schütze, 1995; Lamar et al., 2010b; Lamar et al., 2010a), use a prototypical bi-tag HMM that assigns each word to a latent class (Brown et al., 1992; Clark, 2003), restrict a HMM based Pitman-Yor process to perform one-tag-per-word inference (Blunsom and Cohn, 2011), define a word-based Bayesian multinomial mixture model (Christodoulopoulos et al., 2011), or

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2303–2313, Dublin, Ireland, August 23-29 2014.

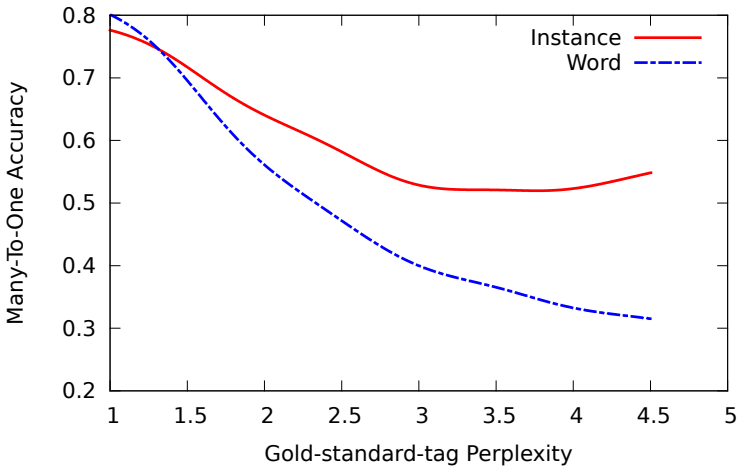


Figure 1: The accuracy comparison of word and instance based part-of-speech induction models as a function of target word ambiguity (as measured by gold-standard-tag perplexity described in Section 3.3) on the Penn Treebank.

construct word vector representations based on co-occurrences with contextual features (Yatbaz et al., 2012).

The obvious limitation of the one-tag-per-word assumption is that instances of ambiguous words that have more than one POS role are grouped into the same class. For example, the word *offer* is tagged as NN(399), VB(105) and VBP(34)[1] in its 538 occurrences in the human labeled Wall Street Journal (WSJ) Section of the Penn Treebank (PTB) corpus (Marcus et al., 1999). If all instances of *offer* are assigned to the most frequent tag NN, 36% (139/538) will be erroneously labeled. In spite of this shortcoming, word-based POS induction systems generally do well because the one-tag-per-word assumption is mostly accurate: 93.69% of the word occurrences are tagged with their most frequent POS tag in the PTB (Toutanova et al., 2003).

In order to handle ambiguous words, models without a strict one-tag-per-word assumption need to group word *instances* into clusters according to their contexts. Some of these instance-based models bias words to have few tags using sparse priors in a Bayesian setting (Goldwater and Griffiths, 2007; Johnson, 2007; Gao and Johnson, 2008), or posterior regularization (Ganchev et al., 2010). Schütze (1993) represents the context of a word instance by concatenating context vectors of its left and right neighboring words, and clusters word instances. Berg-Kirkpatrick et al. (2010) use an EM algorithm where they replace the multinomial components with miniature logistic regressions and achieve the highest instance-based accuracy on PTB. Christodoulopoulos et al. (2010) select prototypes of each cluster from the output of Brown (1992) and feed them to a HMM model that can handle prototypes as features (Haghighi and Klein, 2006). However none of these models achieve results comparable to the best word-based systems.

In this work, we show that one can build an instance-based system that can perform significantly better on highly ambiguous words (see Figure 1) and yet is competitive with word-based systems in overall accuracy.

We follow the state of the art word-based system (Yatbaz et al., 2012) and use probable substitutes of a word instance as its contextual features. The following examples illustrate how such paradigmatic (substitute based) contextual features can capture the similarity between two contexts where a syntagmatic (neighbor based) representation would fail:

[1]NN, VB and VBP are three POS tags from the Penn Treebank corpus and they correspond to singular noun, verb in base form and non-3^{rd}person singular verb in present tense, respectively. The numbers in parentheses are the frequencies.

(1) *"Pierre Vinken, 61 years old, will join the board as a nonexecutive* **director** *Nov. 29."*
director → chairman (.8242), director (.0127), directors (.0127) . . .

(2) *". . . Joseph Corr was succeeded by Frank Lorenzo,* **chief** *of parent Texas Air."*
chief → chairman (.9945), president (.0031), directors (.0012) . . .

Each sentence has the target words marked in bold (**director** and **chief**) and their likely substitutes listed with probabilities[2] in parentheses. Note that the two contexts have no words in common, therefore syntagmatic (neighbor based) contextual features will fail to capture their similarity. However, paradigmatic features such as the top substitutes "chairman", "directors", etc. clearly indicate the similarity and help place these two instances into the same category.

Following (Globerson et al., 2007), we embed words and their contextual, orthographic, and morphological features in a high dimensional Euclidean space that relates their joint probability to distance. In contrast to (Yatbaz et al., 2012) we build an *instance-based* POS induction system where each instance has a vector representation that concatenates the word vector with the average of the contextual feature vectors. We show that clustering of these instance vectors separate different roles of ambiguous words well, and achieve comparable or better performance than the best word-based systems in matching the gold tags on 19 corpora in 15 languages. All the code that can be used to replicate our findings is available at `https://github.com/ai-ku/upos_2014`.

Section 2 describes the instance-based POS induction algorithm, Section 3 gives the results of our experiments, Section 4 compares the output of the induction system with the gold tags, and Section 5 summarizes our contributions.

2 Algorithm

In this section, we describe the steps of our instance-based POS-induction algorithm:

1. Sample r substitutes for each word instance in the target corpus using an n-gram language model.
2. Construct r tuples for each instance where each tuple consists of a sampled substitute, the target word, and the morphological and orthographical features of the target word (see Table 1).
3. Construct Euclidean embeddings of each word and each feature based on all tuples following Gleberson et al.(2007) and Maron et al.(2010).
4. Construct a vector representation for each instance by concatenating the embedding of the target word with the average of its substitute embeddings.
5. Use k-means clustering to cluster the instance vectors where k is equal to the number of gold tags.

Steps 1 and 2 construct a tuple representation for each instance. Table 1 gives some example tuples for Sentence (1) from the previous section. In this example $r = 3$, so three substitutes are sampled for each instance as contextual features. The sampling is with replacement from the substitute word distribution of a context given by the n-gram language model, so some substitute words may be repeated. The target word and its other features are identical for each of the r tuples representing a single instance.

In step 3, we construct Euclidean embeddings for each unique word and feature value using the multivariable version of the CODE algorithm described in (Globerson et al., 2007). Given two categorical variables W and F, the CODE algorithm constructs Euclidean embeddings (vectors) for each of their distinct values in the same space. The distance between the embedding of a w value, $\phi(w)$, and the embedding of an f value, $\psi(f)$, is related to their joint distribution $p(w, f)$ as follows[3]

$$p(w, f) = \frac{1}{Z}\bar{p}(w)\bar{p}(f)e^{-d^2_{w,f}}$$

[2]Substitute probabilities are computed using a 4-gram language model.

[3](Globerson et al., 2007) describes several ways to relate distances to probabilities, the model used here is the marginal-marginal (MM) model.

Word	Subst	Suf	Cap	Num
Vinken	Makhlouf	–	T	F
Vinken	Makhlouf	–	T	F
Vinken	<unk>	–	T	F
61	20	–	F	T
61	2000	–	F	T
61	eleven	–	F	T
years	years	-s	F	F
years	years	-s	F	F
years	years	-s	F	F

Table 1: The tuples constructed for the instances of "Vinken", "61" and "years" from Sentence (1). The elements of each tuple are the target word, sampled substitute, suffix, capitalization, and number features.

where $\bar{p}$ represents empirical probabilities (frequencies from the training data), $d_{w,f}$ is the distance between the embeddings $\phi(w)$ and $\psi(f)$ and $Z = \sum_{w,f} \bar{p}(w)\bar{p}(f)e^{-d^2_{w,f}}$ is a normalization constant. Starting with random vectors for each distinct value of w and f, we use stochastic gradient ascent to move the embedding vectors around to maximize the likelihood given by this model. Calculating the normalization constant Z is the most expensive part of this procedure. We solve this problem following (Maron et al., 2010) who suggest that a constant Z approximation can be used if the embedding vectors are kept on the unit sphere.

As Table 1 shows, considering the target word and its contextual, morphological and orthographic features gives us more than two variables. Yatbaz et al. (2012) adopt the two variable CODE algorithm to this multi-variable case in an ad-hoc manner by considering the target word as w and all other features as distinct f values. We implement the multi-variable extension of CODE suggested by (Globerson et al., 2007) (Section 6.2) which optimizes the following likelihood function:

$$\ell(\phi, \psi^{(1)}, \ldots, \psi^{(K)}) = \sum_{i=1}^{K} \sum_{w, f^{(i)}} \bar{p}(w, f^{(i)}) \log p(w, f^{(i)})$$

where w are the target words, ϕ are the embeddings of target words, K is the number of different types of features[4], $f^{(i)}$ are the values of the i'th feature, and $\psi^{(i)}$ are the embeddings for the values of the i'th feature. This extension can be seen as a set of K bivariate CODE models $p(w, f^{(i)})$ that share the same target word embeddings $\phi(w)$ but build their own feature embeddings $\psi^{(i)}(f^{(i)})$.

Step 4 constructs a vector representation for each word instance with the concatenation of its word type embedding and the average of its r substitute embeddings. If the original embeddings are in d dimensional space, this results in a $2d$ dimensional vector representing an instance.

Step 5 clusters these $2d$ dimensional instance vectors using a modified k-means algorithm with smart initialization (Arthur and Vassilvitskii, 2007) and assigns each instance to one of k clusters.

3 Experiments

In this section we present our instance-based POS induction experiments. Section 3.1 describes the accuracy metrics that we use to evaluate our results. Section 3.2 details the test corpus and the experimental parameters used in the English experiments and compares our results with previous work. Section 3.3 compares the performance of type and instance based systems on ambiguous words. Finally, Section 3.4 extends the language and corpus coverage by applying the best performing instance based models to 19 corpora in 15 languages.

[4]For example the number of features $K = 4$ in Table 1: Subst, Suf, Cap, and Num.

Model	MTO	VM
Clark (2003)	.712	.655
Christodoulopoulos et al. (2011)	.728	.661
Berg-Kirkpatrick et al. (2010)	.755	-
Christodoulopoulos et al. (2010)	.761	.688
Blunsom and Cohn (2011)	.775	.697
Yatbaz et al. (2012)	.8023 (.0070)	.7207 (.0041)
Instance based (Sec. 2)	.7952 (.0030)	.6908 (.0027)

Table 2: Summary of results with MTO and VM scores for POS induction on the Penn Treebank. Standard errors are given in parentheses when available. All the models incorporate orthographic and morphological features. Berg-Kirkpatrick et al. (2010) and Christodoulopoulos et al. (2010) use instance based models.

3.1 Evaluation

We report many-to-one and V-measure scores for our experiments as suggested in (Christodoulopoulos et al., 2010). The many-to-one (MTO) evaluation maps each cluster to its most frequent gold tag and reports the percentage of correctly tagged instances. The MTO score can be increased by simply increasing number of clusters, thus the number of clusters is fixed to match the number of gold tags in each experiment. The V-measure (VM) (Rosenberg and Hirschberg, 2007) is an information theory motivated metric that calculates the harmonic mean of completeness and homogeneity of the clusters. Completeness of a cluster is maximized when all instances of a gold-tag are grouped into the same cluster and the homogeneity is maximized when the members of a cluster belong to the same gold-tag.

3.2 Experimental Settings and Results

To make a direct comparison with previously published results, the Wall Street Journal Section of the Penn Treebank was used as the test corpus (1,173,766 instances, 49,206 unique tokens) for English experiments. PTB uses 45 part-of-speech tags which we used as the gold standard for evaluation in our experiments.

To compute substitutes in a given context we trained a language model using the ukWaC corpus ($\approx$ 2 billion tokens) constructed by crawling the ".uk" Internet domain (Ferraresi et al., 2008)[5]. We used SRILM (Stolcke, 2002) to build a 4-gram language model with interpolated Kneser-Ney discounting. Words that were observed less than 2 times in the language model training data were replaced by <unk> tags, which gave us a vocabulary size of 4,254,946. The perplexity of the 4-gram language model on the PTB is 303 and the unknown word rate is 0.008. For computational efficiency only the top 100 substitutes and their probabilities were computed for each position in the PTB using the FASTSUBS algorithm (Yuret, 2012). We use the same orthographic features defined in (Yatbaz et al., 2012) and generated morphological features using the unsupervised algorithm Morfessor (Creutz and Lagus, 2005).

The experiments were run using the following default settings (unless otherwise stated): (1) each word was kept with its original capitalization; (2) 90 substitutes sampled per instance; (3) the learning rate parameters for S-CODE were set to $\varphi_0 = 50$, $\eta_0 = 0.2$; (4) S-CODE convergence threshold, the log-likelihood difference between two consecutive iterations, was set to 0.001; (5) the S-CODE dimensions and $\tilde{Z}$ were set to 25 and 0.166, respectively; (6) a modified k-means algorithm with smart initialization was used (Arthur and Vassilvitskii, 2007); (7) the number of k-means restarts was set to 128 to improve clustering and reduce variance.

Each experiment was repeated 10 times with different random seeds and the results are reported with standard errors in parentheses or error bars in graphs. Table 2 summarizes all the results reported in this section and the ones we cite from the literature.

[5]We use the Penn Treebank Tokenizer to make the training data compatible with PTB.

3.3 Word vs. Instance-Based Induction

Table 2 shows that the overall many-to-one accuracy of our instance based induction system is comparable to (Yatbaz et al., 2012)[6] and significantly higher than the other published results on the Penn Treebank. However Figure 1 in the introduction suggests that this summary hides the large difference in the answers given by the different systems. In this section we compare the performance of our instance-based model to the word-based model of (Yatbaz et al., 2012) on word types at different levels of ambiguity using the English Penn Treebank results.

We propose the gold-tag perplexity of a word as a measure of its degree of ambiguity defined as:

$$GP(w) = 2^{H(p_w)} = 2^{-\sum_t p_w(t) log_2 p_w(t)}$$

where w is a word, t is a tag, p_w is the gold POS tag distribution of the word w and $H(p_w)$ is the entropy of the p_w distribution. A GP of 1 for a word w indicates that w is always associated with the same POS tag. A word with N equally probable tags would have a GP of N.

Figure 1 plots the gold-tag perplexity versus the smoothed MTO accuracy for the word-based and the instance-based POS induction systems on the Penn Treebank. To compose the plot, we found the best mapping from the induced clusters to the gold-standard tags, then we computed the MTO accuracy for each word using this mapping and plotted the MTO as a function of the word's GP. We used the Nadaraya-Watson kernel regression estimate (Nadaraya, 1964; Watson, 1964) with normal kernel of bandwidth 1.0 to obtain smooth regression lines. The figure shows that the performance of the instance-based induction model does not degrade as much as the word-based model as the ambiguity of the words increase. However, only 14.94% of the instances in the PTB consists of words with GP greater than 1.5 and 45.71% consists of words with GP exactly 1. Thus, the overall accuracy numbers do not adequately reflect the improvement on highly ambiguous words.

3.4 Multilingual Experiments

Following Christodoulopoulos et al. (2011), we extend our experiments to 8 languages from MULTEXT-East (Bulgarian, Czech, English, Estonian, Hungarian, Romanian, Slovene and Serbian) (Erjavec, 2004) and 10 languages from the CoNLL-X shared task (Bulgarian, Czech, Danish, Dutch, German, Portuguese, Slovene, Spanish, Swedish and Turkish) (Buchholz and Marsi, 2006).

To sample substitutes, we trained language models of Bulgarian, Czech, Estonian, Romanian, Danish, German, Dutch, Portuguese, Spanish, Swedish and Turkish with their corresponding TenTen corpora (Jakubíček et al., 2013), and Hungarian, Slovene and Serbian with their corresponding Wikipedia dump files[7]. Serbian shares a common basis with Crotian and Bosnian therefore we trained 3 different language models using Wikipedia dump files of Serbian together with these two languages and measured the perplexities on the MULTEXT-East Serbian corpus. We chose the Croatian language model since it achieved the lowest perplexity score and unknown word ratio on MULTEXT-East Serbian corpus. We use ukWaC corpora to train English language models.

We used the default settings in Section 3.2 and incorporated only the orthographic features[8]. Extracting unsupervised morphological features for languages with different characteristics would be of great value, but it is beyond the scope of this paper. For each language the number of induced clusters is set to the number of tags in the gold-set. To perform meaningful comparisons with the previous work we train and evaluate our models on the training section of MULTEXT-East[9] and CONLL-X languages (Lee et al., 2010).

Table 3 presents the performance of our instance based model on 19 corpora in 15 languages together with the corresponding best published results from $^{\diamond}$(Yatbaz et al., 2012), ‡(Blunsom and Cohn, 2011),

[6]The difference is not statistically significant at $p = 0.05$.

[7]Latest Wikipedia dump files are freely available at `http://dumps.wikimedia.org/` and the text in the dump files can be extracted using WP2TXT (`http://wp2txt.rubyforge.org/`)

[8]All corpora (except German, Spanish and Swedish) label the punctuation marks with the same gold-tag therefore we add an extra *punctuation* feature for those languages.

[9]Languages of MULTEXT-East corpora do not tag the punctuations, thus we add an extra tag for punctuations to the tag-set of these languages.

	Language	Tags	Best Published MTO VM	Instance Based MTO / VM
WSJ	English	45	.802 / **.721** $^{\diamond}$	.795 / .691
MULTEXT-East	Bulgarian	12+1	.665 / **.556***	.664 / .513
	Czech	12+1	.642 / **.539***	**.705** / .511
	English	12+1	.733 / .633*	**.835 / .661**
	Estonian	11+1	.644 / **.533***	.643 / .457
	Hungarian	12+1	**.682** / **.548***	.647/ .459
	Romanian	14+1	.611 / .523*	**.660** / .528
	Slovene	12+1	**.679** / **.567***	.667 / .451
	Serbian	12+1	**.641** / **.510**†	.594/ .402
CoNLL-X Shared Task	Bulgarian	54	.704 / **.596**†	**.751** / .583
	Czech	12	.701‡ / .484*	.701 / .486
	Danish	25	.761‡ / .591*	.761 / .584
	Dutch	13	.711‡ / **.547***	.712 / .537
	German	54	.744* / **.630**†	.749 / .618
	Portuguese	22	.785‡ / **.639***	.782 / .607
	Slovene	29	.642* / **.539**†	.638 / .469
	Spanish	47	**.788**‡ / **.632***	.753/ .602
	Swedish	41	.682 / **.589**†	.681 / .546
	Turkish	30	.628 / .408*	.637 / .401

Table 3: The MTO and VM scores on 19 corpora in 15 languages together with number of induced clusters. Statistically significant results shown in bold ($p < 0.05$).

*(Christodoulopoulos et al., 2011) and †(Clark, 2003). All of the state-of-the-art systems in Table 3 are word-based and incorporate morphological features.

Our MTO results are lower than the best systems on all of data-sets that use language models trained on the Wikipedia corpora. ukWaC and TenTen corpora are cleaner and tokenized better compared to the Wikipedia corpora. These corpora also have larger vocabulary sizes and lower out-of-vocabulary rates. Thus language models trained on these corpora have much lower perplexities and generate better substitutes than the Wikipedia based models. Our model has lower VM scores in spite of good MTO scores on 14 corpora which is discussed in Section 4.

Among the languages for which clean language model corpora were available, our model performs comparable to or significantly better than the best systems on most languages. We show significant improvements on MULTEXT-East Czech, Romanian, and CoNLL-X Bulgarian. Our model achieves the state-of-the-art MTO on MULTEXT-East English and scores comparable MTO with the best model on WSJ. Our model shows comparable results on MULTEXT-East Bulgarian and Estonian, and CoNLL-X Czech, Danish, Dutch, German, Portuguese, Swedish and Turkish in terms of the MTO score. One reason for comparably low MTO on Spanish might be the absence of morphological features.

4 Discussion

In this section we perform further analysis on the clustering output of our model. The example below illustrates the advantage of the instance-based approach:

(1) ... it will also **offer** buyers the option ...
Substitutes: give, help, attract

(2) The **offer** is being launched ...
Substitutes: campaign, project, scheme

The word **offer** is a *verb* in the first sentence and a *noun* in the second one. Clustering the word embeddings can not distinguish the different occurrences of the words (Yatbaz et al., 2012). On the other hand, the substitutes of *offer* in the two sentences can disambiguate the correct category of the corresponding occurrences. In our actual experiments our instance based representation distinguishes the instances of **offer** as *noun* (cluster 26 and 12) and *verb* (cluster 35).

To illustrate how words are distributed in the induced clusters, we compare the most frequent clusters of our model in Section 3 with the most frequent gold-tags of PTB in Figure 2.

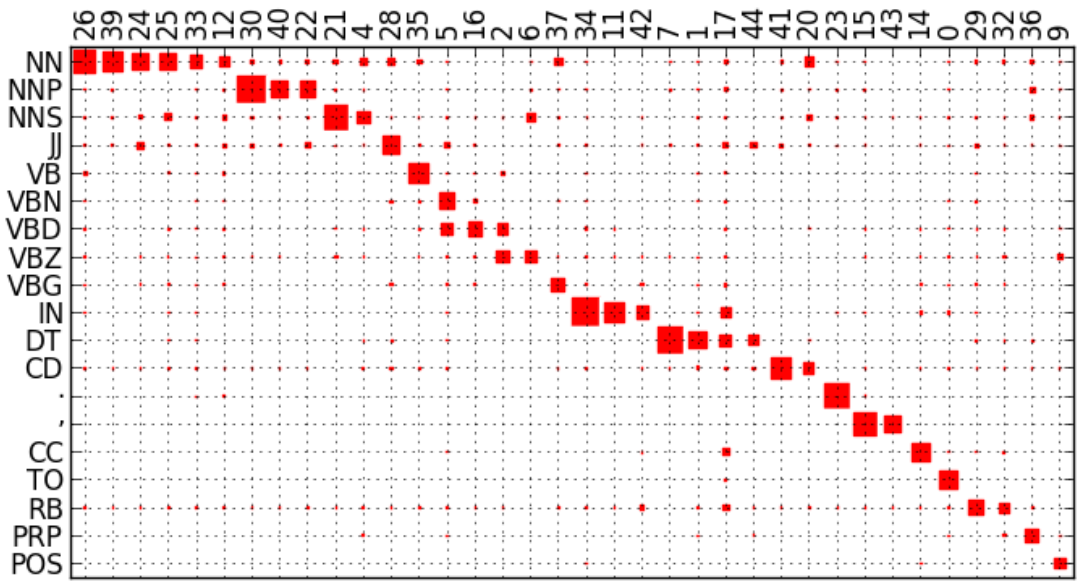


Figure 2: Each row corresponds to a gold tag and each column is an induced tag in the Hinton diagram above. The area of each square is proportional to the joint probability of the given tag and cluster.

The low VM performance of our instance-based model compared to the state-of-the-art word-based systems on some languages is due to the completeness part of the VM score. The Hinton diagram in Figure 2 shows that large gold-tag groups are split into several clusters based on the substitutability of words in that particular cluster (rows of the Hinton diagram). For example, proper nouns (*NNP*) are split into three major clusters such that titles like *Mr.* or person names are in (40), nationality or country related words like *Japanese* or U.S are in (22), and the rest of the proper nouns in cluster (30).

The gold-tags of PTB, on the other hand, do not always respect whether words with the same tag are substitutable for one another. Freudenthal et al. (2005) argues, from the child language acquisition perspective, that the standard linguistic definition of syntactic groups requires the substitutability of words in a syntactic category. Word pairs that are placed in the same category in the PTB, such as "Mr." and "Friday", "be" and "run", "not" and "gladly", "of" and "into" are clearly not generally substitutable.

Another noteworty example of completeness error is that our model splits the punctuation mark (,) class of PTB into the clusters 15 and 43 based on the different usage patterns. The majority of the (,) instances in cluster 15 are used in relative clauses, reported speech clauses or conjunctions while cluster 43 generally consists of (,) instances that are used in non-essential clauses (ex: Time, the largest newsweekly, . . .).

5 Contributions

Our main contributions can be summarized as follows:

- We introduced an instance based POS induction system that can handle ambiguous words and is competitive with the word-based systems in overall accuracy.
- We extended the S-CODE framework to handle more than two categorical variables.
- Our instance based system scores 79.5% many-to-one accuracy on the Penn Treebank and achieves results that are significantly better than or comparable with the best published systems on 15 out of 19 corpora in 15 languages.
- All our code and data, including the substitute distributions and word vectors for the PTB,

MULTEXT-East and CoNLL-X shared task corpora are available at the authors' website at https://github.com/ai-ku/upos_2014.

Acknowledgements

We would like to thank Adam Kilgarriff and the Sketch Engine[10] team for making their corpora available to us. This work was supported in part by the Scientific and Technical Research Council of Turkey (TÜBİTAK Project 112E277).

References

B. Ambridge and E.V.M. Lieven, 2011. *Child Language Acquisition: Contrasting Theoretical Approaches*, chapter 6.1. Cambridge University Press.

D. Arthur and S. Vassilvitskii. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics.

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

Phil Blunsom and Trevor Cohn. 2011. A hierarchical pitman-yor process hmm for unsupervised part of speech induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 865–874, Portland, Oregon, USA, June. Association for Computational Linguistics.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.*, 18:467–479, December.

Sabine Buchholz and Erwin Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 149–164, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2010. Two decades of unsupervised pos induction: how far have we come? In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 575–584, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2011. A bayesian mixture model for pos induction using multiple features. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 638–647, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics - Volume 1*, EACL '03, pages 59–66, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of AKRR'05, International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 106–113, Espoo, Finland, June.

Tomaž Erjavec. 2004. MULTEXT-east version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In *Fourth International Conference on Language Resources and Evaluation, LREC'04*, pages 1535–1538. ELRA.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pages 47–54.

D. Freudenthal, J.M. Pine, and F. Gobet. 2005. On the resolution of ambiguities in the extraction of syntactic categories through chunking. *Cognitive Systems Research*, 6(1):17–25.

[10] https://www.sketchengine.co.uk

Kuzman Ganchev, João Graça, Jennifer Gillenwater, and Ben Taskar. 2010. Posterior regularization for structured latent variable models. *J. Mach. Learn. Res.*, 99:2001–2049, August.

Jianfeng Gao and Mark Johnson. 2008. A comparison of bayesian estimators for unsupervised hidden markov model pos taggers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 344–352, Stroudsburg, PA, USA. Association for Computational Linguistics.

Amir Globerson, Gal Chechik, Fernando Pereira, and Naftali Tishby. 2007. Euclidean embedding of co-occurrence data. *J. Mach. Learn. Res.*, 8:2265–2295, December.

Sharon Goldwater and Tom Griffiths. 2007. A fully bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 744–751, Prague, Czech Republic, June. Association for Computational Linguistics.

Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 320–327, Stroudsburg, PA, USA. Association for Computational Linguistics.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlỳ, and Vít Suchomel. 2013. The tenten corpus family. In *International Conference on Corpus Linguistics, Lancaster*.

Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 296–305, Prague, Czech Republic, June. Association for Computational Linguistics.

Michael Lamar, Yariv Maron, and Elie Bienenstock. 2010a. Latent-descriptor clustering for unsupervised pos induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 799–809, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Lamar, Yariv Maron, Mark Johnson, and Elie Bienenstock. 2010b. Svd and clustering for unsupervised pos tagging. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 215–219, Uppsala, Sweden, July. Association for Computational Linguistics.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2010. Simple type-level unsupervised pos tagging. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 853–861, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3. Linguistic Data Consortium, Philadelphia.

Yariv Maron, Michael Lamar, and Elie Bienenstock. 2010. Sphere embedding: An application to part-of-speech induction. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1567–1575.

Elizbar A Nadaraya. 1964. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142.

A. Rosenberg and J. Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420.

H. Schütze and J. Pedersen. 1993. A Vector Model for syntagmatic and paradigmatic relatedness. In *Proceedings of the 9th Annual Conference of the University of Waterloo Centre for the New OED and Text Research*, Oxford, England.

Hinrich Schütze. 1995. Distributional part-of-speech tagging. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, EACL '95, pages 141–148, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Andreas Stolcke. 2002. Srilm-an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286, November.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Geoffrey S Watson. 1964. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 359–372.

Mehmet Ali Yatbaz, Enis Sert, and Deniz Yuret. 2012. Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 940–951, Jeju Island, Korea, July. Association for Computational Linguistics.

Deniz Yuret. 2012. Fastsubs: An efficient and exact procedure for finding the most likely lexical substitutes based on an n-gram language model. *Signal Processing Letters, IEEE*, 19(11):725–728, Nov.

Solving Substitution Ciphers with Combined Language Models

Bradley Hauer **Ryan Hayward** **Grzegorz Kondrak**
Department of Computing Science
University of Alberta
Edmonton, AB, Canada
{bmhauer,hayward,gkondrak}@ualberta.ca

Abstract

We propose a novel approach to deciphering short monoalphabetic ciphers that combines both character-level and word-level language models. We formulate decipherment as tree search, and use Monte Carlo Tree Search (MCTS) as a fast alternative to beam search. Our experiments show a significant improvement over the state of the art on a benchmark suite of short ciphers. Our approach can also handle ciphers without spaces and ciphers with noise, which allows us to explore its applications to unsupervised transliteration and deniable encryption.

1 Introduction

Monoalphabetic substitution is a well-known method of enciphering a *plaintext* by converting it into a *ciphertext* of the same length using a *key*, which is equivalent to a permutation of the alphabet (Figure 1). The method is elegant and easy to use, requiring only the knowledge of a key whose length is no longer than the size of the alphabet. There are over 10^{26} possible 26-letter keys, so brute-force decryption is infeasible. Manual decipherment of substitution ciphers typically starts with frequency analysis, provided that the ciphertext is sufficiently long, followed by various heuristics (Singh, 1999).

In this paper, we investigate the task of automatically solving substitution ciphers. Complete automation of the key discovery process remains an active area of research (Ravi and Knight, 2008; Corlett and Penn, 2010; Nuhn et al., 2013). The task is to recover the plaintext from the ciphertext without the key, given only a corpus representing the language of the plaintext. The key is a 1-1 mapping between plaintext and ciphertext alphabets, which are assumed to be of equal length. Without loss of generality, we assume that both alphabets are composed of the same set of symbols, so that the key is equivalent to a permutation of the alphabet. Accurate and efficient automated decipherment can be applied to other problems, such as optical character recognition (Nagy et al., 1987), decoding web pages that utilize an unknown encoding scheme (Corlett and Penn, 2010), cognate identification (Berg-Kirkpatrick and Klein, 2011), bilingual lexicon induction (Nuhn et al., 2012), machine translation without parallel training data (Ravi and Knight, 2011), and archaeological decipherment of lost languages (Snyder et al., 2010).

Our contribution is a novel approach to the problem that combines both character-level and word-level language models. We formulate decipherment as a tree search problem, and find solutions with beam search, which has previously been applied to decipherment by Nuhn et al. (2013), or Monte Carlo Tree Search (MCTS), an algorithm originally designed for games, which can provide accurate solutions in less time. We compare the speed and accuracy of both approaches. On a benchmark set of variable-length ciphers, we achieve significant improvement in terms of accuracy over the state of the art. Additional experiments demonstrate that our approach is robust with respect to the lack of word boundaries and the presence of noise. In particular, we use it to recover transliteration mappings between different scripts without parallel data, and to solve the *Gold Bug* riddle, a classic example of a substitution cipher. Finally, we investigate the feasibility of deniable encryption with monoalphabetic substitution ciphers.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2314–2325, Dublin, Ireland, August 23-29 2014.

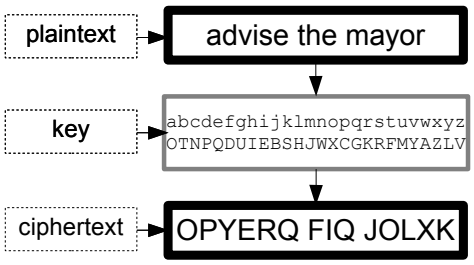


Figure 1: An example of encryption with a substitution cipher.

The paper is organized as follows. After reviewing previous work on automated decipherment in Section 2, we describe our approach to combining character-level and word-level language models with respect to key scoring (Section 3), and key generation (Section 4). In Section 5, we introduce Monte Carlo Tree Search and its adaptation to decipherment. In Section 6, we discuss several evaluation experiments and their results. Section 7 is devoted to experiments in deniable encryption.

2 Related Work

Kevin Knight has been the leading proponent of attacking decipherment problems with NLP techniques, as well as framing NLP problems as decipherment. Knight and Yamada (1999) introduce the topic to the NLP community by demonstrating how to decode unfamiliar writing scripts using phonetic models of known languages. Knight et al. (2006) explore unsupervised learning methods, including the expectation-maximization (EM) algorithm, for a variety of decipherment problems. Ravi and Knight (2009) formulate the problem of unsupervised transliteration as decipherment in order to reconstruct cross-lingual phoneme mapping tables, achieving approximately 50% character accuracy on U.S. names written in the Japanese Katakana script. Reddy and Knight (2011) apply various computational techniques to analyze an undeciphered medieval document. Knight et al. (2011) relate a successful decipherment of a nineteenth-century cipher, which was achieved by combining both manual and computational techniques.

In the remainder of this section, we focus on the work specifically aimed at solving monoalphabetic substitution ciphers. Olson (2007) presents a method that improves on previous dictionary-based approaches by employing an array of selection heuristics. The solver attempts to match ciphertext words against a word list, producing candidate solutions which are then ranked by "trigram probabilities". It is unclear how these probabilities are computed, but the resulting language model seems deficient. For example, given a ciphertext for plaintext *"it was a bright cold day in april*" (the opening of George Orwell's novel *Nineteen Eighty-Four*), the solver[1] produces *"us far a youngs with had up about*". Our new approach, which employs word-level language models, correctly solves this cipher.

Ravi and Knight (2008) formulate decipherment as an integer programming problem in which the objective function is defined by a low-order character language model; an integer program solver then finds the solution that is optimal with respect to the objective function. This method is slow, precluding the use of higher order language models. Our reimplementation of their 2-gram solver deciphers *"it was a bright cold day in april*" as *"ae cor o blathe wind dof as oulan*". By contrast, our approach incorporates word-level information and so tends to avoid out-of-vocabulary words.

Norvig (2009) describes a hill-climbing method that involves both word and character language models, but the models are only loosely combined; specifically, the word model is used to select the best solution from a small number of candidates identified by the character model. When applied to the cipher that corresponds to our example sentence from Orwell, the solver[2] returns *"ache red tab scoville magenta i*".

[1] `http://www.blisstonia.com/software/Decrypto` (accessed August 1, 2013)
[2] `http://norvig.com/ngrams` (accessed June 2, 2014)

Corlett and Penn (2010) use fast heuristic A* search, which can handle much longer ciphers than the method of Ravi and Knight (2008), while still finding the optimal solution. The authors report results only on ciphers of at least 6000 characters, which are much easier to break than short ciphers. The ability to break shorter ciphers implies the ability to break longer ones, but the converse is not true. Our approach achieves a near-zero error rate for ciphers as short as 128 characters.

Nuhn et al. (2013) set the state of the art by employing beam search to solve substitution ciphers. Their method is inexact but fast, allowing them to incorporate higher-order (up to 6-gram) character language models. Our work differs in incorporating word-level information for the generation and scoring of candidate keys, which improves decipherment accuracy.

3 Key Scoring

Previous work tend to employ either character-level language models or dictionary-type word lists. However, word-level language models have a potential of improving the accuracy and speed of decipherment. The information gained from word n-gram frequency is often implicitly used in manual decipherment. For example, a 150-year old cipher of Edgar Allan Poe was solved only after three-letter ciphertext words were replaced with high-frequency unigrams *the*, *and*, and *not*.[3] Similarly, a skilled cryptographer might guess that a repeated 'XQ YWZ' sequence deciphers as the high-frequency bigram *"of the*". We incorporate this insight into our candidate key scoring function.

On the other hand, our character-level language model helps guide the initial stages of the search process, when few or no words are discernible, towards English-like letter sequences. In addition, if the plaintext contains out-of-vocabulary (OOV) words, which do not occur in the training corpus, the character model will favor pronounceable letter sequences. For example, having identified most of the words in plaintext *"village of XeYoviY and burned it*", our solver selects *pecovic* as the highest scoring word that fits the pattern, which in fact is the correct solution.

In order to assign a score to a candidate key, we apply the key to the ciphertext, and compute the probability of the resulting letter sequence using a combined language model that incorporates both character-level and word-level information. With unigram, bigram, and trigram language models over both words and characters trained on a large corpus, n-gram models of different orders are combined by deleted interpolation (Jelinek and Mercer, 1980). The smoothed word trigram probability $\hat{P}$ is:

$$\hat{P}(w_k|w_{k-2}w_{k-1}) = \lambda_1 P(w_k) + \lambda_2 P(w_k|w_{k-1}) + \lambda_3 P(w_k|w_{k-2}w_{k-1}),$$

such that the λs sum to 1. The linear coefficients are determined by successively deleting each trigram from the training corpus and maximizing the likelihood of the rest of the corpus (Brants, 2000). The probability of text $s = w_1, w_2, \ldots, w_n$ according to the smoothed word language model is:

$$P_W(s) = P(w_1^n) = \prod_{k=1}^{n} \hat{P}(w_k|w_{k-2}w_{k-1}).$$

The unigram, bigram, and trigram character language models are combined in a similar manner to yield $P_C(s)$. The final score is then computed as a linear combination of the log probabilities returned by both character and word components:

$$\text{score}(s) = \chi \log P_C(s) + (1 - \chi) \log P_W(s),$$

with the value of χ optimized on a development set. The score of a key is taken to be the score of the decipherment that it produces.

The handling of the OOV words is an important feature of the key scoring algorithm. An incomplete decipherment typically contains many OOV words, which according to the above equations would result in probability $P_W(s)$ being zero. In order to avoid this problem, we replace all OOV words in a decipherment with a special UNKNOWN token for the computation of $P_W(s)$. Prior to deriving the word language models, a sentence consisting of a single UNKNOWN token is appended to the training corpus. As a result, word n-grams that include an UNKNOWN token are assigned very low probability, encouraging the solver to favor decipherments containing fewer OOV words.

[3] http://www.newswise.com/articles/edgar-allen-poe-cipher-solved

4 Key Mutation

The process of generating candidate keys can be viewed as constructing a search tree, where a modified key is represented as a child of an earlier key. The root of the tree contains the initial key, which is generated according to simple frequency analysis (i.e., by mapping the nth most common ciphertext character to the nth most common character in the training corpus). We repeatedly spawn new tree leaves by modifying the keys of current leaves, while ensuring that each node in the tree has a unique key. The fitness of each new key is evaluated by scoring the resulting decipherment, as described in Section 3. At the end of computation, we return the key with the highest score as the solution.

There are an exponential number of possible keys, so it is important to generate new keys that are likely to achieve a higher score than the current key. We exploit this observation: any word n-gram can be represented as a pattern, or sequence, of repeated letters (Table 1). We identify the pattern represented by each word n-gram in the ciphertext, and find a set of *pattern-equivalent* n-grams from the training corpus. For each such n-gram, we generate a corresponding new key from the current key by performing a sequence of transpositions.

Pattern	p-equivalent n-grams
ABCD	said, from, have
ABCC	will, jazz, tree
ABCA	that, says, high
ABCD EFG	from you, said the
ABCA ABD	that the, says sam
ABC DEEFGBCHICG	the bookshelves

Table 1: Examples of pattern-equivalent n-grams.

Pattern-equivalence (abbreviated as p-equivalence) induces an equivalence relation between n-grams (Moore et al., 1999). Formally, two n-grams u and v are p-equivalent ($u \stackrel{p}{=} v$) if and only if they satisfy the following three conditions, where ␣ stands for the space character:

1. $|u| = |v|$
2. $\forall i$: $u_i = \text{␣} \Leftrightarrow v_i = \text{␣}$
3. $\forall i, j$: $u_i = u_j \Leftrightarrow v_i = v_j$

For example, consider ciphertext ‘ZXCZ ZXV’. Adopting *“that”*, which is p-equivalent to ‘ZXCZ’, as a temporary decipherment of the first word, we generate a new key in which Z maps to *t*, X to *h*, and C to *a*. This is accomplished by three letter-pair transpositions in the parent key, producing a child key where ‘ZXCZ’ deciphers to *“that”*. Further keys are generated by matching ‘ZXCZ’ to other p-equivalent words, such as *“says”* and *“high”*. The process is repeated for the second word ‘ZXV’, and then for the entire bigram ‘ZXCZ ZXV’. Each such match induces a series of transpositions resulting in a new key. Leaf expansion is summarized in Figure 3.

In order to avoid spending too much time expanding a single node, we limit the number of replacements for each n-gram in the current decipherment to the k most promising candidates, where k is a parameter optimized on a development set. Note that n-grams excluded in this way may still be included as part of a higher-order n-gram. For example, if the word *birddog* is omitted in favor of more promising candidates, it might be considered as a part of the bigram *struggling birddog*.

Two distinct modes of ranking the candidate n-grams are used throughout the solving process. In the initial stage, n-grams are ranked according to the score computed using the method described in Section 3. Thus, the potential replacements for a given ciphertext n-gram are the highest scoring p-equivalent n-grams from the training corpus regardless of the form of the decipherment implied by the current key. Afterwards, candidates are ranked according to their Hamming distance to the current decipherment, with score used only to break ties. This two-stage approach is designed to exploit the fact that the solver typically gets closer to the correct decipherment as the search progresses.

```
1: Root contains InitialKey
2: for m iterations do
3:     recursively select optimal Path from Root
4:     Leaf = last node of Path
5:     BestLeaf = EXPAND(Leaf, CipherText)
6:     append BestLeaf to Path
7:     Max = Path node with the highest score
8:     assign score of Max to all nodes in Path
```

Figure 2: MCTS for decipherment.

5 Tree Search

Nuhn and Ney (2013) show that finding the optimal decipherment with respect to a character bigram model is NP-hard. Since our scoring function incorporates a language model score, choosing an appropriate tree search technique is crucial in order to minimize the number of search errors, where the score of the returned solution is lower than the score of the actual plaintext. In this section we describe two search algorithms: an adaptation of Monte Carlo Tree Search (MCTS), and a version of beam search.

5.1 Monte Carlo Tree Search

MCTS is a search algorithm for heuristic decision making. Starting from an initial state that acts as the root node, MCTS repeats these four steps: (1) **selection** – starting from the root, recursively pick a child until a leaf is reached; (2) **expansion** – add a set of child nodes to the leaf; (3) **simulation** – simulate the evaluation of the leaf node state; (4) **backpropagation** – recursively ascend to the root, updating the simulation result at all nodes on this path. This process continues until a state is found which passes a success threshold, or time runs out.

Previous work with MCTS has focused on board games, including Hex (Arneson et al., 2010) and Go (Enzenberger et al., 2010), but it has also been employed for problems unrelated to game playing (Previti et al., 2011). Although originally designed for two-player games, MCTS has also been applied to single-agent search (Browne et al., 2012). Inspired by such single-agent MCTS methods (Schadd et al., 2008; Matsumoto et al., 2010; Méhat and Cazenave, 2010), we frame decipherment as a single-player game with a large branching factor, in which the simulation step is replaced with a heuristic scoring function. Since we have no way of verifying that the current decipherment is correct, we stop after performing m iterations. The value of m is determined on a development set.

The function commonly used for comparing nodes in the tree is the upper-confidence bound (UCB) formula for single-player MCTS (Kocsis and Szepesvári, 2006). The formula augments our scoring function from Section 3 with an additional term:

$$UCB(n) = \text{score}(n) + C\sqrt{\frac{ln(v(p(n)))}{v(n)}}$$

where $p(n)$ is the parent of node n, and $v(n)$ is the number of times that n has been visited. The second term favors nodes that have been visited relatively infrequently in comparison with their parents. The value of C is set on a development set.

Figure 2 summarizes our implementation. Each iteration begins by finding a path through the tree that is currently optimal according to the UCB. The path begins at the root, includes a locally optimal child at each level, and ends with a leaf. The leaf is expanded using the function EXPAND shown in Figure 3. The highest-scoring of the generated children is then appended to the optimal path. If the score of the new leaf (*not* the UCB) is higher than the score of its parent, we backpropagate that score to all nodes along the path leading from the root. This encourages further exploration along all or part of this path.

5.2 Beam Search

Beam search is a tree search algorithm that uses a size-limited list of nodes currently under consideration, which is referred to as the beam. If the beam is full, a new node can be added to it only if it has a higher

```
1: function EXPAND(Leaf, CipherText)
2:     for all word n-grams w in CipherText do
3:         for k best w' s.t. w' ≡^p w do
4:             NewLeaf = Modify(Leaf, w ↦ w')
5:             if NewLeaf not in the tree then
6:                 add NewLeaf as a child of Leaf
7:             if score(NewLeaf) > score(BestLeaf) then
8:                 BestLeaf = NewLeaf
9:     return BestLeaf
```

Figure 3: Leaf expansion.

score than at least one node currently in the beam. In such a case, the lowest-scoring node is removed from the beam and any further consideration.

Nuhn et al. (2013) use beam search for decipherment in their character-based approach. Starting from an empty root node, a partial key is extended by one character in each iteration, so that each level of the search tree corresponds to a unique ciphertext symbol. The search ends when the key covers the entire ciphertext.

By contrast, we apply beam search at the word n-gram level. The EXPAND subroutine defined in Figure 3 is repeatedly invoked for a specified number of iterations (a tunable parameter). In each iteration, the algorithm analyzes a set of word n-gram substitutions, which may involve multiple characters, as described in Section 4. The search stops early if the beam becomes empty. On short ciphers (32 characters or less), the best solution is typically found within the first five iterations, but this can only be confirmed after the search process is completed.

6 Experiments

In order to evaluate our approach and compare it to previous work, we conducted several experiments. We created three test sets of variable-length ciphers: (1) with spaces, (2) without spaces, and (3) with spaces and added encipherment noise. In addition, we tested our system on Serbian Cyrillic, and the Gold Bug cipher.

We derive our English language models from a subset of the New York Times corpus (LDC2003T05) containing 17M words. From the same subset, we obtain letter-frequency statistics, as well as the lists of p-equivalent n-grams. For comparison, Ravi and Knight (2008) use 50M words, while Nuhn et al. (2013) state that they train on a subset of the Gigaword corpus without specifying its size.

6.1 Substitution Ciphers

Following Ravi and Knight (2008) and Nuhn et al. (2013), we test our approach on a benchmark set of ciphers of lengths, 2, 4, 8, ..., 256, where each length is represented by 50 ciphers. The plaintexts are randomly extracted from the Wikipedia article on *History*, which is quite different from our NYT training corpus. Spaces are preserved, and the boundaries of the ciphers match word boundaries.

Figure 4 shows the decipherment error rate of the beam-search version of our algorithm vs. the published results of the best-performing variants of Ravi and Knight (2008) and Nuhn et al. (2013): letter 3-gram and 6-gram, respectively. The decipherment error rate is defined as the ratio of the number of incorrectly deciphered characters to the length of the plaintext. Our approach achieves a statistically significant improvement on ciphers of length 8 and 16. Shorter ciphers are inherently hard to solve, while the error rates on longer ciphers are close to zero. Unfortunately, Nuhn et al. (2013) only provide a graph of their error rates, which in some cases prevents us from confirming the statistical significance of the improvements (c.f. Table 2).

Examples of decipherment errors are shown in Table 3. As can be seen, the proposed plaintexts are often perfectly reasonable given the cipher letter pattern. The solutions proposed for very short ciphers are usually high-frequency words; for example, the 2-letter ciphers matching the pattern 'AB'

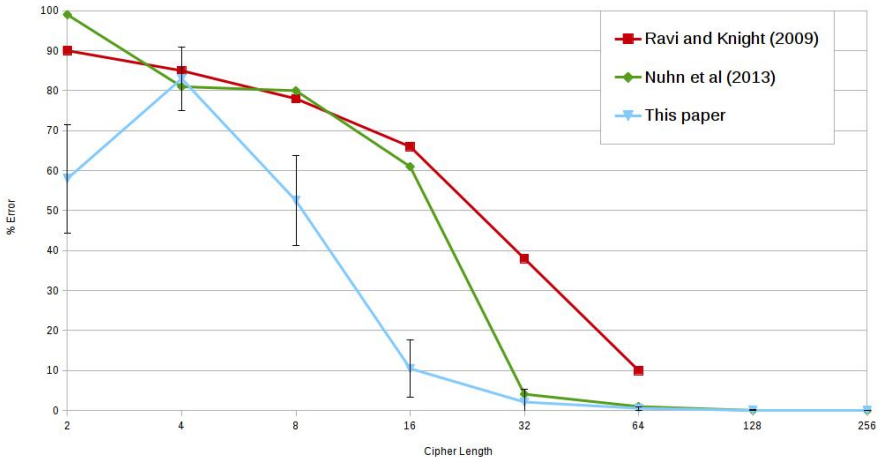


Figure 4: Average decipherment error rate as a function of cipher length on the Wikipedia test set.

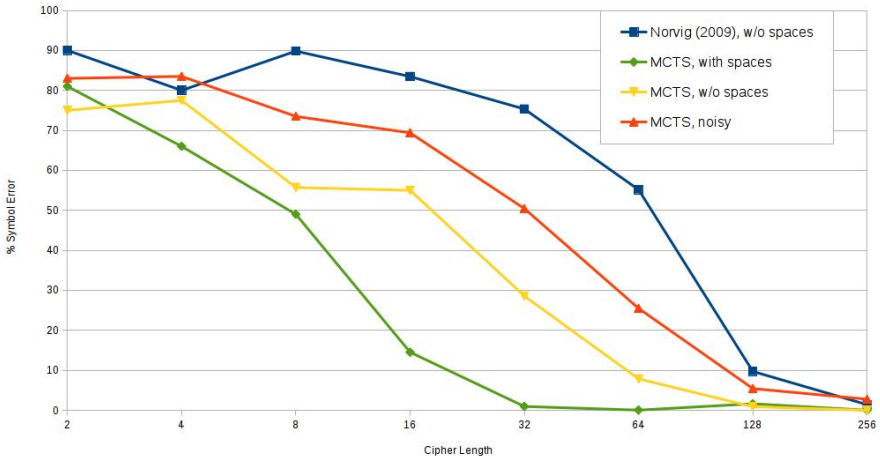


Figure 5: Average decipherment error rate as a function of cipher length on the NYT test set.

	Wikipedia			NYT				
	with spaces			with spaces		no spaces	noisy	
	Beam	MCTS	Greedy	Beam	MCTS	MCTS	Beam	MCTS
2	58.00	58.00	58.00	81.00	81.00	75.00	83.00	83.00
4	83.00	83.00	83.00	66.00	66.00	77.50	83.50	83.50
8	52.50	52.50	52.50	49.00	49.00	55.71	73.50	73.50
16	10.50	12.62	18.50	13.50	14.50	55.00	69.75	69.38
32	2.12	6.12	10.88	0.88	0.94	28.57	46.81	50.44
64	0.56	0.72	2.50	0.03	0.03	7.85	16.66	25.47
128	0.14	0.16	0.16	0.00	1.61	0.87	5.20	5.41
256	0.00	0.00	0.10	0.02	0.02	0.00	2.73	2.75

Table 2: Average decipherment error rate of our solver as a function of cipher length on the Wikipedia and the NYT test sets.

Cipher length	Cipher pattern	Actual plaintext	Decipherment
2	AB	to	of
4	ABCD	from	said
4	ABBC	look	been
8	ABCDCEFG	slobodan	original
8	ABCDE FG	filed by	would be
16	ABCCDEE BFG HBCI	jarrett and mark	carroll and part
16	ABCDE FGCHA IJKL	group along with	drugs would make

Table 3: Examples of decipherment errors.

are invariably deciphered as *"of"*. The errors in ciphers of length 32 or more tend to be confined to individual words, which are often OOV names.

6.2 Beam Search vs. MCTS

The error rates of the two versions of our algorithm are very close, with a few exceptions (Table 2). Out of 400 ciphers with spaces in the Wikipedia test set, the MCTS variant correctly solves 260 out of 400 ciphers, compared to 262 when beam search is used. In 9 MCTS solutions and 3 beam search solutions, the score of the proposed decipherment is lower than the score of the actual plaintext, which indicates a search error.

By setting the beam size to one, or the value of C in MCTS to zero, the two search techniques are reduced to *greedy* search. As shown in Table 2, in terms of accuracy, greedy search is worse than MCTS on the lengths of 16, 32, and 64, and roughly equal on other lengths. This suggests that an intelligent search strategy is important for obtaining the best results.

In terms of speed, the MCTS version outperforms beam search, thanks to a smaller number of expanded nodes in the search tree. For example, it takes on average 9 minutes to solve a cipher of length 256, compared to 41 minutes for the beam search version. Direct comparison of the execution times with the previous work is difficult because of variable computing configurations, as well as the unavailability of the implementations. However, on ciphers of the length of 128, our MCTS version takes on average 197 seconds, which is comparable to 152 seconds reported by Nuhn et al. (2013), and faster than our reimplementation of the bigram solver of Ravi and Knight (2008) which takes on average 563 seconds. The trigram solver of Ravi and Knight (2008) is even slower, as evidenced by the fact that they report no corresponding results on ciphers longer than 64 letters.

6.3 Noisy Ciphers

Previous work has generally focused on noise-free ciphers. However, in real-life applications, we may encounter cases of imperfect encipherment, in which some characters are incorrectly mapped. Corlett and Penn (2010) identify the issue of noisy ciphers as a worthwhile future direction. Adding noise also increases a cipher's security, as it alters the pattern of letter repetitions in words. In this section, we evaluate the robustness of our approach in the presence of noise.

In order to quantify the effect of adding noise to ciphers, we randomly corrupt $log_2(n)$ of the ciphertext letters, where n is the length of the cipher. Our results on such ciphers are shown in Table 2. As expected, adding noise to the ciphertexts increases the error rate in comparison with ciphers without noise. However, our algorithm is still able to break most of the ciphers of length 64 and longer, and makes only occasional mistakes on ciphers of length 256. Beam search is substantially better than MCTS only on lengths of 32 and 64. These results indicate that our word-oriented approach is reasonably robust with respect to the presence of noise.

6.4 Croatian and Serbian

We further test the robustness of our approach by performing an experiment on decipherment of an unknown script. For this experiment, we selected Croatian and Serbian, two closely related languages

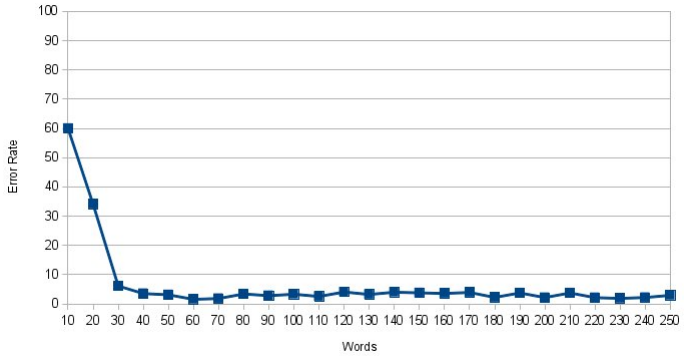


Figure 6: The decipherment error rate on a Serbian sample text as a function of the ciphertext length.

that are written in different scripts (Latin and Cyrillic). The correspondence between the two script alphabets is not exactly one-to-one: Serbian Cyrillic uses 30 symbols, while Croatian Latin uses 27. In particular, the Cyrillic characters љ, њ, and џ are represented in the Latin script as digraphs *lj*, *nj*, and *dž*. In addition, there are differences in lexicon and grammar between the two languages, which make this task a challenging case of noisy encipherment.

In the experiment, we treat a short text in Serbian as enciphered Croatian and attempt to recover the key, which in this case is the mapping between the characters in the two writing scripts. Each letter with a diacritic is considered as different from the same letter with no diacritic. We derive the word and character language models from the Croatian part of the ECI Multilingual Corpus, which contains approximately 720K word tokens. For testing, we use a 250-word, 1583-character sample from the Serbian version of the Universal Declaration of Human Rights.

сва људска бића рађају се слободна и једнака у достојанству и правима она су обдарена разумом и свешћу и треба једни према другима
sva šudska bi**h**a ra**l**aju se sžobodna i jednaka u dostojanstvu i pravima ona su obdarena ra**č**umom i sve**ch**u i treba jedni prema dru**z**ima

Table 4: Serbian Cyrillic deciphered as Croatian. The decipherment errors are shown in boldface.

The decipherment error rate on the Serbian ciphertext drops quickly, leveling at about 3% at the length of 50 words (Figure 6). The residual error rate reflects the lack of correct mapping for the three Serbian letters mentioned above. As can be seen in Table 4, the actual decipherment of a 30-word ciphertext contains only a handful of isolated errors. On the other hand, a pure frequency-based approach fails on this task with a mapping error rate close to 90%.

6.5 Ciphers Without Spaces

Removing spaces that separate words is another way of increasing the security of a cipher. The assumption is that the intended recipient, after applying the key, will still be able to guess the location of word boundaries, and recover the meaning of the message. We are interested in testing our approach on such ciphers, but since it is dependent on word language models, we need to first modify it to identify word boundaries. In particular, the two components that require word boundaries are the scoring function (Section 3), and the search tree node expansion (Section 5).

In order to compute the scoring function, we try to infer word boundaries in the current decipherment using the following simple greedy algorithm. The current decipherment is scanned repeatedly from left to right in search for words of length L, where L gradually decreases from the length of the longest word in the training corpus, down to the minimal value of 2. If a word is found, the process is applied recursively to both remaining parts of the ciphertext. We use a fast greedy search instead of a slower but more accurate dynamic programming approach as this search must be executed each time a key is evaluated.

In the search tree node expansion step, for each substring of length at least 2 in the current decipherment, we attempt to replace it with all pattern-equivalent n-grams (with spaces removed). As a result,

53‡‡†305))6*;4826)4‡.)4‡);806*;48†8¶60))85;1‡(;:‡*8†83(88)5*†;46(;88*96*?;8)*‡(;485);5*†2:*‡(;4956*2(5*-4)8¶8*;4069285);)6†8)4

agoodglassinthebishopshostelinthedevilsseatfortyonedegreesandthirteenminutesnortheastandbynorthmainbranchseventhlimbeastsidesh

Table 5: The beginning of the Gold Bug cipher and its decipherment.

each key spawns a large number of children, increasing both time and memory usage. Overall, the modified algorithm is as much as a hundred times slower than the original algorithm. However, when MCTS is used as search method, we are still able to perform the decipherment in reasonable time.

For testing, we remove spaces from both the plaintexts and ciphertexts, and reduce the number of ciphers to 10 for each cipher length. Our results, shown in Figure 5, compare favorably to the solver of (Norvig, 2009), which is designed to work on ciphers without spaces.

The final test of our decipherment algorithm is the cipher from *The Gold Bug* by Edgar Alan Poe. In that story, the 204-character cipher gives the location of hidden treasure. Our implementation finds a completely correct solution, the beginning of which is shown in Table 5. Both experiments reported in this section confirm that our word-based approach works well even when spaces are removed from ciphers.

7 Deniable Encryption

In one of Stanisław Lem's novels, military cryptographers encipher messages in such a way that the ciphertext appears to be plain text (Lem, 1973). Canetti et al. (1997) investigate a related idea, in which the ciphertext "looks like" an encryption of a plaintext that is different from the real message. In the context of monoalphabetic substitution ciphers, we define the task as follows: given a message, find an encipherment key yielding a ciphertext that resembles natural language text. For example, *"game with planes*" is a deniable encryption of the message *"take your places*" (the two texts are p-equivalent).

We applied our solver to a set of sentences from the text of *Nineteen Eighty-Four*, treating each sentence as a ciphertext. In order to ensure that the alternative plaintexts are distinct from the original sentences, we modified our solver to disregard candidate keys that yield a solution containing a content word from the input. For example, *"fine hours*" was not deemed an acceptable deniable encryption of *"five hours*". With this condition added, alternative plaintexts were produced for all 6531 sentences. Of these, 1464 (22.4%) were determined to be composed entirely of words seen in training. However, most of these deniable encryptions were either non-grammatical or differed only slightly from the actual plaintexts. It appears that substitution ciphers that preserve spaces fail to offer sufficient flexibility for finding deniable encryptions.

In the second experiment, we applied our solver to a subset of 757 original sentences of length 32 or less, with spaces removed. The lack of spaces allows for more flexibility in finding deniable encryptions. For example, the program finds *"draft a compromise*" as a deniable encryption of *"zeal was not enough*". None of the produced texts contained out-of-vocabulary words, but most were still ungrammatical or nonsensical. Allowing for some noise to be introduced into the one-to-one letter mapping would likely result in more acceptable deniable encryptions, but our current implementation can handle noise only on the input side.

8 Conclusion

We have presented a novel approach to the decipherment of monoalphabetic substitution ciphers that combines character and word-level language models. We have proposed Monte Carlo Tree Search as a fast alternative to beam search on the decipherment task. Our experiments demonstrate significant improvement over the current state of the art. Additional experiments show that our approach is robust in handling ciphers without spaces, and ciphers with noise, including the practical application of recovering transliteration mappings between Serbian and Croatian.

In the future, we would like to extend our approach to handle homophonic ciphers, in which the one-to-one mapping restriction is relaxed. Another interesting direction is developing algorithms to generate syntactically correct and meaningful deniable encryptions.

Acknowledgements

This research was supported by the Natural Sciences and Engineering Research Council of Canada and the Alberta Innovates Technology Futures.

References

Broderick Arneson, Ryan B Hayward, and Philip Henderson. 2010. Monte Carlo Tree Search in Hex. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(4):251–258.

Taylor Berg-Kirkpatrick and Dan Klein. 2011. Simple effective decipherment via combinatorial optimization. In *Empirical Methods in Natural Language Processing*, pages 313–321.

Thorsten Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43.

Ran Canetti, Cynthia Dwork, Moni Naor, and Rafi Ostrovsky. 1997. Deniable encryption. In *Advances in Cryptology–CRYPTO'97*, pages 90–104.

Eric Corlett and Gerald Penn. 2010. An exact A* method for deciphering letter-substitution ciphers. In *the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1040–1047.

Markus Enzenberger, Martin Muller, Broderick Arneson, and Richard Segal. 2010. Fuego – an open-source framework for board games and go engine based on Monte Carlo tree search. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(4):259–270.

F. Jelinek and R.L. Mercer. 1980. Interpolated estimation of Markov source parameters from sparse data. *Pattern recognition in practice*.

Kevin Knight and Kenji Yamada. 1999. A computational approach to deciphering unknown scripts. In *ACL Workshop on Unsupervised Learning in Natural Language Processing*, pages 37–44.

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *the COLING/ACL 2006 Main Conference Poster Sessions*, pages 499–506.

Kevin Knight, Beáta Megyesi, and Christiane Schaefer. 2011. The Copiale cipher. In *the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 2–9.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based Monte-Carlo Planning. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Euro. Conf. Mach. Learn.*, pages 282–293, Berlin, Germany. Springer.

Stanisław Lem. 1973. *Memoirs found in a bathtub*. The Seabury Press.

Shimpei Matsumoto, Noriaki Hirosue, Kyohei Itonaga, Kazuma Yokoo, and Hisatomo Futahashi. 2010. Evaluation of simulation strategy on single-player Monte-Carlo tree search and its discussion for a practical scheduling problem. In *the International MultiConference of Engineers and Computer Scientists*, volume 3, pages 2086–2091.

Jean Méhat and Tristan Cazenave. 2010. Combining UCT and nested Monte Carlo search for single-player general game playing. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(4):271–277.

Dennis Moore, W.F. Smyth, and Dianne Miller. 1999. Counting distinct strings. *Algorithmica*, 23(1):1–13.

George Nagy, Sharad Seth, and Kent Einspahr. 1987. Decoding substitution ciphers by means of word matching with application to ocr. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):710–715.

Peter Norvig. 2009. Natural language corpus data. In Toby Segaran and Jeff Hammerbacher, editors, *Beautiful data: the stories behind elegant data solutions*. O'Reilly.

Malte Nuhn and Hermann Ney. 2013. Decipherment complexity in 1:1 substitution ciphers. In *the 51st Annual Meeting of the Association for Computational Linguistics*, pages 615–621.

Malte Nuhn, Arne Mauser, and Hermann Ney. 2012. Deciphering foreign language by combining language models and context vectors. In *the 50th Annual Meeting of the Association for Computational Linguistics*, pages 156–164.

Malte Nuhn, Julian Schamper, and Hermann Ney. 2013. Beam search for solving substitution ciphers. In *the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1568–1576.

Edwin Olson. 2007. Robust dictionary attack of short simple substitution ciphers. *Cryptologia*, 31(4):332–342.

Alessandro Previti, Raghuram Ramanujan, Marco Schaerf, and Bart Selman. 2011. Applying UCT to Boolean satisfiability. In *Theory and Applications of Satisfiability Testing-SAT 2011*, pages 373–374. Springer.

Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Empirical Methods in Natural Language Processing*, pages 812–819.

Sujith Ravi and Kevin Knight. 2009. Learning phoneme mappings for transliteration without parallel data. In *NAACL*, pages 37–45.

Sujith Ravi and Kevin Knight. 2011. Deciphering foreign language. In *the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 12–21.

Sravana Reddy and Kevin Knight. 2011. What we know about the Voynich manuscript. In *the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 78–86.

Maarten PD Schadd, Mark HM Winands, H Jaap Van Den Herik, Guillaume MJ-B Chaslot, and Jos WHM Uiterwijk. 2008. Single-player Monte-Carlo tree search. In *Computers and Games*, pages 1–12. Springer.

Simon Singh. 1999. *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*. Random House.

Benjamin Snyder, Regina Barzilay, and Kevin Knight. 2010. A statistical model for lost language decipherment. In *the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1048–1057.

Unsupervised Word Segmentation in Context

Gabriel Synnaeve and **Isabelle Dautriche**
LSCP, DEC
ENS Ulm, Paris, France
gabriel.synnaeve@gmail.com
isabelle.dautriche@gmail.com

Benjamin Börschinger
Institut für Computerlinguistik
Universität Heidelberg, Heidelberg, Germany.
benjamin.boerschinger@gmail.com

Mark Johnson
Department of Computer Science
Macquarie University, Sydney, Australia
mark.johnson@mq.edu.au

Emmanuel Dupoux
LSCP, DEC
EHESS, Paris, France
emmanuel.dupoux@gmail.com

Abstract

This paper extends existing word segmentation models to take non-linguistic context into account. It improves the token F-score of a top performing segmentation models by 2.5% on a 27k utterances dataset. We posit that word segmentation is easier in-context because the learner is not trying to access irrelevant lexical items. We use topics from a Latent Dirichlet Allocation model as a proxy for "activities" contexts, to label the *Providence* corpus. We present Adaptor Grammar models that use these context labels, and we study their performance with and without context annotations at test time.

1 Introduction and Previous Works

Segmentation of the speech stream into lexical units plays a central role in early language acquisition. Because words are generally not uttered in isolation, one of the first task for infants learning a language is to extract the words that make up the utterances they hear. Experimental research has shown that infants are able to segment fluent speech into word-like units within the first year of life (Jusczyk and Aslin, 1995). How does this ability emerge? There is evidence that infants use a broad array of linguistic cues to perform word segmentation (e.g., phonotactics (Jusczyk et al., 1993a), prosodic information (Jusczyk et al., 1993b), statistical regularities (Saffran et al., 1996)). Past experimental and modeling research on speech segmentation has mainly focused on linguistic cues, treating them as independent from other non-linguistic cues naturally occurring in the child learning environment. Yet, language appears in context and is constrained by the events occurring in the daily life of the child. For example, during an eating event one is most likely to speak about food, while during a zoo-visit event, people are more likely to talk about the animals they see. Activity contexts may provide a natural structure to speech that would be readily be accessible to children. A recent study using dense recordings of a single child's language development (Roy et al., 2006) showed that words appearing in specific activity contexts are learned faster (Roy et al., 2012). Relatedly, Johnson et al. (2010) showed that Adaptor Grammars (AGs) performed better on a segmentation task when the model has access to a hand-annotated set of objects present in the environment, that it can use to learn simultaneously word-object associations (see also (Frank et al., 2009)). This supports the view that integrating multiple sources of information, linguistic and non-linguistic, can improve learning.

Following this idea, we posit that information from the broader context in which a word has been uttered may simplify the learning problem faced by the child. In particular, our hypothesis postulates that speech segmentation is easier when using vocabularies that are related to a specific activity (eating,

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2326–2334, Dublin, Ireland, August 23-29 2014.

Table 1: Most probable words in the 7 final topics

egg	book	ball	truck	name	color	block
apple	shape	cat	car	school	bear	battery
banana	square	hat	fire	time	crayon	minute
milk	circle	tree	piece	today	hair	phone
butter	triangle	fish	train	day	head	puzzle
≈food	≈shapes	≈playing	≈toys	≈time	≈drawing	"garbage"

playing...), or place (kitchen, bedroom...). To evaluate this hypothesis, we applied topic modeling (Blei et al., 2003) to automatically derive activity contexts on a corpus of child directed speech, the *Providence* corpus (Demuth et al., 2006), and tested the influence of such topics on a word segmentation task extending the AG models used in (Börschinger et al., 2012). We found that a model augmented with the assumption that words are dependent upon the topic of the discourse (as a proxy for activity context) performs better than the same model without access to the discourse topic. This suggests that the broader context in which sentences are uttered may help in the word segmentation process, and could presumably be used at various stages of language development.

The paper is structured as follows. Section 2 presents a novel approach to augment a corpus with contextual annotations derived from topic models. Section 3 quickly explains Adaptor Grammars, the framework that we used to express all our models. Section 4 presents all the models that were used in the results. Section 5 describes the Providence corpus and the experimental setup. Section 6 shows our quantitative and qualitative results. Finally, we discuss the implications for models of language learning.

2 Topics as Proxies for Contexts

Roy et al. (2012) found high correlations between human-annotated activity contexts and topics from a latent Dirichlet allocation model (LDA) (Blei et al., 2003), thus showing that using topics as proxies for contexts is a sound approach. Topic modeling infers a topic distribution for each "document" (a bag of words) in the corpus. Since "documents" were not annotated in our corpus, we developed the following 3-step approach to automatically segment it into documents.

Firstly, for *all* the children of the Providence corpus, we used recording sessions as hard document boundaries. We considered as a "possible document" every contiguous sequence of sentences separated by at least 10 seconds of silence, according to the orthographic transcript. We also identified "possible documents" using cues such as "bye/hi", indicating a change of participants. This segmentation resulted in an over-segmented corpus (compared to context switches), yielding a total of $16,742$ documents.

Secondly, we used the *gensim* software (Řehůřek and Sojka, 2010) to train a topic model (LDA)[1], and get the topic distributions for each of these documents. We used the symmetric KL-divergence to measure the distance between two topic distributions before and after a "possible document" boundary. If the distance was above a threshold, we considered this boundary as a document boundary. Otherwise we merged both "possible documents" through this silence. The threshold was set empirically to discriminate between two topic distributions that correspond to different activity contexts. After this step, we assume that each of the resulting $8,634$ documents maps to an activity context.

Thirdly, we applied LDA again on this new segmentation to get the topic distribution, hence the activity context, of each document. The number of topics is qualitatively chosen to correspond to the number of main activity contexts (eating / playing / drawing / etc.) that occur in the Providence dataset (we used 7 topics), the resulting most topic specific words are shown in Table 1. Finally, for each document, we got a distribution on topics, and we annotated the document with the most probable topic. By doing that, we throw away graded information about the distribution on topics for each document. We could make use of the full distribution, but here we are only interested in the most probable topic as a proxy for activity context. We do not posit that the infants learn the topic models on linguistic cues while bootstrapping speech and segmentation, but rather that they get activity context from non-linguistic cues.

[1]We did LDA only on nouns (as they contain most of the semantics), weighted by TF-IDF.

3 Adaptor Grammars

Adaptor Grammars (Johnson et al., 2007) are an extension of probabilistic context-free grammars (PCFGs) that learn probability of entire subtrees as well as probabilities of rules. A PCFG (N, W, R, S, θ) consists of a start symbol S, N and W disjoints sets of nonterminals and terminal symbols respectively. R is a set of rules producing elements of N or W. Finally, θ is a set of distributions over the rules $R_X, \forall X \in N$ (R_X are the rules that expand X). An AG $(N, W, R, S, \theta, A, C)$ extends the above PCFG with a subset ($A \subseteq N$) of adapted nonterminals, each of them ($X \in A$) having an associated adaptor ($C_X \in C$). An AG defines a distribution over trees $G_X, \forall X \in N \cup W$. If $X \notin A$, then G_X is defined exactly as for a PCFG:

$$G_X = \sum_{\substack{X \to Y_1 \dots Y_n \\ \in R_X}} \theta_{X \to Y_1 \dots Y_n} \mathrm{TD}_X(G_{Y_1} \dots G_{Y_n})$$

With $\mathrm{TD}_X(G_1 \dots G_n)$ the distribution over trees with root node X and each subtree $t_i \sim G_i$ i.i.d. If $X \in A$, then there is an additional indirection (composition) with the distribution H_X:

$$G_X = \sum_{\substack{X \to Y_1 \dots Y_n \\ \in R_X}} \theta_{X \to Y_1 \dots Y_n} \mathrm{TD}_X(H_{Y_1} \dots H_{Y_n})$$

$$H_X \sim C_X(G_X)$$

We used C_X adaptors following the Pitman-Yor process (PYP) (Perman et al., 1992; Teh, 2006) with parameters a and b. The PYP generates (Zipfian) type frequencies that are similar to those that occur in natural language (Goldwater et al., 2011). Metaphorically, if there are n customers and m tables, the $n+1$th customer is assigned to table z_{n+1} according to (δ_k is the Kronecker delta function):

$$z_{n+1} | z_1 \dots z_n \sim \frac{ma + b}{n + b} \delta_{m+1} + \sum_{k=1}^{m} \frac{n_k - a}{n + b} \delta_k$$

For an AG, this means that adapted non-terminals ($X \in A$) either expand to a previously generated subtree ($T(X)_k$) with probability proportional to how often it was visited (n_k), or to a new subtree ($T(X)_{m+1}$) generated through the PCFG with probability proportional to $ma + b$.

4 Word segmentation models

4.1 Unigram model

This most basic model just generates words as sequences of phonemes. As $\underline{Word}$ is underlined, it means it is adapted, and thus we learn a "word unit -like" vocabulary. $Phon$ is a nonterminal that expands to all the phonemes of the language under consideration.

$$Sentence \to Word^+$$
$$\underline{Word} \to Phon^+$$

where :

$$Word^+ \quad \Leftrightarrow \begin{cases} Words \to Word \\ Words \to Word\, Words \end{cases}$$

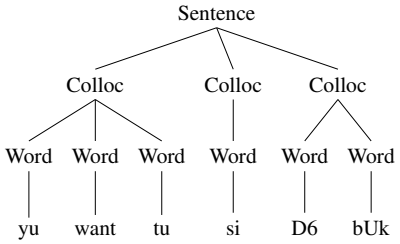
4.2 Collocations and Syllabification

The *baseline* that we are using is commonly called the "colloc-syll" model (Johnson, 2008; Börschinger et al., 2012) and is reported at 78% token F-score on the standard Brent version of the Bernstein-Ratner corpus corpus (Johnson, 2008). It posits that sentences are collocations of words, and words are composed of syllables. (Goldwater et al., 2009) showed how an assumption of independence between words (a unigram model) led to under-segmentation. So, above the $Word$ level, we take the collocations (co-occurring sequences) of words into account.

Furthermore, there is evidence that 8-month-old infants track syllable frequencies (Saffran et al., 1996), and the "colloc-syll" model can take that into account. $Word$ splits into general syllables and initial- or final- specific syllables. Syllables consist of onsets or codas (producing consonants), and nuclei (vowels). Onsets, nuclei and codas are adapted, thus allowing this model to memorize sequences or consonants or sequences of vowels, dependent on their position in the word. Consonants and vowels are the pre-terminals, their derivation is specified in the grammar into phonemes of the language.

$$Sentence \rightarrow Colloc^{+}$$
$$\underline{Colloc} \rightarrow Word^{+}$$
$$\underline{Word} \rightarrow StructSyll$$

For notations purposes, all this syllabification is appended after $Word$ by $\underline{Word} \rightarrow StructSyll$. All details about the collocations and syllabification grammars can be found in (Johnson, 2008). Here is an example of a (good) parse of "`yuwanttusiD6bUk`" with this model, skipping the $StructSyll$ derivations:



4.3 Including topics (contexts)

To allow for the model to make use of the topics (used as proxies for contexts), we modify the grammar by prefixing utterances with topic number (similarly to (Johnson et al., 2010)), $\forall K \in \#topics$:

$$Sentence \rightarrow \_tK\ Colloc^{+}_{tK}$$
$$\underline{Colloc_{tK}} \rightarrow Word^{+}_{tK}$$

For each $\underline{Word_{tK}}$, we can derive it into a common adapted $\underline{Word}$ by $\underline{Word_{tK}} \rightarrow Word$. Consider this lower level adaptor ($\underline{Word}$): it learns a shared vocabulary independently of the topic (all contexts that will derive `b U k` will increment the `Word(b U k)` pseudo-count). This $Word$-hierarchical model is called *share vocab*.

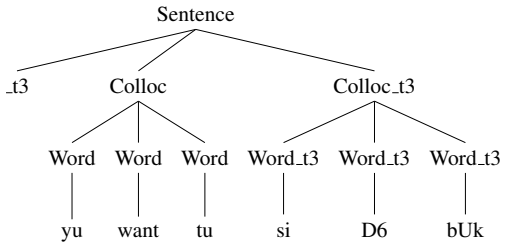
Alternatively, we can learn a separate vocabulary for each topic, by having directly: $\underline{Word_{tK}} \rightarrow StructSyll$ (note that all words then share the same syllabic structure). Words are split across different topics and need to be adapted for each topic in which they appear. This flat structure vocabulary model is called *split vocab*.

4.4 Allowing for non context-specific words

Sentences are not composed only of context-specific words, thus we need a third type of extension that allows for topic-independent and topic-specific words to mix. For this, we add topic-independent types of $Colloc$ and $Word$ that can be used across all topics, but we force each sentence to have at least one topical collocation:

$$\begin{aligned}
Sentence &\rightarrow _tK\ (Colloc^{+}|Colloc^{+}_{tK})\ Colloc^{+}_{tK} \\
&\quad\ (Colloc^{+}|Colloc^{+}_{tK}) \\
\underline{Colloc_{tK}} &\rightarrow Word^{+}_{tK} \\
\underline{Colloc} &\rightarrow Word^{+} \\
\underline{Word_{tK}} &\rightarrow StructSyll \\
\underline{Word} &\rightarrow StructSyll
\end{aligned}$$

Parentheses denote that these terms are optionals, and "|" denotes "or". Both $\underline{Word_{tK}}$ and $\underline{Word}$ are adapted, but this time on the same level of hierarchy. This model allows the use of both topic-specific and common words in sentences, and it learns $\#topics + 1$ vocabularies. We call this model *with common*. An example of a correct parse with this model is given by:



5 Experimental setup

The *Providence* corpus (Demuth et al., 2006) consists of audio and video, weekly or bi-weekly, recordings of 6 monolingual English-speaking children home interactions. Each recording is approximatively 1 hour long. This corpus spans approximatively from their first to third year. We used the whole corpus to extract the topics to get more stable and general activity contexts. For all the following results, we used only the Naima portion between 11 months and 24 months, consisting in 26,425 utterances (sentences) and 135,389 tokens (words). The input consist in DARPABET-encoded sequences of phonemes with about 4200 word-types in the Naima subset. We followed the same preparation procedure as in (Börschinger et al., 2012), where more details about the corpus can be found.

We used the last version of Mark Johnson's Adaptor Grammars software[2]. All the additional code (preparation, topics, grammars, learning) to reproduce these experiments and results is freely available online[3], along with the datasets annotations derived from topic modeling[4]. For the adaptors, we used a $Beta(1, 1)$ (uniform) prior on the PYP a parameter, and a sparse $Gamma(100, 0.01)$ prior on the PYP b parameter. We ran 500 iterations (finishing at $\approx 0.05\%$ of log posterior variation between the lasts iterations) with several runs for each subset of the Naima dataset.

6 Results

6.1 Unsupervised words segmentation

Table 2: Mean (token and boundary) F-scores (f), precisions (p), and recalls (r) for different models depending on the size of dataset (age range).

months	baseline			share vocab			split vocab			with common		
token	f	p	r	f	p	r	f	p	r	f	p	r
11-12	**.80**	.79	.81	.77	.76	.78	.77	.75	.78	.77	.75	.78
11-15	.81	.81	.82	.76	.78	.75	.81	.79	.82	**.82**	.81	.83
11-19	.82	.82	.83	.77	.78	.76	.81	.81	.82	**.83**	.82	.84
11-22	.81	.82	.81	.77	.79	.75	.82	.81	.83	**.83**	.82	.84
boundary	f	p	r	f	p	r	f	p	r	f	p	r
11-12	**.90**	.88	.91	.88	.87	.89	.87	.85	.90	.88	.85	.90
11-15	**.91**	.91	.92	.89	.91	.86	**.91**	.89	.92	**.91**	.90	.93
11-19	**.92**	.92	.93	.90	.92	.88	**.92**	.91	.93	**.92**	.91	.94
11-22	.92	.93	.91	.90	.93	.87	.92	.91	.93	**.93**	.91	.94

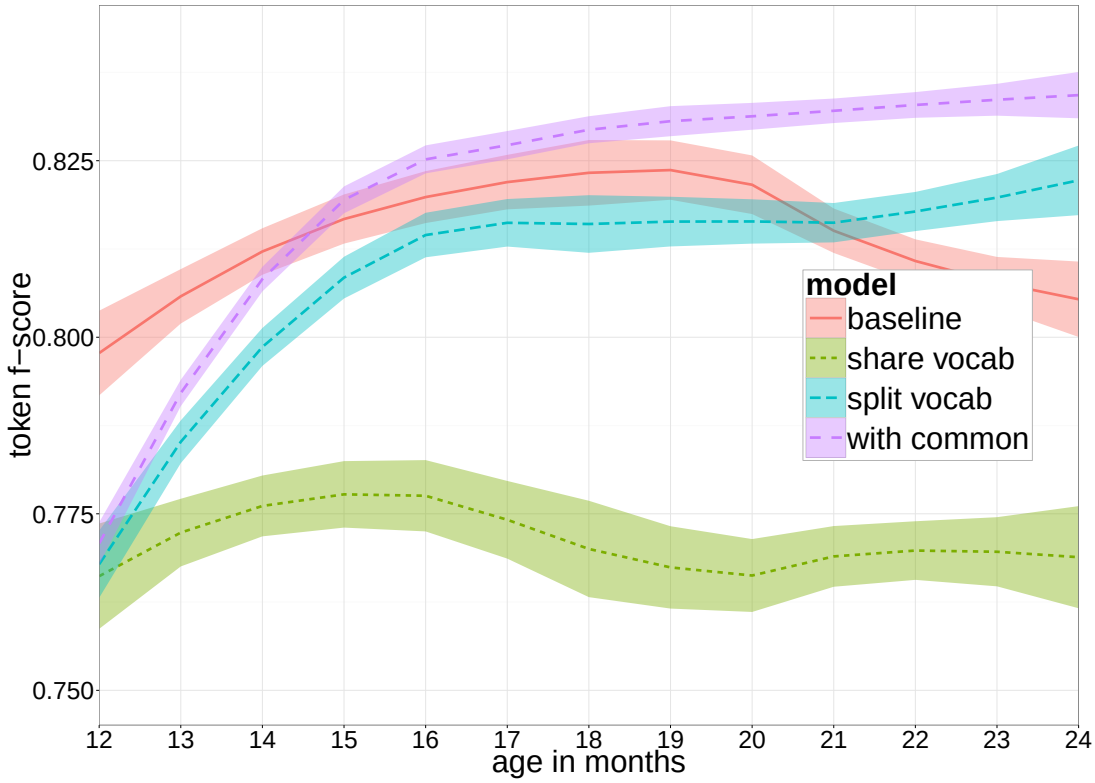
The key metric of interest is the token F-score (harmonic mean of precision and recall of words). Table 2 gives all the scores for an increasingly large dataset (as in (Börschinger et al., 2012)). Figure 1 shows the month-by-month evolution of the token F-score of the different models. We can see that

[2] http://web.science.mq.edu.au/~mjohnson/

[3] https://github.com/SnippyHolloW/contextual_word_segmentation

[4] https://github.com/SnippyHolloW/contextual_word_segmentation/tree/master/ProvidenceFinal/Final

Figure 1: Token F-scores (and standard deviations) evolution with an increasingly bigger and richer dataset (11 months to "X-axis value" months), computed on 8 runs of 500 iterations per data point.



context-based models need more data to get good performances (several vocabularies to learn), but they seem more resilient to over-segmentation.

Preliminary results confirm the trend of *baseline* scores getting slowly worse at **25** and **26** months while *with common* and *split vocab* stabilize (not plotted here). We also tried models for which we can have the "common vocabulary" derived only at the level of the collocations (making topic-specific collocations topic-pure as in *split vocab* for instance), or only at the level of the words (allowing for topic-specific collocations deriving in only common words if needed). Both models are worse than *split vocab* and *with common*.

Using a shared global vocabulary while being able to learn (through adaptation) different topic-specific vocabularies does not seem to be a solution: *share vocab* performs worse than the *baseline*. Token recall and boundary recall are worse off (see Table 2), suggesting that fewer words are correctly adapted. Maybe that is because this is the only model with two levels of adapted word hierarchies ($\underline{Word_{tK}}$ and $\underline{Word}$). Sharing a lower-level vocabulary ($\underline{Word}$) still does not allow for context vocabularies ($\underline{Word_{tK}}$) to mix, thus is simply harder to train. Having only one vocabulary per context (*split vocab*) is a slight improvement over the *baseline*, even though it is not significant (95% confidence interval) before 22 months. Models allowing for both topic-specific vocabularies and a common vocabulary to be learned are the best: *with common* is significantly (95% confidence interval) better than the *baseline*, starting from 20 months (Figure 1). The improvement seems to be due to better token (and boundary) recall (Table 2), suggesting that more words are learned. By looking at their lexicons at 24 months, topic-dependent models have slightly larger lexicon recalls and worse lexicon precisions than the *baseline*. This means that the additional true word-types that they learn are more frequently correctly used than the false word-types (otherwise the token F-scores would be reversed, e.g. between *split vocab* and baseline).

Figure 2: Mean token F-scores (and standard deviations) on 20% held-out test data for 6 different random splits of Naima from 11 to 22 months, 500 iterations each. Grey for *baseline* on *test*, green and blue for context-dependent models on *test* and *no prefix* conditions respectively.

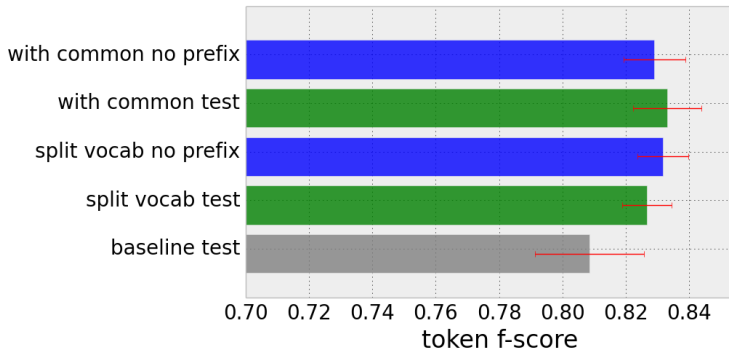


Table 3: Most probable words ($\propto P(word|topic = k)$) in the 7 recovered topics at test time without topic annotations (*no prefix* condition) for the *with common* model (we omitted phonemes clusters yielding non-words).

bread	elephant	lego	Michael	skinny	stick	bubble
delicious	owl	doctor	shorts	massage	remember	pasta
avocado	wearing	brush	towel	ostrich	track	spirals
porridge	turkey	change	shirts	nurse	forget	squirrels
raisin	haircut	squeeze	pirates	hammer	oink	thumb
biscuit	turtle	music	tangled	ruby	towed	pentagon
food	animals	play	clothes	(messy)	verbs	≈shapes

6.2 Recovery of the topics on held-out data

To check whether these models generalize to unseen utterances, and possibly unseen vocabulary, we looked at the scores of held-out data (80/20% train/test split of the Naima 11 to 22 months dataset). Token F-scores for this *test* condition are shown in green and grey in Figure 2. This separates low-frequency collocations to be used at test time and those seen at training time, both for context aware models and the basic *baseline* model. The F-scores show the same pattern as in the previous experiment, with context-aware models (*with common* and *split vocab* here) performing better than the *baseline*.

The topics are learned on the orthographic transcription of the whole *Providence* corpus (6 children), while we test only on the Naima dataset. Still, to check that these results are not simply due to additional information (leaked somehow in the form of the $\_tK$ prefix), we produced another held-out condition, without topic ($\_tK$) prefixes. Models can use topic-specific vocabularies learned during training, but they are given no context information at test time. Token F-scores for this *no prefix* condition are shown in blue (and grey for the baseline) in Figure 2. The fact that *no prefix* performance is on par with the *test* condition means that contextual cues are not only important at test time, but particularly so while learning the vocabulary. In other words, the model acquires its vocabularies making use of the additional context. In the *test* setting, it is evaluated on novel utterances for which additional context information is available. In the *no prefix* condition it is evaluated on novel utterances for which no additional context information is available. This means that topic-specific vocabulary learned during training is successfully used in a consistent way at test time. To confirm this qualitatively, we looked at the most probable words (after unsupervised segmentation from the phonemic input) in recovered topics at test time in the *no prefix* condition. They are shown in Table 3, and they exhibit some of the topics that were found on the orthographic transcript (as they are not limited to nouns, a topic for "verbs" appears).

7 Conclusion

We have shown that contextual information helps segmenting speech into word-like units. We used topic modeling as a proxy for richer contextual annotations, as (Roy et al., 2012) have shown high correlation between contexts and automatically derived topics. We modified existing Adaptor Grammar segmentation models (Johnson, 2008; Johnson and Goldwater, 2009), to be able to learn topic-specific vocabularies. We applied this approach to a large child directed speech corpus that was previously used for segmentation (Börschinger et al., 2012). Our model with the capacity to use both a topic-specific vocabulary and a common vocabulary (*with common*) produces better segmentation scores, ending up with at least 2.5% better absolute F-scores than its context-oblivious counterpart (*baseline*). More generally, both models that learn specialized vocabularies do not get worse F-scores with increasing data (Figure 1). Particularly, they seem to fix a well-known problem of previous models like "colloc-syll" (our *baseline*), that "overlearn" by over-segmenting frequent morphemes as single words (Börschinger et al., 2012). We have controlled for the additional information of giving the topic ($\_tK$), and we have found out that contextual information helps at training time.

It would be interesting to look into the link between semantics and syntax in recovered topics. Further work should integrate syntax (e.g. function words), stress cues and prosody from the audio signal (Börschinger and Johnson, 2014), use even less supervision for contexts, and be applied to other languages. We believe that language acquisition is not a simple sequential process and that segmentation, syntax, and word meaning bootstrap each others. This is only a first step towards integrating multiple sources of information and different modalities at all steps of language acquisition.

Acknowledgments

This project is funded in part by the European Research Council (ERC-2011-AdG-295810 BOOTPHON), the Agence Nationale pour la Recherche (ANR-10-LABX-0087 IEC, ANR-10-IDEX-0001-02 PSL*), the Fondation de France, the Ecole de Neurosciences de Paris, and the Region Ile de France (DIM cerveau et pense).

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

Benjamin Börschinger and Mark Johnson. 2014. Exploring the role of stress in Bayesian word segmentation using Adaptor Grammars. *Transactions of the Association of Computational Linguistics*, 2:93–104, February.

Benjamin Börschinger, Katherine Demuth, and Mark Johnson. 2012. Studying the effect of input size for bayesian word segmentation on the providence corpus. In *COLING*, pages 325–340.

Katherine Demuth, Jennifer Culbertson, and Jennifer Alter. 2006. Word-minimality, epenthesis and coda licensing in the early acquisition of english. *Language and Speech*, 49(2):137–173.

Michael C Frank, Noah D Goodman, and Joshua B Tenenbaum. 2009. Using speakers' referential intentions to model early cross-situational word learning. *Psychological Science*, 20(5):578–585.

Sharon Goldwater, Thomas L Griffiths, and Mark Johnson. 2009. A bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2011. Producing power-law distributions and damping word frequencies with two-stage language models. *Journal of Machine Learning Research*, 12(Jul):2335–2382.

Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325. Association for Computational Linguistics.

Mark Johnson, Thomas L Griffiths, and Sharon Goldwater. 2007. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. *Advances in neural information processing systems*, 19:641.

Mark Johnson, Katherine Demuth, Michael Frank, and Bevan Jones. 2010. Synergies in learning words and their referents. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1018–1026.

Mark Johnson. 2008. Using adaptor grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *ACL*, pages 398–406.

Peter W. Jusczyk and Richard N. Aslin. 1995. Infants detection of the sound patterns of words in fluent speech. *Cognitive psychology*, 29(1):123.

Peter W. Jusczyk, Anne Cutler, and Nancy J. Redanz. 1993a. Infants' preference for the predominant stress patterns of english words. *Child development*, 64(3):675687.

Peter W. Jusczyk, Angela D. Friederici, Jeanine MI Wessels, Vigdis Y. Svenkerud, and Ann Marie Jusczyk. 1993b. Infants sensitivity to the sound patterns of native language words. *Journal of Memory and Language*, 32(3):402420.

Mihael Perman, Jim Pitman, and Marc Yor. 1992. Size-biased sampling of poisson point processes and excursions. *Probability Theory and Related Fields*, 92(1):21–39.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May. ELRA. `http://is.muni.cz/publication/884893/en`.

Deb Roy, Rupal Patel, Philip DeCamp, Rony Kubat, Michael Fleischman, Brandon Roy, Nikolaos Mavridis, Stefanie Tellex, Alexia Salata, Jethran Guinness, et al. 2006. The human speechome project. In *Symbol Grounding and Beyond*, pages 192–196. Springer.

Brandon C Roy, Michael C Frank, and Deb Roy. 2012. Relating activity contexts to early word learning in dense longitudinal data. In *Proceedings of the 34th Annual Cognitive Science Conference*.

Jenny R. Saffran, Richard N. Aslin, and Elissa L. Newport. 1996. Statistical learning by 8-month old infants. *Science*, 274(5294):1926–1928.

Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992.

Relation Classification via Convolutional Deep Neural Network

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou and Jun Zhao
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing 100190, China
{djzeng,kliu,swlai,gyzhou,jzhao}@nlpr.ia.ac.cn

Abstract

The state-of-the-art methods used for relation classification are primarily based on statistical machine learning, and their performance strongly depends on the quality of the extracted features. The extracted features are often derived from the output of pre-existing natural language processing (NLP) systems, which leads to the propagation of the errors in the existing tools and hinders the performance of these systems. In this paper, we exploit a convolutional deep neural network (DNN) to extract lexical and sentence level features. Our method takes all of the word tokens as input without complicated pre-processing. First, the word tokens are transformed to vectors by looking up word embeddings[1]. Then, lexical level features are extracted according to the given nouns. Meanwhile, sentence level features are learned using a convolutional approach. These two level features are concatenated to form the final extracted feature vector. Finally, the features are fed into a softmax classifier to predict the relationship between two marked nouns. The experimental results demonstrate that our approach significantly outperforms the state-of-the-art methods.

1 Introduction

The task of relation classification is to predict semantic relations between pairs of nominals and can be defined as follows: given a sentence S with the annotated pairs of nominals e_1 and e_2, we aim to identify the relations between e_1 and e_2 (Hendrickx et al., 2010). There is considerable interest in automatic relation classification, both as an end in itself and as an intermediate step in a variety of NLP applications.

The most representative methods for relation classification use supervised paradigm; such methods have been shown to be effective and yield relatively high performance (Zelenko et al., 2003; Bunescu and Mooney, 2005; Zhou et al., 2005; Mintz et al., 2009). Supervised approaches are further divided into feature-based methods and kernel-based methods. Feature-based methods use a set of features that are selected after performing textual analysis. They convert these features into symbolic IDs, which are then transformed into a vector using a paradigm that is similar to the bag-of-words model[2]. Conversely, kernel-based methods require pre-processed input data in the form of parse trees (such as dependency parse trees). These approaches are effective because they leverage a large body of linguistic knowledge. However, the extracted features or elaborately designed kernels are often derived from the output of pre-existing NLP systems, which leads to the propagation of the errors in the existing tools and hinders the performance of such systems (Bach and Badaskar, 2007). It is attractive to consider extracting features that are as independent from existing NLP tools as possible.

To identify the relations between pairs of nominals, it is necessary to a skillfully combine lexical and sentence level clues from diverse syntactic and semantic structures in a sentence. For example, in the sentence "The $[\text{fire}]_{e_1}$ inside WTC was caused by exploding $[\text{fuel}]_{e_2}$", to identify that *fire* and *fuel* are in a

[1]A word embedding is a distributed representation for a word. For example, Collobert et al. (2011) use a 50-dimensional vector to represent a word.

[2]http://en.wikipedia.org/wiki/Bag-of-words_model

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2335–2344, Dublin, Ireland, August 23-29 2014.

Cause-Effect relationship, we usually leverage the marked nouns and the meanings of the entire sentence. In this paper, we exploit a convolutional DNN to extract lexical and sentence level features for relation classification. Our method takes all of the word tokens as input without complicated pre-processing, such as Part-of-Speech (POS) tagging and syntactic parsing. First, all the word tokens are transformed into vectors by looking up word embeddings. Then, lexical level features are extracted according to the given nouns. Meanwhile, sentence level features are learned using a convolutional approach. These two level features are concatenated to form the final extracted feature vector. Finally, the features are feed into a softmax classifier to predict the relationship between two marked nouns.

The idea of extracting features for NLP using convolutional DNN was previously explored by Collobert et al. (2011), in the context of POS tagging, chunking (CHUNK), Named Entity Recognition (NER) and Semantic Role Labeling (SRL). Our work shares similar intuition with that of Collobert et al. (2011). In (Collobert et al., 2011), all of the tasks are considered as the sequential labeling problems in which each word in the input sentence is given a tag. However, our task, "relation classification", can be considered a multi-class classification problem, which results in a different objective function. Moreover, relation classification is defined as assigning relation labels to pairs of words. It is thus necessary to specify which pairs of words to which we expect to assign relation labels. For that purpose, the position features (PF) are exploited to encode the relative distances to the target noun pairs. To the best of our knowledge, this work is the first example of using a convolutional DNN for relation classification.

The contributions of this paper can be summarized as follows.

- We explore the feasibility of performing relation classification without complicated NLP pre-processing. A convolutional DNN is employed to extract lexical and sentence level features.
- To specify pairs of words to which relation labels should be assigned, position features are proposed to encode the relative distances to the target noun pairs in the convolutional DNN.
- We conduct experiments using the SemEval-2010 Task 8 dataset. The experimental results demonstrate that the proposed position features are critical for relation classification. The extracted lexical and sentence level features are effective for relation classification. Our approach outperforms the state-of-the-art methods.

2 Related Work

Relation classification is one of the most important topics in NLP. Many approaches have been explored for relation classification, including unsupervised relation discovery and supervised classification. Researchers have proposed various features to identify the relations between nominals using different methods.

In the unsupervised paradigms, contextual features are used. Distributional hypothesis theory (Harris, 1954) indicates that words that occur in the same context tend to have similar meanings. Accordingly, it is assumed that the pairs of nominals that occur in similar contexts tend to have similar relations. Hasegawa et al. (2004) adopted a hierarchical clustering method to cluster the contexts of nominals and simply selected the most frequent words in the contexts to represent the relation between the nominals. Chen et al. (2005) proposed a novel unsupervised method based on model order selection and discriminative label identification to address this problem.

In the supervised paradigm, relation classification is considered a multi-classification problem, and researchers concentrate on extracting more complex features. Generally, these methods can be categorized into two types: feature-based and kernel-based. In feature-based methods, a diverse set of strategies have been exploited to convert the classification clues (such as sequences and parse trees) into feature vectors (Kambhatla, 2004; Suchanek et al., 2006). Feature-based methods suffer from the problem of selecting a suitable feature set when converting the structured representation into feature vectors. Kernel-based methods provide a natural alternative to exploit rich representations of the input classification clues, such as syntactic parse trees. Kernel-based methods allow the use of a large set of features without explicitly extracting the features. Various kernels, such as the convolution tree kernel (Qian et

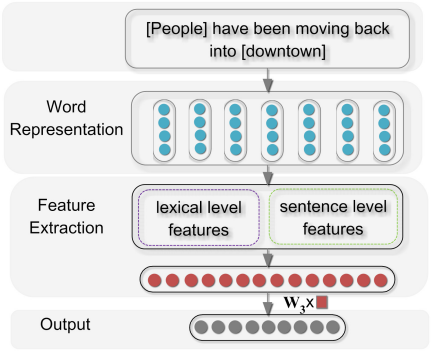


Figure 1: Architecture of the neural network used for relation classification.

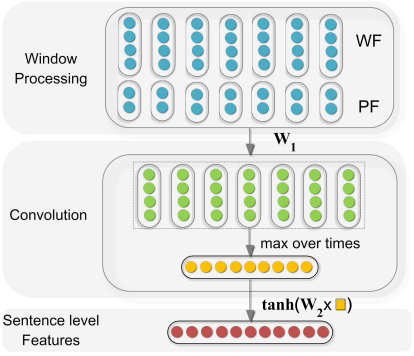


Figure 2: The framework used for extracting sentence level features.

al., 2008), subsequence kernel (Mooney and Bunescu, 2005) and dependency tree kernel (Bunescu and Mooney, 2005), have been proposed to solve the relation classification problem. However, the methods mentioned above suffer from a lack of sufficient labeled data for training. Mintz et al. (2009) proposed distant supervision (DS) to address this problem. The DS method selects sentences that match the facts in a knowledge base as positive examples. The DS algorithm sometimes faces the problem of wrong labels, which results in noisy labeled data. To address the shortcoming of DS, Riedel et al. (2010) and Hoffmann et al. (2011) cast the relaxed DS assumption as multi-instance learning. Furthermore, Takamatsu et al. (2012) noted that the relaxed DS assumption would fail and proposed a novel generative model to model the heuristic labeling process in order to reduce the wrong labels.

The supervised method has been demonstrated to be effective for relation detection and yields relatively high performance. However, the performance of this method strongly depends on the quality of the designed features. With the recent revival of interest in DNN, many researchers have concentrated on using *Deep Learning* to learn features. In NLP, such methods are primarily based on learning a distributed representation for each word, which is also called a word embeddings (Turian et al., 2010). Socher et al. (2012) present a novel recursive neural network (RNN) for relation classification that learns vectors in the syntactic tree path that connects two nominals to determine their semantic relationship. Hashimoto et al. (2013) also use an RNN for relation classification; their method allows for the explicit weighting of important phrases for the target task. As mentioned in Section 1, it is difficult to design high quality features using the existing NLP tools. In this paper, we propose a convolutional DNN to extract lexical and sentence level features for relation classification; our method effectively alleviates the shortcomings of traditional features.

3 Methodology

3.1 The Neural Network Architecture

Figure 1 describes the architecture of the neural network that we use for relation classification. The network takes an input sentence and discovers multiple levels of feature extraction, where higher levels represent more abstract aspects of the inputs. It primarily includes the following three components: *Word Representation*, *Feature Extraction* and *Output*. The system does not need any complicated syntactic or semantic preprocessing, and the input of the system is a sentence with two marked nouns. Then, the word tokens are transformed into vectors by looking up word embeddings. In succession, the lexical and sentence level features are respectively extracted and then directly concatenated to form the final feature vector. Finally, to compute the confidence of each relation, the feature vector is fed into a softmax classifier. The output of the classifier is a vector, the dimension of which is equal to the number of predefined relation types. The value of each dimension is the confidence score of the corresponding relation.

Features	Remark
L1	Noun 1
L2	Noun 2
L3	Left and right tokens of noun 1
L4	Left and right tokens of noun 2
L5	WordNet hypernyms of nouns

Table 1: Lexical level features.

3.2 Word Representation

In the *word representation* component, each input word token is transformed into a vector by looking up word embeddings. Collobert et al. (2011) reported that word embeddings learned from significant amounts of unlabeled data are far more satisfactory than the randomly initialized embeddings. In relation classification, we should first concentrate on learning discriminative word embeddings, which carry more syntactic and semantic information, using significant amounts of unlabeled data. Unfortunately, it usually takes a long time to train the word embeddings[3]. However, there are many trained word embeddings that are freely available (Turian et al., 2010). A comparison of the available word embeddings is beyond the scope of this paper. Our experiments directly utilize the trained embeddings provided by Turian et al.(2010).

3.3 Lexical Level Features

Lexical level features serve as important cues for deciding relations. The traditional lexical level features primarily include the nouns themselves, the types of the pairs of nominals and word sequences between the entities, the quality of which strongly depends on the results of existing NLP tools. Alternatively, this paper uses generic word embeddings as the source of base features. We select the word embeddings of marked nouns and the context tokens. Moreover, the WordNet hypernyms[4] are adopted as MVRNN (Socher et al., 2012). All of these features are concatenated into our lexical level features vector $\mathbf{l}$. Table 1 presents the selected word embeddings that are related to the marked nouns in the sentence.

3.4 Sentence Level Features

As mentioned in section 3.2, all of the tokens are represented as word vectors, which have been demonstrated to correlate well with human judgments of word similarity. Despite their success, single word vector models are severely limited because they do not capture long distance features and semantic compositionality, the important quality of natural language that allows humans to understand the meanings of a longer expression. In this section, we propose a max-pooled convolutional neural network to offer sentence level representation and automatically extract sentence level features. Figure 2 shows the framework for sentence level feature extraction. In the *Window Processing* component, each token is further represented as Word Features (WF) and Position Features (PF) (see section 3.4.1 and 3.4.2). Then, the vector goes through a convolutional component. Finally, we obtain the sentence level features through a non-linear transformation.

3.4.1 Word Features

Distributional hypothesis theory (Harris, 1954) indicates that words that occur in the same context tend to have similar meanings. To capture this characteristic, the WF combines a word's vector representation and the vector representations of the words in its context. Assume that we have the following sequence of words.

$$S : [\text{People}]_0 \quad \text{have}_1 \quad \text{been}_2 \quad \text{moving}_3 \quad \text{back}_4 \quad \text{into}_5 \quad [\text{downtown}]_6$$

The marked nouns are associated with a label y that defines the relation type that the marked pair contains. Each word is also associated with an index into the word embeddings. All of the word tokens of the sentence S are then represented as a list of vectors $(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_6)$, where $\mathbf{x}_i$ corresponds to the word

[3]Collobert et al. (2011) proposed a *pairwise ranking* approach to train the word embeddings, and the total training time for an English corpus (Wikipedia) was approximately four weeks.

[4]http://sourceforge.net/projects/supersensetag/

embedding of the i-th word in the sentence. To use a context size of w, we combine the size w windows of vectors into a richer feature. For example, when we take $w = 3$, the WF of the third word "moving" in the sentence *S* is expressed as $[\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4]$. Similarly, considering the whole sentence, the WF can be represented as follows:

$$\{[\mathbf{x}_s, \mathbf{x}_0, \mathbf{x}_1],\ [\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2],\ \cdots,\ [\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_e]\}^5$$

3.4.2 Position Features

Relation classification is a very complex task. Traditionally, structure features (e.g., the shortest dependency path between nominals) are used to solve this problem (Bunescu and Mooney, 2005). Apparently, it is not possible to capture such structure information only through WF. It is necessary to specify which input tokens are the target nouns in the sentence. For this purpose, PF are proposed for relation classification. In this paper, the PF is the combination of the relative distances of the current word to w_1 and w_2. For example, the relative distances of "moving" in sentence S to "people" and "downtown" are 3 and -3, respectively. In our method, the relative distances also are mapped to a vector of dimension d_e (a hyperparameter); this vector is randomly initialized. Then, we obtain the distance vectors $\mathbf{d}_1$ and $\mathbf{d}_2$ with respect to the relative distances of the current word to w_1 and w_2, and PF = $[\mathbf{d}_1, \mathbf{d}_2]$. Combining the WF and PF, the word is represented as $[\text{WF}, \text{PF}]^T$, which is subsequently fed into the convolution component of the algorithm.

3.4.3 Convolution

We will see that the word representation approach can capture contextual information through combinations of vectors in a window. However, it only produces local features around each word of the sentence. In relation classification, an input sentence that is marked with target nouns only corresponds to a relation type rather than predicting label for each word. Thus, it might be necessary to utilize all of the local features and predict a relation globally. When using neural network, the convolution approach is a natural method to merge all of the features. Similar to Collobert et al. (2011), we first process the output of *Window Processing* using a linear transformation.

$$\mathbf{Z} = \mathbf{W}_1\mathbf{X} \tag{1}$$

$\mathbf{X} \in \mathbb{R}^{n_0 \times t}$ is the output of the *Window Processing* task, where $n_0 = w \times n$, n (a hyperparameter) is the dimension of feature vector, and t is the token number of the input sentence. $\mathbf{W}_1 \in \mathbb{R}^{n_1 \times n_0}$, where n_1 (a hyperparameter) is the size of hidden layer 1, is the linear transformation matrix. We can see that the features share the same weights across all times, which greatly reduces the number of free parameters to learn. After the linear transformation is applied, the output $\mathbf{Z} \in \mathbb{R}^{n_1 \times t}$ is dependent on t. To determine the most useful feature in the each dimension of the feature vectors, we perform a max operation over time on $\mathbf{Z}$.

$$m_i = \max \mathbf{Z}(i, \cdot) \quad 0 \le i \le n_1 \tag{2}$$

where $\mathbf{Z}(i, \cdot)$ denote the i-th row of matrix $\mathbf{Z}$. Finally, we obtain the feature vector $\mathbf{m} = \{m_1, m_2, \cdots, m_{n_1}\}$, the dimension of which is no longer related to the sentence length.

3.4.4 Sentence Level Feature Vector

To learn more complex features, we designed a non-linear layer and selected hyperbolic tanh as the activation function. One useful property of tanh is that its derivative can be expressed in terms of the function value itself:

$$\frac{d}{dx}\tanh x = 1 - \tanh^2 x \tag{3}$$

It has the advantage of making it easy to compute the gradient in the backpropagation training procedure. Formally, the non-linear transformation can be written as

$$\mathbf{g} = \tanh(\mathbf{W}_2\mathbf{m}) \tag{4}$$

[5] $\mathbf{x}_s$ and $\mathbf{x}_e$ are special word embeddings that correspond to the beginning and end of the sentence, respectively.

$\mathbf{W_2} \in \mathbb{R}^{n_2 \times n_1}$ is the linear transformation matrix, where n_2 (a hyperparameter) is the size of hidden layer 2. Compared with $\mathbf{m} \in \mathbb{R}^{n_1 \times 1}$, $\mathbf{g} \in \mathbb{R}^{n_2 \times 1}$ can be considered higher level features (sentence level features).

3.5 Output

The automatically learned lexical and sentence level features mentioned above are concatenated into a single vector $\mathbf{f} = [\mathbf{l}, \mathbf{g}]$. To compute the confidence of each relation, the feature vector $\mathbf{f} \in \mathbb{R}^{n_3 \times 1}$ (n_3 equals n_2 plus the dimension of the lexical level features) is fed into a softmax classifier.

$$\mathbf{o} = \mathbf{W}_3 \mathbf{f} \tag{5}$$

$\mathbf{W_3} \in \mathbb{R}^{n_4 \times n_3}$ is the transformation matrix and $\mathbf{o} \in \mathbb{R}^{n_4 \times 1}$ is the final output of the network, where n_4 is equal to the number of possible relation types for the relation classification system. Each output can be then interpreted as the confidence score of the corresponding relation. This score can be interpreted as a conditional probability by applying a softmax operation (see Section 3.6).

3.6 Backpropagation Training

The DNN based relation classification method proposed here could be stated as a quintuple $\theta = (\mathbf{X}, \mathbf{N}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3)$[6]. In this paper, each input sentence is considered independently. Given an input example s, the network with parameter θ outputs the vector $\mathbf{o}$, where the i-th component o_i contains the score for relation i. To obtain the conditional probability $p(i|x, \theta)$, we apply a softmax operation over all relation types:

$$p(i|x, \theta) = \frac{e^{o_i}}{\sum_{k=1}^{n_4} e^{o_k}} \tag{6}$$

Given all our (suppose T) training examples $(x^{(i)}; y^{(i)})$, we can then write down the log likelihood of the parameters as follows:

$$J(\theta) = \sum_{i=1}^{T} \log p(y^{(i)}|x^{(i)}, \theta) \tag{7}$$

To compute the network parameter θ, we maximize the log likelihood $J(\theta)$ using a simple optimization technique called stochastic gradient descent (SGD). $\mathbf{N}, \mathbf{W}_1, \mathbf{W}_2$ and $\mathbf{W}_3$ are randomly initialized and $\mathbf{X}$ is initialized using the word embeddings. Because the parameters are in different layers of the neural network, we implement the backpropagation algorithm: the differentiation chain rule is applied through the network until the word embedding layer is reached by iteratively selecting an example (x, y) and applying the following update rule.

$$\theta \leftarrow \theta + \lambda \frac{\partial \log p(y|x, \theta)}{\partial \theta} \tag{8}$$

4 Dataset and Evaluation Metrics

To evaluate the performance of our proposed method, we use the SemEval-2010 Task 8 dataset (Hendrickx et al., 2010). The dataset is freely available[7] and contains 10,717 annotated examples, including 8,000 training instances and 2,717 test instances. There are 9 relationships (with two directions) and an undirected *Other* class. The following are examples of the included relationships: *Cause-Effect*, *Component-Whole* and *Entity-Origin*. In the official evaluation framework, directionality is taken into account. A pair is counted as correct if the order of the words in the relationship is correct. For example, both of the following instances S_1 and S_2 have the relationship *Component-Whole*.

S_1 : The [haft]$_{e_1}$ of the [axe]$_{e_2}$ is make $\cdots \Rightarrow$ Component-Whole(e_1,e_2)
S_2 : This [machine]$_{e_1}$ has two [units]$_{e_2}$ $\cdots \Rightarrow$ Component-Whole(e_2,e_1)

[6] $\mathbf{N}$ represents the word embeddings of WordNet hypernyms.
[7] http://docs.google.com/View?id=dfvxd49s_36c28v9pmw

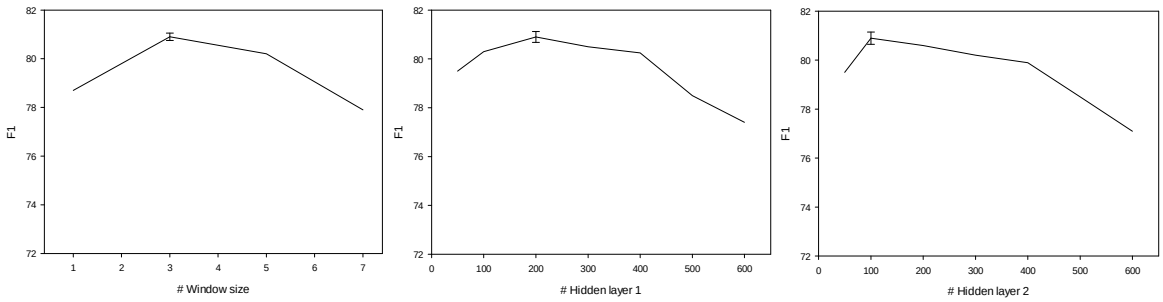


Figure 3: Effect of hyperparameters.

However, these two instances cannot be classified into the same category because *Component-Whole(e_1,e_2)* and *Component-Whole(e_2,e_1)* are different relationships. Furthermore, the official ranking of the participating systems is based on the macro-averaged F1-scores for the nine proper relations (excluding *Other*). To compare our results with those obtained in previous studies, we adopt the macro-averaged F1-score and also account for directionality into account in our following experiments[8].

5 Experiments

In this section, we conduct three sets of experiments. The first is to test several variants via cross-validation to gain some understanding of how the choice of hyperparameters impacts upon the performance. In the second set of experiments, we make comparison of the performance among the convolutional DNN learned features and various traditional features. The goal of the third set of experiments is to evaluate the effectiveness of each extracted feature.

5.1 Parameter Settings

In this section, we experimentally study the effects of the three parameters in our proposed method: the window size in the convolutional component w, the number of hidden layer 1, and the number of hidden layer 2. Because there is no official development dataset, we tuned the hyperparameters by trying different architectures via 5-fold cross-validation.

In Figure 3, we respectively vary the number of hyper parameters w, n_1 and n_2 and compute the F1. We can see that it does not improve the performance when the window size is greater than 3. Moreover, because the size of our training dataset is limited, the network is prone to overfitting, especially when using large hidden layers. From Figure 3, we can see that the parameters have a limited impact on the results when increasing the numbers of both hidden layers 1 and 2. Because the distance dimension has little effect on the result (this is not illustrated in Figure 3), we heuristically choose $d_e = 5$. Finally, the word dimension and learning rate are the same as in Collobert et al. (2011). Table 2 reports all the hyperparameters used in the following experiments.

Hyperparameter	*Window size*	*Word dim.*	*Distance dim.*	*Hidden layer 1*	*Hidden layer 2*	*Learning rate*
Value	$w = 3$	$n = 50$	$d_e = 5$	$n_1 = 200$	$n_2 = 100$	$\lambda = 0.01$

Table 2: Hyperparameters used in our experiments.

5.2 Results of Comparison Experiments

To obtain the final performance of our automatically learned features, we select seven approaches as competitors to be compared with our method in Table 3. The first five competitors are described in Hendrickx et al. (2010), all of which use traditional features and employ SVM or MaxEnt as the classifier. These systems design a series of features and take advantage of a variety of resources (WordNet, ProBank, and FrameNet, for example). *RNN* represents recursive neural networks for relation classification, as

[8]The corpus contains a Perl-based automatic evaluation tool.

Classifier	Feature Sets	F1
SVM	POS, stemming, syntactic patterns	60.1
SVM	word pair, words in between	72.5
SVM	POS, stemming, syntactic patterns, WordNet	74.8
MaxEnt	POS, morphological, noun compound, thesauri, Google n-grams, WordNet	77.6
SVM	POS, prefixes, morphological, WordNet, dependency parse, Levin classed, ProBank, FrameNet, NomLex-Plus, Google n-gram, paraphrases, TextRunner	82.2
RNN	-	74.8
	POS, NER, WordNet	77.6
MVRNN	-	79.1
	POS, NER, WordNet	82.4
Proposed	word pair, words around word pair, WordNet	**82.7**

Table 3: Classifier, their feature sets and the F1-score for relation classification.

proposed by Socher et al. (2012). This method learns vectors in the syntactic tree path that connect two nominals to determine their semantic relationship. The *MVRNN* model builds a single compositional semantics for the minimal constituent, including both nominals as *RNN* (Socher et al., 2012). It is almost certainly too much to expect a single fixed transformation to be able to capture the meaning combination effects of all natural language operators. Thus, *MVRNN* assigns a matrix to every word and modifies the meanings of other words instead of only considering word embeddings in the recursive procedure.

Table 3 illustrates the macro-averaged F1 measure results for these competing methods along with the resources, features and classifier used by each method. Based on these results, we make the following observations:

(1) Richer feature sets lead to better performance when using traditional features. This improvement can be explained by the need for semantic generalization from training to test data. The quality of traditional features relies on human ingenuity and prior NLP knowledge. It is almost impossible to manually choose the best feature sets.

(2) *RNN* and *MVRNN* contain feature learning procedures; thus, they depend on the syntactic tree used in the recursive procedures. Errors in syntactic parsing inhibit the ability of these methods to learn high quality features. *RNN* cannot achieve a higher performance than the best method that uses traditional features, even when POS, NER and WordNet are added to the training dataset. Compared with *RNN*, the *MVRNN* model can capture the meaning combination effectively and achieve a higher performance.

(3) Our method achieves the best performance among all of the compared methods. We also perform a t-test ($p \leqslant 0.05$), which indicates that our method significantly outperforms all of the compared methods.

5.3 The Effect of Learned Features

	Feature Sets	F1
Lexical	L1	34.7
	+L2	53.1
	+L3	59.4
	+L4	65.9
	+L5	73.3
Sentence	WF	69.7
	+PF	78.9
Combination	all	82.7

Table 4: Score obtained for various sets of features on for the test set. The bottom portion of the table shows the best combination of lexical and sentence level features.

In our method, the network extract lexical and sentence level features. The lexical level features primarily contain five sets of features (L1 to L5). We performed ablation tests on the five sets of features from the lexical part of Table 4 to determine which type of features contributed the most. The results are

presented in Table 4, from which we can observe that our learned lexical level features are effective for relation classification. The F1-score is improved remarkably when new features are added. Similarly, we perform experiment on the sentence level features. The system achieves approximately 9.2% improvements when adding PF. When all of the lexical and sentence level features are combined, we achieve the best result.

6 Conclusion

In this paper, we exploit a convolutional deep neural network (DNN) to extract lexical and sentence level features for relation classification. In the network, position features (PF) are successfully proposed to specify the pairs of nominals to which we expect to assign relation labels. The system obtains a significant improvement when PF are added. The automatically learned features yield excellent results and can replace the elaborately designed features that are based on the outputs of existing NLP tools.

Acknowledgments

This work was sponsored by the National Basic Research Program of China (No. 2014CB340503) and the National Natural Science Foundation of China (No. 61272332, 61333018, 61202329, 61303180). This work was supported in part by Noah's Ark Lab of Huawei Tech. Co. Ltd. We thank the anonymous reviewers for their insightful comments.

References

Nguyen Bach and Sameer Badaskar. 2007. A review of relation extraction. *Literature review for Language and Statistics II*.

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

Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. 2005. Unsupervised feature selection for relation extraction. In *Proceedings of the International Joint Conference on Natural Language Processing*, pages 262–267.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 415–422.

Kazuma Hashimoto, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. 2013. Simple customization of recursive neural networks for semantic relation classification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1372–1376.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó. Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 33–38.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 541–550.

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics on Interactive poster and demonstration sessions*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 1003–1011.

Raymond J Mooney and Razvan C Bunescu. 2005. Subsequence kernels for relation extraction. In *Advances in neural information processing systems*, pages 171–178.

Longhua Qian, Guodong Zhou, Fang Kong, Qiaoming Zhu, and Peide Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 697–704.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part III*, pages 148–163.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211.

Fabian M. Suchanek, Georgiana Ifrim, and Gerhard Weikum. 2006. Combining linguistic and statistical analysis to extract relations from web documents. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 712–717.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, pages 721–729.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394.

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

GuoDong Zhou, Su Jian, Zhang Jie, and Zhang Min. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 427–434.

A context-based model for Sentiment Analysis in Twitter

Andrea Vanzo and **Danilo Croce** and **Roberto Basili**
Department of Enterprise Engineering
University of Roma Tor Vergata
Via del Politecnico 1, 00133 Roma Italy
{vanzo,croce,basili}@info.uniroma2.it

Abstract

Most of the recent literature on Sentiment Analysis over Twitter is tied to the idea that the sentiment is a function of an incoming tweet. However, tweets are filtered through streams of posts, so that a wider context, e.g. a topic, is always available. In this work, the contribution of this contextual information is investigated. We modeled the polarity detection problem as a sequential classification task over streams of tweets. A Markovian formulation of the Support Vector Machine discriminative model as embodied by the SVM^{hmm} algorithm has been here employed to assign the sentiment polarity to entire sequences. The experimental evaluation proves that sequential tagging effectively embodies evidence about the contexts and is able to reach a relative increment in detection accuracy of around 20% in F1 measure. These results are particularly interesting as the approach is flexible and does not require manually coded resources.

1 Introduction

Since in the Web 2.0 users can write about their life, personal experiences, share contents about facts and ideas, Social Networks became valuable sources of opinions and sentiments. This huge amount of data is crucial in the study of the interactions and dynamics of subjectivity on the Web, largely relevant for marketing tasks. Twitter is one among these microblogging services that counts about a billion of active users and 500 million of daily messages[1]. However, the analysis of this huge amount of information is still challenging, as language is very informal, affected by misspelling and characterized by slang and *#hashtags*, i.e. special user-generated tags used to contextualize different tweets around a specific topic.

Researches focused on the computational study and automatic recognition of opinions and sentiments as they are expressed in free texts. It gave rise to what is currently known as Sentiment Analysis, a set of tasks aiming to detect the subjective attitude of a writer with respect to some topic. Many Sentiment Analysis studies map sentiment detection in a *Machine Learning* (ML) setting (Pang and Lee, 2008), where labeled data, i.e. known examples, allow to induce the detection function from real world examples. In general, sentiment detection in tweets has been generally treated as any other text classification task, as proved by most papers participating to the *Sentiment Analysis in Twitter* task in SemEval-2013 challenge (Nakov et al., 2013): a computational representation for an incoming instance is generated by just considering one tweet at a time. The short length of the message and the resulting semantic ambiguity are critical limitations and make the task very complex. Let us consider the following example, in which a tweet from `ColMustard` cites `SergGray`:

`ColMustard` : *@SergGray Yes, I totally agree with you about the substitutions! #Bayern #Freiburg*

The tweet sounds like to be a reply to the previous one. Notice how no lexical nor syntactic property allows to determine the sentiment polarity. However, if we look at the entire conversation that follows:

`ColMustard` : *Amazing match yesterday!!#Bayern vs. #Freiburg 4-0 #easyvictory*
`SergGray` : *@ColMustard Surely, but #Freiburg wasted lot of chances to score.. wrong substitutions by #Guardiola during the 2nd half!!*
`ColMustard` : *@SergGray Yes, I totally agree with you about the substitutions! #Bayern #Freiburg*

[1] http://expandedramblings.com/

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2345–2354, Dublin, Ireland, August 23-29 2014.

it is easy to establish that a first positive tweet has been produced, followed by a second negative one so that the third tweet is negative as well. It is the conversation that allows us here to disambiguate even a very short message and properly characterize it according to its author and posting time.

We want here to capitalize such a richer set of observations (i.e. entire conversations) and to define a context-sensitive SA model along two lines: first, by enriching a tweet representation to include the conversation information, and then introducing a more complex classification model that works over an entire tweet sequence and not on one tweet (i.e. the target) at a time. Accordingly, in the paper we will first focus on different representations of tweets that can be made available to the sentiment detection process. They will also account for contexts, that are *conversations*, as chains of tweets that are reply to the previous ones, and *topics*, built around hashtags. These are in fact topics made explicit by users, such as events (*#easyvictory*) or people (*#Guardiola*). It represents a wider notion of conversation that enforces the sense of belonging to a community. From a computational perspective, the polarity detection of a tweet in a context is here modeled as a sequential classification task. In fact, both conversation and topic-based context are arbitrarily long sequences of messages, ordered according to *time* with the target tweet being the last. The SVM^{hmm} learning algorithm (Altun et al., 2003) has been employed, as it allows to classify an instance (here, a tweet) within an entire sequence. While SVM based classifiers allow to recognize the sentiments from one specific tweet at a time, the SVM^{hmm} learning algorithm collectively labels all tweets in a sequence. It is thus expected to capture patterns within a conversation and apply them in novel sequences, through a standard decoding task.

While all the above contexts extend a tweet representation, they are still *local* to a specific notion of conversation. In this work, we also explore the somehow more abstract notion of contexts given by the emotional attitude shown by each user in his overall usage of Twitter. In the above example, `ColMustard` shows a specific attitude while discussing about the Bayern Munchen. We can imagine that this feature characterizes most of its future messages at least about football. We suggest to enrich the tweet representation with features that *synthesize* a user's profile, in order to catch possible biases towards a particular sentiment polarity. This is quite interesting as it has been shown that communities behave in a coherent way and users tend to take stable standing points. Experimental evaluation (Chapter 4) proves the effectiveness of this proposed sequential tagging approach combined with the adopted contextual information, improving the percentage of correctly recognized tweets up to 12%.

A survey of the existing approaches is presented into Section 2. Then, Section 3 provides an account of the context-based models: conversation, topic-based and user sentiment profiling. The experimental evaluation into Section 4 prove the positive impact of social dynamics on the SA task.

2 Related Work

Sentiment Analysis has been described as a *Natural Language Processing* task at many levels of granularity. Starting from being mapped into a *document level* classification task (Turney, 2002; Pang and Lee, 2004), it has been also applied at *sentence level* (Hu and Liu, 2004; Kim and Hovy, 2004) and more recently at the *phrase level* (Wilson et al., 2005; Agarwal et al., 2009).

The spreading of microblog services where users post real-time opinions about "everything", poses newer and different challenges. Indeed, classical approaches to Sentiment Analysis (Pang et al., 2002; Pang and Lee, 2008) are not directly applicable to tweets: while most of them focus on relatively large texts, e.g. movie or product reviews, tweets are very short and fine-grained lexical analysis is required. Nevertheless, the great prominence of Social Media during the last few years encouraged a focus on the sentiment detection over a microblogging domain. Recent works tried to model the sentiment in tweets (Go et al., 2009; Pak and Paroubek, 2010; Kouloumpis et al., 2011; Davidov et al., 2010; Bifet and Frank, 2010; Croce and Basili, 2012; Barbosa and Feng, 2010; Zanzotto et al., 2011; Si et al., 2013; Agarwal et al., 2011). Specific approaches and feature modeling are used to improve accuracy levels in tweet polarity recognition. For example, the use of *n*-grams, POS tags, polarity lexicons and tweet specific features (e.g. hashtags, re-tweets) are some of the component exploited by these works, in combination with different machine learning algorithms: among these latter, probabilistic paradigms, e.g. Naive Bayes (Pak and Paroubek, 2010), or Kernel-based machines, as discussed in (Barbosa and Feng,

2010; Agarwal et al., 2011; Castellucci et al., 2013), are mostly employed. An interesting perspective, where a kind of contextual information is studied, is presented in (Mukherjee and Bhattacharyya, 2012): the sentiment detection of tweets is here modeled according to lexical features as well as discourse relations like the presence of connectives, conditionals and semantic operators like *modals* and *negations*. Nevertheless, in all the above approaches, features are derived only from lexical resources or from the tweet itself and no contextual information is exploited. However, given one tweet targeted for sentiment detection, more awareness about its content is available to writers and readers by the entire stream of related posts immediately preceding it. In order to exploit this wider information, a Markovian extension of a Kernel-based categorization approach is proposed in the next section.

3 A context based model for Sentiment Analysis in Twitter

As discussed in the introduction, contextual information about one tweet stems from various aspects: an explicit conversation, the user attitude or the overall set of recent tweets about a topic (for example an hastag like `#Bayern`). As individual perspectives on the context are independent (a conversation may or may not depend on user preference or cheer) and they also obey to different notion of analogies or similarity, we should avoid a unified feature vector, but employ independent representations. A structured view on a tweet can thus be provided by considering it as multifaceted entity where a set of vectors, each one contributing to one aspect of the overall representation, exhibits a specific similarity metrics. Notice how this is exactly what Kernel-based learning supports, whereas the combination of the different Kernel functions can be easily made a Kernel function itself (Shawe-Taylor and Cristianini, 2004). Kernel functions are used to capture specific aspects of the semantic relatedness between two tweets and are easily integrated in various Machine Learning algorithms, such as SVM.

3.1 Representing tweets through different Kernel functions

Many Machine Learning approaches for Sentiment Analysis in Twitter benefited by complex ways of modeling of individual tweets, as discussed in many works (Nakov et al., 2013). The representation we propose makes use of individual Kernels as models of different aspects usable within a SVM paradigm.

Bag of Word Kernel (BoWK). The simplest Kernel function describes the lexical overlap between tweets, thus represented as vectors, whose dimensions correspond to the different words. Components denote the presence or not of the corresponding word in the text and Kernel function corresponds to the *cosine similarity* between vector pairs. Even if very simple, the BoW model is one of the most informative representation in Sentiment Analysis, as emphasized since (Pang et al., 2002).

Lexical Semantic Kernel (LSK). Lexical information in tweets can be very sparse, as we will also show in the next Section 4. In order to extend the BoW model, we provide a further vector representation aiming to generalize the lexical information. It can be obtained for every term of a dictionary by a co-occurrence Word Space built according to the Distributional Analysis described in (Sahlgren, 2006). A word-by-context matrix, M, is built through large scale corpus analysis and then processed through *Latent Semantic Analysis* (Landauer and Dumais, 1997). The dimensionality of the space represented by M can be reduced through Singular Value Decomposition (SVD) (Golub and Kahan, 1965). The original statistical information about M is captured by the new k-dimensional space, which preserves the global structure while removing low-variance dimensions, i.e. distribution noise. The result is that every word is projected in the reduced Word Space and a vector for each tweet is represented through the linear combination of the co-occurring word vectors (also called *additive linear combination* in (Mitchell and Lapata, 2010)). The resulting Kernel function is the *cosine similarity* between tweet vector pairs, in line with (Cristianini et al., 2002). Notice that the adoption of a distributional approach does not limit the overall application, as it can be automatically applied without relying on any manually coded resource.

User Sentiment Profile Context (**USPK**). A source of evidence about a tweet is its author, with his attitude towards some polarities. Specific features based on the users' previous tweets can be derived as follows. Let $t_i \in \mathcal{T}$ be a tweet and $i \in \mathbb{N}^+$ its identifier. The *User Profile Context* (U_i) can be defined as the set of the last H tweets posted by the author of t_i, hereafter denoted by u_i. This information is a body of evidence about the opinion holder's profile on which a further tweet representation can be defined. A tweet t_i is here mapped into a three dimensional vector $\vec{\mu}_i = \left(\mu_i^1, \mu_i^2, \mu_i^3\right)$, where each component μ_i^j is

the indicator of polarity inclination, i.e. *positive*, *negative* and *neutral*, expressed through the conditional probability $P(j \mid u_i)$ for the polarity labels $j \in \mathcal{Y}$ given the user u_i. We can suppose that, for each $t_k \in U_i$, its corresponding label y_k is available either as a gold standard annotation or predicted in a semi-supervised fashion by trained classifiers. The estimation of $\mu_i^j \approx P(j \mid u_i)$, is a σ-parameterized *Laplace smoothed version* of the observations in U_i: $\mu_i^j = \sum_{k=1}^{|U_i|}(\mathbb{1}_{\{y_k=j\}}(t_k) + \sigma)/(|U_i| + \sigma|\mathcal{Y}|)$ where $\sigma \in \mathbb{R}$ is the smoothing parameter, $j \in \mathcal{Y}$, i.e. the set of polarity labels. The Kernel function, called User Sentiment Profile Kernel (USPK), is the *cosine similarity* between two vectors $(\vec{\mu}_i, \vec{\mu}_m)$.

The multiple Kernel approach. Whenever the different Kernels are available, we can apply a linear combination αBoWK+βLSK or αBoWK+βLSK+γUSPK in order to exploit lexical and semantic properties captured by BoWK and LSK, or user properties as captured by USPK.

3.2 Modeling tweet conversation as a sequential tagging problem

The User Sentiment Profile Kernel (UPSK) can be seen as an implicit representation of a context describing the writer. However, contextual information is usually embodied by the stream of tweets in which the target one t_i is immersed. Usually, the stream is something available to a reader and includes an entire *conversation* (where links to the previous tweets are made explicit and are supposed to be all available) or a *topic*, i.e. a hashtag, the reader has searched for. In all cases, the stream give rise to an entire sequence on which sequence labeling can be applied: the target tweet is here always labeled within the entire sequence, where contextual constraints are provided by the preceding tweets. More formally, two types of context are defined:

Conversational context. For every tweet $t_i \in \mathcal{T}$, let $r(t_i) : \mathcal{T} \rightarrow \mathcal{T}$ be a function that returns either the tweet to which t_i is a reply to, or $null$ if t_i is not a reply. Then, the *conversation-based context* $\Lambda_i^{C,l}$ of tweet t_i (i.e., the *target tweet*) is the sequence of tweet iteratively built by applying $r(\cdot)$, until l tweets have been selected or $r(\cdot) = null$. In other words, l allows to limit the size of the input context. An example of conversation-based context is given in Section 1.

Topical context. Hashtags allow to aggregate different tweets around a specific topic. An entire tweet sequence can be derived including the n tweets preceding the target t_i that contain the same hashtag set. This is usually the output of a search in Twitter and it is likely the source information that influenced the writer's opinion. Let $t_i \in \mathcal{T}$ be a tweet and $h(i) : \mathcal{T} \rightarrow \mathcal{P}(\mathcal{H})$ be a function that returns the entire hashtag set $H_i \subseteq \mathcal{H}$ observed into t_i. Then, the *hashtag-based context* $\Lambda_i^{H,l}$ for a tweet t_i (i.e., *target tweet*) is a sequence of the most recent l tweets t_j such that $H_j \cap H_i \neq \emptyset$, i.e. t_j and t_i share at least one hashtag, and t_j has been posted before t_i. As an example, the following hashtag-based context of size 4 has been obtained about `#Bayern`:

`MrGreen` : *Fun fact: #Freiburg is the only #Bundesliga team #Pep has never beaten in his coaching career. #Bayern*
`MrsPeacock` : *Young starlet Xherdan #Shaqiri fires #Bayern into a 2-0 lead. Is there any hope for #Freiburg? pic.twitter.com/krzbFJFJyN*
`ProfPlum` : *It is clear that #Bayern is on a rampage leading by 4-0, the latest by Mandzukic... hoping for another 2 goals from #bayernmunich*
`MissScarlet` : *Noooo! I cant believe what #Bayern did!*

It is clear that `MissScarlet` expressed an opinion, but the corresponding polarity is easily evident when the entire stream is available about the *#Bayern* hashtag. As well as in a conversational context, a specific context size n can be imposed by focusing only on the last n tweets of the sequence. Once different representations and contexts are available a structured learning-based approach can be applied to Sentiment Analysis. Firstly, we will discuss a discriminative learning approach that follows the multi-classification schema proposed in (Joachims et al., 2009), namely *SVM*^{multiclass} . Then a sequence labeling approach, based on the *SVM*^{hmm} learning algorithm (Altun et al., 2003), will be introduced, as an explicit account of both *conversational* and *topical* contexts.

The multi-class approach. The *SVM*^{multiclass} schema described in (Joachims et al., 2009) is applied[2] to implicitly compare all polarity labels and select the most likely one, using the multi-class formulation described in (Crammer and Singer, 2001). The algorithm thus acquires a specific function $f_y(x)$ for

[2] `http://svmlight.joachims.org/svm_multiclass.html`

each sentiment polarity label $y \in \mathcal{Y}$, where $\mathcal{Y} = \{positive, negative, neutral\}$. Given a feature vector $x \in \mathcal{X}$ representing a tweet t_i, *SVM*^{multiclass} allows to predict a specific polarity $y^* \in \mathcal{Y}$ by applying the discriminant function $y^* = \arg\max_{y \in \mathcal{Y}} f_y(x_i)$, where $f_y(x) = w_y \cdot x$ is a linear classifier associated to each label y. Given a training set $(x_1, y_1) \ldots (x_n, y_n)$, the learning algorithm determines each classifier parameters w_y by solving the following optimization problem:

$$min\ \frac{1}{2} \sum_{i=1\ldots k} \|w_i\|^2 + \frac{C}{n} \sum_{i=1\ldots n} \xi_i \qquad s.t.\ \forall i, \forall y \in \mathcal{Y} : x_i \cdot w_{y_i} \geq x_i \cdot w_y + 100\Delta(y_i, y) - \xi_i$$

where C is a regularization parameter that trades off margin size and training error, while $\Delta(y_i, y)$ is the loss function that returns 0 if y_i equals y, and 1 otherwise.

The markovian approach. The sentiment prediction of a target tweet can be seen as a sequential classification task over a context, and the *SVM*^{hmm} algorithm can be thus applied. Given an input sequence $\mathbf{x} = (x_1 \ldots x_l) \subseteq \mathcal{X}$, where $\mathbf{x}$ is a tweet context, e.g. the *conversational* and the *hashtag-based* one (i.e. $\Lambda_i^{C,l}$ and $\Lambda_i^{H,l}$, respectively) and x_i is a feature vector representing a tweet, the model predicts a tag sequence $\mathbf{y} = (y_1 \ldots y_l) \in \mathcal{Y}^+$ after learning a linear discriminant function $F : \mathcal{P}(\mathcal{X}) \times \mathcal{Y}^+ \rightarrow \mathbb{R}$ over input/output pairs. The labeling $f(\mathbf{x})$ is thus defined as: $f(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}^+} F(\mathbf{x}, \mathbf{y}; \mathbf{w})$. It is obtained by maximizing F over the response variable, $\mathbf{y}$, for a specific given input, $\mathbf{x}$. In these models, F is linear in some combined feature representation of inputs and outputs $\Phi(\mathbf{x}, \mathbf{y})$, i.e. $F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$. As Φ extracts meaningful properties from an observation/label sequence pair $(\mathbf{x}, \mathbf{y})$, in *SVM*^{hmm} it is modeled through two types of features: interactions between attributes of the observation vectors x_i and a specific label y_i (i.e. **emissions** of x_i by y_i) as well as interactions between neighboring labels y_i along the chain (**transitions**). In other words, Φ is defined so that the complete labeling $\mathbf{y} = f(\mathbf{x})$ can be computed efficiently from F, using a *Viterbi-like decoding algorithm*, according to the linear discriminant function

$$\mathbf{y}^* = \arg\max_{\mathbf{y} \in \mathcal{Y}^+} \{ \sum_{i=1\ldots l} [\sum_{j=1\ldots k} (x_i \cdot w_{y_{i-j}\ \ldots\ y_i}) + \Phi_{tr}(y_{i-j}, \ldots, y_i) \cdot w_{tr}] \}$$

In the training phase, *SVM*^{hmm} solves the following optimization problem given training examples $(\mathbf{x}^1, \mathbf{y}^1) \ldots (\mathbf{x}^n, \mathbf{y}^n)$ of sequences of feature vectors $\mathbf{x}^j$ with their correct tag sequences $\mathbf{y}^j$

$$min\ \frac{1}{2}\|w\|^2 + \frac{C}{n} \sum_{i=1\ldots n} \xi_i$$

$$s.t. \quad \forall \mathbf{y}, n : \{ \sum_{i=1\ldots l} (x_i^n \cdot w_{y_i^n}) + \Phi_{tr}(y_{i-1}^n, y_i^n) \cdot w_{tr} \} \geq \{ \sum_{i=1\ldots l} (x_i^n \cdot w_{y_i}) + \Phi_{tr}(y_{i-1}, y_i) \cdot w_{tr} \} + \Delta(y^n, y)$$

where $\Delta(y^n, y)$ is the loss function, computed as the number of misclassified tags in the sequence, $(x_i \cdot w_{y_i})$ represents the emissions and $\Phi_{tr}(y_{i-1}, y_i)$ the transitions. Indeed, through *SVM*^{hmm} learning the label for the target tweet is made dependent on its context history. The markovian setting thus acquires patterns across tweet sequences to recognize sentiment even for truly ambiguous tweets.

4 Experimental Evaluation

The aim of the experiments is to estimate the contribution of the proposed contextual models to the accuracy reachable in different scenarios, whereas rich contexts (e.g. popular hashtags) are possibly made available or just singleton tweets, with no context, are targeted.

We adopted the *"Sentiment Analysis in Twitter"* dataset[3] as it has been made available in the *ACL SemEval-2013* (Nakov et al., 2013). However, in order to rely on tweet identifiers (needed to retrieve contexts from Twitter servers), only the Training and Development portions of the data (11,338 examples), for which id's were made available, have been employed. As about 10,045 tweets were available from the servers,[4] a static split 80/10/10 in *Training/Held-out/Test* respectively, has been carried out as reported in Table 1. As the performance evaluation is always carried out against one target tweet (in analogy with the benchmark of SemEval-2013), the multi-classification may happen when no context is available (i.e. there is no conversation nor hashtag to built the context from) or when a rich conversational or topical context is available. In Table 1 different datasets are shown in columns 2-4, 5-7 and 8-10

[3] http://www.cs.york.ac.uk/semeval-2013/task2/index.php?id=data

[4] Several original messages were no longer reachable during the experiment time of March-June 2013

respectively: the entire corpus of 10,045 is represented in columns 2-4, while 5-7 and 8-10 represents the subsets of target tweets for which a conversational or topical context, respectively, was available. Conversational contexts are available only for 1,391 tweets (columns 5-7), while hashtag-based contexts include 1,912 instances (columns 8-10).

	whole dataset			conversation-filtered			hashtag-filtered		
	train	**dev**	**test**	**train**	**dev**	**test**	**train**	**dev**	**test**
Positive	2984	359	387	454	51	56	621	83	66
Negative	1271	147	142	197	31	24	245	28	22
Neutral	3790	495	470	455	68	55	688	79	80
	8045	1001	999	1106	150	135	1554	190	168

Table 1: Whole dataset composition

As tweets are noisy texts, a pre-processing phase has been applied to improve the quality of linguistic features observable and reduce data sparseness. In particular, a normalization step is applied to each post: fully capitalized words are converted in lowercase; reply marks are replaced with the pseudo-token `USER`, hyperlinks by `LINK`, *hashtags* by `HASHTAG` and emoticons by special tokens[5]. Afterwards, an almost standard NLP chain is applied through the *Chaos* parser (Basili et al., 1998; Basili and Zanzotto, 2002). In particular, each tweet, with its pseudo-tokens produced by the normalization step, is mapped into a sequence of POS tagged lemmas. Emoticons are treated as nouns. In order to feed the LSK, lexical vectors correspond to a Word Space derived from a corpus of about 1.5 million tweets, downloaded during the experimental period and using the topic names from the trial material as query terms. Every word w in such corpus is represented as one co-occurrence vector as in (Sahlgren, 2006) with the setting discussed in (Croce and Previtali, 2010): left and right co-occurrence scores are obtained in a window of size $n = \pm 5$ around each w. Vector components w_f correspond to Pointwise Mutual Information values $pmi(w, f)$ between the word w (the row) and the feature f. Dimensionality reduction is applied to the co-occurrence matrix, through SVD, with a dimensionality cut of $k = 250$.

Existing state-of-the-art approaches neglect the tweet context, so that datasets with labeled contexts are not available: USPK or the markovian approach would not be applicable. The solution consisted in creating a *semi-supervised Gold-Standard* by training the multi-class classifier (not employing any context) fed through a combination of BoWK and LSK Kernel functions and get the classification of all tweets within the context of at least one target tweet. Unfortunately, this can introduce noise, but it is a realistic solution to a cold-start approach, easily portable to other datasets.

Performance scores report the classification accuracy in terms of Precision, Recall and standard F-measure. However, in line with SemEval-2013, we also report the F_1^{pmn} score as the arithmetic mean between the F_1s of *positive*, *negative* and *neutral* classes.

4.1 Experiment 1: Using contexts in a general tweet classification setting

A first experiment has been run to validate the impact of contextual information over generic tweets, independently from the availability of the context. In this case, the entire data set is used. The different settings adopted are reported in independent rows, corresponding to different classification approaches:

- *multi-class* refers to the application of the multi-classification of $SVM^{multiclass}$, that does not require any context and can be considered as a baseline for the employed Kernel combinations;
- *conversation* refers to the SVM^{hmm} classifier observing the conversation-based contexts. The training and testing of the classifier is here run with different *context size*s, by parameterizing l in $\Lambda_i^{C,l}$;
- likewise, *hashtag* refers to the SVM^{hmm} classifier observing the topic-based contexts, when hashtags are considered. Different *context size*s have been considered, by parameterizing l in $\Lambda_i^{H,l}$.

In both *conversation* and *hashtag* models, when no context is available, the SVM^{hmm} classifier acts on a sequence of length one, and no transition is used. Table 2 shows the empirical results over the whole test dataset. The first general outcome is that algorithmic baselines, i.e. context-free models that use no contextual information, in the multi-class rows are better performing whenever richer representations are provided. The LSA information (+8.29%) as well as the user profiling (+10.73%) seem beneficial in

[5]We normalized 113 well-known emoticons in 13 classes.

	Context size	Precision			Recall			$\mathbf{F_1}$			$\mathbf{F_1^{pnn}}$
	l	*pos*	*neg*	*neu*	*pos*	*neg*	*neu*	*pos*	*neg*	*neu*	
BoWK											
multi-class	-	.713	.496	.680	.649	.401	.770	.679	.444	.723	**.615** (-)
conversation	3	.761	.493	.695	.651	.465	.789	.702	.478	.739	.640 (+*4.07%*)
	6	.728	.500	.718	.677	.479	.768	.701	.489	.742	.644 (+*4.72%*)
	∞	.723	.511	.722	.695	.472	.762	.709	.491	.741	**.647 (+*5.20%*)**
hashtag	3	.766	.533	.675	.633	.401	.821	.693	.458	.741	.631 (+*2.60%*)
	6	.727	.575	.711	.682	.514	.770	.704	.543	.740	.662 (+*7.64%*)
	16	.717	.561	.730	.693	.549	.755	.704	.555	.743	**.667 (+*8.46%*)**
	31	.717	.533	.738	.705	.570	.732	.711	.551	.735	.666 (+*8.29%*)
BoWK+LSK											
multi-class	-	.754	.595	.704	.674	.486	.804	.712	.535	.751	**.666** (-)
conversation	3	.759	.595	.712	.682	.486	.811	.718	.535	.758	.670 (+*0.60%*)
	6	.760	.536	.737	.713	.521	.781	.736	.529	.758	.674 (+*1.20%*)
	∞	.774	.554	.717	.682	.542	.791	.725	.548	.752	**.675 (+*1.35%*)**
hashtag	3	.731	.541	.737	.729	.556	.732	.730	.549	.734	.671 (+*0.75%*)
	6	.770	.580	.736	.700	.585	.789	.733	.582	.762	**.693 (+*4.05%*)**
	16	.742	.519	.732	.693	.570	.751	.717	.544	.742	.667 (+*0.15%*)
	31	.751	.537	.729	.685	.556	.774	.716	.547	.751	.671 (+*0.75%*)
BoWK+LSK+USPK											
multi-class	-	.778	.612	.716	.680	.500	.830	.726	.550	.768	**.681** (-)
conversation	3	.771	.563	.689	.625	.507	.817	.690	.533	.748	.657 (*-3.67%*)
	6	.753	.654	.707	.693	.493	.806	.721	.562	.753	**.679 (*-0.29%*)**
	∞	.767	.566	.713	.690	.514	.791	.727	.539	.750	.672 (*-1.32%*)
hashtag	3	.753	.556	.735	.693	.599	.766	.721	.576	.750	.683 (+*0.29%*)
	6	.747	.594	.735	.711	.556	.779	.728	.575	.756	**.686 (+*0.73%*)**
	16	.742	.519	.742	.700	.592	.745	.721	.553	.743	.672 (*-1.32%*)
	31	.738	.530	.739	.693	.556	.766	.715	.543	.752	.670 (*-1.62%*)

Table 2: Evaluation results on whole dataset.

their relative improvements with respect to the simple BoW Kernel accuracy. Second, almost all context-driven models (i.e. SVM^{hmm} operating on different context sizes) improve *wrt* their $SVM^{multiclass}$ counterpart. Every polarity category benefits from the introduction of contexts, although this is particularly true for the negative (*neg*) case, where a 15.5% of the entire dataset examples are available: it seems clear that contexts allow to compensate against poor training conditions.

4.2 Experiment 2: Measuring the full impact of context-based models over rich contexts

Given the above outcomes, a second set of experiments has been run against the subset of the test data restricted to tweets for which rich contexts are available, as introduced in Table 1. In Figure 1, the performances of different learning paradigms and Kernels trained and tested over these corpora are shown. On the Left of the figure, the performance over the conversation-filtered corpus (Table 1) are reported: these tweets are characterized by rich conversational contexts of different increasing sizes on the X-axis. On the Right of Figure 1, the corresponding performances obtained over the hashtag-filtered corpus are reported. As the number of available examples in both test corpora is much smaller, the baselines corresponding to the $SVM^{multiclass}$ approach are lower.

On the contrary, such poorer training evidence does not seem to afflict the contextual models in both corpora, as the markovian modeling seems to bring a straight benefit. In particular, increasing amount of contextual information is usually beneficial to accuracy scores. In general, the SVM^{hmm} accuracy plots seem to increase up to a given context size, that is around 6 for conversational contexts vs. 16 previous tweets for topical contexts. It seems that a wider context (i.e. a window of 8 or 10 tweets) is not so beneficial, as the generalization emphasized by LSK and USPK tends to diverge. Different genres of discussions seem to provide different useful contexts for sentiment detection. The overall benefit reachable by SVM^{hmm} relatively to the $SVM^{multiclass}$ baseline is striking as only rich contexts are used for training and testing. The BoW Kernel over the conversation corpus has an overall relative improvement of 18.26% in F_1^{pnn}, where the richer BoWK+LSK Kernel improves of about 5.94%. Boosts in F_1^{pnn} over topical contexts are more significant: 23.73% for the BoW Kernel vs. 17.93% for BoWK+LSK. This latter Kernel is optimal, suggesting that user profiling requires possibly a richer description that is not entirely captured by the vectors of the user sentiment profile. In fact USPK, when combined with

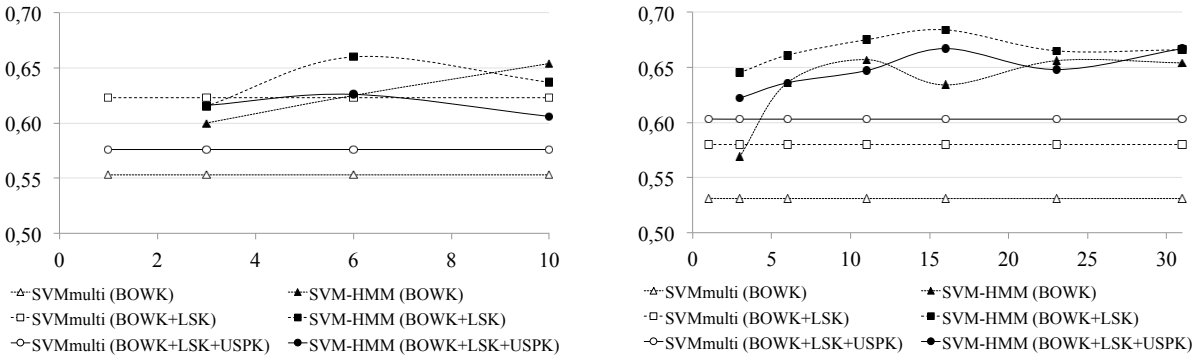


Figure 1: The F_1^{pnn} measure of the different classifiers vs. different context sizes. On the Left: performances when conversational contexts are employed. On the Right: topical contexts are adopted.

BOWK+LSK into the markovian approach, seems to not provide any useful contribution. A clash between the global information (as modeled by the USPK) and the local information (embedded in the recent tweets about a topic) is here observed: when these enter in an opposition, the contrast penalizes the accuracy of the linear combination of Kernels. In general, the improvements implied by contextual information are related to the treatment of particularly ambiguous tweets. In a conversation, such as

`MrGreen` :	*Cannot wait to meet @therealjuicyj and @RealWizKhalifa with @Hill_Gonzz November 29th #trippyniqqas*	$(positive)$
`ColMustard` :	*@MrGreen where they gone be??*	$(neutral)$
`MrGreen` :	*@ColMustard New Orleans!!!*	$(positive)$
`ColMustard` :	*@MrGreen house of blues?*	$(neutral)$
`MrGreen` :	*@ColMustard no it's at the UNO lakefront arena*	$(neutral)$
`ColMustard` :	*@MrGreen I'm going Tuesday to the house of blues to see ASAP Rocky*	$(neutral)$

the switch to a *neutral* mode characterizing the target tweet is a consequence of the entire sequence and captured as a pattern. The contribution of the topical contexts is finally evident in the following example:

	... ...	
`ProfPlum` :	*Can't wait to get out there with my boys Go Team! #goeagles*	$(positive)$
`MrsPeacock` :	*GO my awesome team @WestCoastEagles!!!!! #goeagles #weflyhigh :D*	$(positive)$
`MissScarlet` :	*Let's go eagles :) #goeagles*	$(positive)$
`SergGray` :	*keen for the eagles game today. #goeagles*	$(positive)$

5 Conclusions

In this work the role of contextual information in supervised Sentiment Analysis over Twitter is investigated. While the task is eminently linguistic, as resources and phenomena lie in the textual domain, other semantic dimensions are worth to be explored. In this work, three types of context for a target tweet have been studied. Structured Learning through a markovian approach has been adopted to inject contextual evidence (e.g. the history of preceding posts) in the classification of the most recent, i.e. a target, tweet. The improvement of accuracy in the investigated task are striking as for the large applicability of the approach that does not require additional manually coded resources. The different employed contexts show specific but systematic benefits. On the one side, this proofs the correctness of the initial intuitions. Moreover, the observed relative improvements around 20% over tweets characterized by rich topical or conversational contexts (see Fig. 1) suggest that larger training datasets can even provide better results. In these first experiments, user modeling has only been partially explored, whereas the USPK model does not seem very effective. In fact, USPK seems to express a more static notion of context (i.e. the attitude of the user as observed across a longer period than individual conversations) and two different notions (i.e. information embedded into recent tweets) risk to be incompatible. However, the learning of the optimal Kernel combination as well as a proper history size for the USPK are still worth of deeper investigation. Finally, user interaction dynamics are particularly complex in social networks and deserve better representations about reputation, authority and influence in future explorations.

References

Apoorv Agarwal, Fadi Biadsy, and Kathleen R. Mckeown. 2009. Contextual phrase-level polarity analysis using lexical affect scoring and syntactic n-grams. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 24–32, Stroudsburg, PA, USA. Association for Computational Linguistics.

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, pages 30–38, Stroudsburg, PA, USA. Association for Computational Linguistics.

Y. Altun, I. Tsochantaridis, and T. Hofmann. 2003. Hidden Markov support vector machines. In *Proceedings of the International Conference on Machine Learning*.

Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING (Posters)*, pages 36–44. Chinese Information Processing Society of China.

Roberto Basili and Fabio Massimo Zanzotto. 2002. Parsing engineering and empirical robustness. *Nat. Lang. Eng.*, 8(3):97–120, June.

Roberto Basili, Maria Teresa Pazienza, and Fabio Massimo Zanzotto. 1998. Efficient parsing for information extraction. In *Proc. of the European Conference on Artificial Intelligence*, pages 135–139.

Albert Bifet and Eibe Frank. 2010. Sentiment knowledge discovery in twitter streaming data. In *Proceedings of the 13th International Conference on Discovery Science*, DS'10, pages 1–15, Berlin, Heidelberg. Springer-Verlag.

Giuseppe Castellucci, Simone Filice, Danilo Croce, and Roberto Basili. 2013. Unitor: Combining syntactic and semantic kernels for twitter sentiment analysis. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 369–374, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

K. Crammer and Y. Singer. 2001. On the algorithmic implementation of multi-class svms. *Journal of Machine Learning Research*, 2:265–292.

Nello Cristianini, John Shawe-Taylor, and Huma Lodhi. 2002. Latent semantic kernels. *J. Intell. Inf. Syst.*, 18(2-3):127–152, March.

Danilo Croce and Roberto Basili. 2012. Grammatical feature engineering for fine-grained ir tasks. In Giambattista Amati, Claudio Carpineto, and Giovanni Semeraro, editors, *IIR*, volume 835 of *CEUR Workshop Proceedings*, pages 133–143. CEUR-WS.org.

Danilo Croce and Daniele Previtali. 2010. Manifold learning for the semi-supervised induction of framenet predicates: An empirical investigation. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '10, pages 7–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING (Posters)*, pages 241–249. Chinese Information Processing Society of China.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing*, pages 1–6.

G. Golub and W. Kahan. 1965. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics*, 2(2).

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

Thorsten Joachims, Thomas Finley, and Chun-Nam Yu. 2009. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59.

Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. 2011. Twitter sentiment analysis: The good the bad and the omg! In Lada A. Adamic, Ricardo A. Baeza-Yates, and Scott Counts, editors, *ICWSM*. The AAAI Press.

T. Landauer and S. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104(2):211–240.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

Subhabrata Mukherjee and Pushpak Bhattacharyya. 2012. Sentiment analysis in twitter with lightweight discourse analysis. In *Proceedings of COLING*, pages 1847–1864.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 312–320, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May. European Language Resources Association (ELRA).

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86.

Magnus Sahlgren. 2006. *The Word-Space Model*. Ph.D. thesis, Stockholm University.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA.

Jianfeng Si, Arjun Mukherjee, Bing Liu, Qing Li, Huayi Li, and Xiaotie Deng. 2013. Exploiting topic based twitter sentiment for stock prediction. In *ACL (2)*, pages 24–29.

Peter D. Turney. 2002. Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 417–424, Stroudsburg, PA, USA. Association for Computational Linguistics.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fabio M. Zanzotto, Marco Pennaccchiotti, and Kostas Tsioutsiouliklis. 2011. Linguistic Redundancy in Twitter. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 659–669, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Author Index